Machine Text Detectors are Membership Inference Attacks

Ryuto Koike [*1,2] **Liam Dugan** [*2] **Masahiro Kaneko** [3] **Chris Callison-Burch** [2] **Naoaki Okazaki** [1]

Abstract

We theoretically and empirically demonstrate the *transferability*, i.e., how well a method originally developed for one task performs on the other, between membership inference attacks (MIAs) and machine-generated text detection. We prove that the asymptotically optimal metric is identical for both tasks, unify existing methods under this metric, and hypothesize that how well a method approximates it shapes its transferability. Our large-scale experiments show a strong rank correlation ($\rho \approx 0.7$) in cross-task performance, and notably, a machine text detector achieves the strongest performance among evaluated methods on both tasks. To facilitate cross-task development and fair evaluation, we introduce MINT, a unified evaluation suite implementing 15 recent methods from both tasks. https://github.com/ryuryukke/mint

1. Introduction

Large language models (LLMs) have demonstrated human-level text generation and understanding capabilities, impacting various fields. Despite their positive societal implications, their negative consequences have increasingly been reported. For instance, LLMs may leak personal (Lukas et al., 2023) or copyrighted information (Wei et al., 2024) through memorization of training data (Carlini et al., 2021; Morris et al., 2025). Beyond privacy, LLMs also raise challenges to authorship authenticity, as they can be exploited to mass-produce propaganda (Goldstein et al., 2024) or cheat on student assignments (Guardian, 2025).

Many recent lines of work aim to mitigate such negative implications of LLMs, including MIAs and machine text detection. MIAs classify whether a given text was included in a language model's training data (Carlini et al., 2022; Mattern et al., 2023; Shi et al., 2024), helping identify potential leaks of personal or copyrighted information. In contrast, machine text detection classifies text as human-written or machine-generated (Mitchell et al., 2023; Hans et al., 2024), helping identify misuse of LLMs such as misinformation.

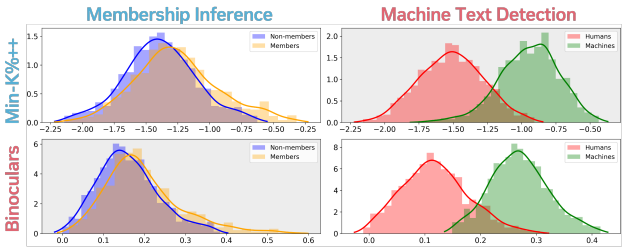

Figure 1. Predicted score distributions of *Min-K%++* (state-of-the-art MIA) and *Binoculars* (state-of-the-art detector) across both tasks. Shaded areas indicate the cross-task setting. The methods from two separate tasks produce **strikingly similar** score distributions within each task, suggesting their transferability.

While the two tasks target different goals, their methods often use similar signals based on a language model's probability distribution. In MIAs, training members tend to have higher likelihoods than non-members; similarly, in detection, machine-generated texts often have higher likelihoods than human-written texts. Thus, both tasks use likelihood or entropy as baselines. Indeed, as shown in Figure 1, our pilot study reveals strikingly similar score distributions of the state-of-the-art MIA and detector across both tasks. Despite this shared property, the two tasks have been studied independently, potentially leading to biased evaluations and conclusions that overlook stronger methods or valuable insights from the other task.

Motivated by this gap, we theoretically and empirically study the *transferability* between MIAs and detection, i.e., how well a method developed for one task performs on the other. Theoretically, we prove that both tasks share the same asymptotically optimal metric: *the likelihood ratio between the target model distribution and the true population distribution*. We further unify many existing methods from both tasks as approximations of this metric and hypothesize that a method's transferability is correlated with how well it approximates the metric.

*Equal contribution; Ryuto Koike did his work while visiting the University of Pennsylvania. [1]Institute of Science Tokyo [2]University of Pennsylvania [3]Mohamed bin Zayed University of Artificial Intelligence. Correspondence to: Ryuto Koike <ryuto.koike@nlp.comp.isct.ac.jp>, Liam Dugan <ldugan@seas.upenn.edu>.

Published at ICML 2026 Workshop on the Impact of Memorization on Trustworthy Foundation Models, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

Empirically, we quantify transferability through large-scale experiments with 7 state-of-the-art MIA methods and 5 state-of-the-art detectors across 13 domains and 10 generators. We evaluate each method on both tasks and compute the rank correlation between their performance rankings. We find that methods effective in one task often remain effective in the other, with a strong rank correlation ($\rho \approx 0.7$) in cross-task performance. Notably, *Binoculars* (Hans et al., 2024), originally designed for detection, achieves the strongest performance among evaluated methods on both tasks, demonstrating the practical impact of transferability.

2. Theoretical Transferability between MIAs and Machine Text Detection

2.1. Task Formulation

Let $\mathcal{X}$ be the set of all token sequences, $P_{\mathcal{Q}}$ the "true" probability distribution of human-written text over $\mathcal{X}$, and $\mathcal{M}$ a language model that induces a distribution $P_{\mathcal{M}}$ over $\mathcal{X}$.

Machine Text Detection. The objective is to determine whether a text $x \in \mathcal{X}$ is written by humans or generated by the model $\mathcal{M}$. This can be formalized as follows:

$$H_0 : x \sim P_{\mathcal{Q}}, \quad H_1 : x \sim P_{\mathcal{M}}.$$

Membership Inference. The objective is to determine whether or not the model $\mathcal{M}$ is trained on a text $x \in \mathcal{X}$. Let $\mathcal{T}$ be a training algorithm, and $\mathcal{D}$ a random training dataset sampled from $P_{\mathcal{Q}}$. This can be formalized as follows, which differs only in whether x is included during training,

$$H_0 : \mathcal{M} \leftarrow \mathcal{T}(\mathcal{D}), \quad H_1 : \mathcal{M} \leftarrow \mathcal{T}(\mathcal{D} \cup \{x\}).$$

The goal of both tasks is to design a test statistic $f(x; \mathcal{M})$ that rejects the respective null hypothesis H_0 with maximal statistical power.

2.2. Unifying MIAs and Machine Text Detection as Likelihood Ratio Tests

Theorem 2.2 (Unified Optimality). Let $\mathcal{X}$ be the set of all token sequences, $\mathcal{M}$ a language model trained to maximize the likelihood of a training set $\mathcal{D} \subset \mathcal{X}$, $P_{\mathcal{M}}$ the probability distribution induced by $\mathcal{M}$, and $P_{\mathcal{Q}}$ a probability distribution over the subset of $\mathcal{X}$ that models the "true" human distribution of text. Then the test statistic

$$\Lambda(x) = \frac{P_{\mathcal{M}}(x)}{P_{\mathcal{Q}}(x)}$$

achieves optimal accuracy at a given false positive rate (Type I error) *for both machine text detection and membership inference* under standard asymptotic regularity conditions.

Proof. We prove this by showing that $\Lambda(x)$ coincides with the likelihood ratio test for both tasks.

Step 1: Machine text detection. Consider testing

$$H_0 : x \sim P_{\mathcal{Q}}, \quad H_1 : x \sim P_{\mathcal{M}}.$$

The likelihood ratio test for this hypothesis is

$$\Lambda_{\text{MGT}}(x) = \frac{P_{\mathcal{M}}(x)}{P_{\mathcal{Q}}(x)}$$

which exactly coincides with the proposed statistic $\Lambda(x)$. By the Neyman-Pearson lemma (Neyman & Pearson, 1933), this test statistic is most powerful at a given Type I error.

Step 2: Membership inference. Consider testing

$$H_0 : \mathcal{M} \leftarrow \mathcal{T}(\mathcal{D}), \quad H_1 : \mathcal{M} \leftarrow \mathcal{T}(\mathcal{D} \cup \{x\}).$$

Following Carlini et al. (2022), membership inference can be formulated as a likelihood ratio test between the distributions over trained models with and without the target data point x,

$$\Lambda_{\text{MI}}(x) = \frac{P(\mathcal{M} \mid H_1)}{P(\mathcal{M} \mid H_0)}.$$

The exact likelihood over models is intractable. Under our assumptions, however, the likelihood ratio depends on the trained model only through the scalar statistic $P_{\mathcal{M}}(x)$, which is sufficient for distinguishing H_0 and H_1 (see Appendix A for a detailed proof of this). Under the asymptotic assumption that $\mathcal{M}$ has infinite capacity and perfectly maximizes the likelihood of the training set, $P_{\mathcal{M}}(x)$ equals the empirical frequency of x in the training set $\mathcal{D}$. Let $K = N \cdot P_{\mathcal{M}}(x)$ denote the number of occurrences of x in the training set, where $N = |\mathcal{D}|$. Then the likelihood ratio further reduces to

$$\Lambda_{\text{MI}}(x) = \frac{P(K \mid H_1)}{P(K \mid H_0)}.$$

Let q_x be shorthand for $P_{\mathcal{Q}}(x)$, we have

$$K \mid H_0 \sim \text{Binomial}(N, q_x),$$
$$K \mid H_1 \sim 1 + \text{Binomial}(N-1, q_x).$$

For any $k \in \{1, \ldots, N\}$,

$$\frac{\Pr[K = k \mid H_1]}{\Pr[K = k \mid H_0]} = \frac{\binom{N-1}{k-1} q_x^{k-1} (1-q_x)^{N-k}}{\binom{N}{k} q_x^{k} (1-q_x)^{N-k}} = \frac{k}{N q_x}.$$

Substituting $k = N \cdot p_x$ with $p_x = P_{\mathcal{M}}(x)$ yields

$$\Lambda_{\text{MI}}(x) = \frac{p_x}{q_x}.$$

Therefore, the likelihood ratio test for membership inference reduces to

$$\Lambda_{\text{MI}}(x) = \frac{P_{\mathcal{M}}(x)}{P_{\mathcal{Q}}(x)}.$$

□

2.3. Classifying Methods as Approximate Likelihood Ratio Tests

Since the true population distribution $P_{\mathcal{Q}}$ is inaccessible, methods in both tasks employ various strategies to approximate it. They can be broadly divided into two approaches:

Approximation via External Reference. The first approach approximates the true distribution by leveraging an external distribution $P_{\mathcal{M}_{\text{ref}}}$ as a surrogate for the true population distribution $P_{\mathcal{Q}}$,

$$P_{\mathcal{Q}}(x) \approx P_{\mathcal{M}_{\text{ref}}}(x).$$

$P_{\mathcal{M}_{\text{ref}}}$ can be another language model (*Reference* (Carlini et al., 2021) from MIAs), a byte-level frequency distribution (*Zlib* (Carlini et al., 2021) from MIAs), a token-level frequency distribution (*DC-PDD* (Zhang et al., 2024) from MIAs), or cross-model entropy (*Binoculars* (Hans et al., 2024) from detection).

Approximation via Text Sampling. The second approach leverages text sampling to approximate the true distribution $P_{\mathcal{Q}}$. The likelihood under the true distribution is approximated by the expected likelihood of multiple perturbations $\tilde{x}$ of the target text x,

$$P_{\mathcal{Q}}(x) \approx \mathbb{E}_{\tilde{x}\sim\phi(\cdot|x)}[P_{\mathcal{M}}(\tilde{x})].$$

where $\phi(\cdot|x)$ is a perturbation function that samples variations of the target text x. This strategy is employed by the *Neighborhood attack* (Mattern et al., 2023) from MIAs, *DetectGPT* (Mitchell et al., 2023), and *Fast-DetectGPT* (Bao et al., 2024) from detection.

We provide a breakdown of the tested methods in our work and, where applicable, include a reformulation to fit the categorization in Table 1. We discuss in more detail in Appendix B the steps we took to get to this general formulation for each example in the author's own notation.

3. Quantifying Transferability between MIAs and Machine Text Detection

3.1. Experimental Setup

Membership Inference. We use the MIMIR dataset (Duan et al., 2024), a large-scale MIA benchmark consisting of **5 domains**[1] in the Pile (Gao et al., 2020): Wikipedia, Pile CC, PubMed Central, ArXiv, and HackerNews. Members and non-members are sampled from the training and test sets of the Pile, and 13-gram filtering is used to ensure no leakage. We target **5 models** of PYTHIA (Biderman et al., 2023) with 160M, 1.4B, 2.8B, 6.7B, and 12B parameters.

[1]To compare the two tasks under a common condition, we focus on textual domains as in machine text detection. See Appendix D for full results on MIAs.

Machine Text Detection. We use the RAID dataset (Dugan et al., 2024), a large-scale detection benchmark consisting of generated text and human-written text in **8 domains**: Wikipedia, News, Abstracts, Recipes, Reddit, Poetry, Books, and Reviews, and **5 models**: GPT-2-XL (Radford et al., 2019), MPT-30B-Chat (Team, 2023), LLaMA-2-70B-Chat (Touvron et al., 2023) as open-source models and ChatGPT (OpenAI, 2023) and GPT-4 (OpenAI et al., 2024) as closed-source models.

Evaluation Measures. To assess empirical transferability, we compute the performance ranking of all unique methods on both tasks using **AUROC** and report **Spearman's rank correlation** to judge how closely the two rankings match. A high rank correlation implies that the strong methods for one task also perform competitively on the other. We provide justifications for these evaluation choices in Appendix C.4.

Methods Tested. We consider **7 MIA methods**: *Reference* (Carlini et al., 2021), *Zlib* (Carlini et al., 2021), *Neighborhood attack* (Mattern et al., 2023), *Min-K%* (Shi et al., 2024), *Min-K%++* (Zhang et al., 2025), *ReCaLL* (Xie et al., 2024), and *DC-PDD* (Zhang et al., 2024), **5 detectors**: *DetectGPT* (Mitchell et al., 2023), *Fast-DetectGPT* (Bao et al., 2024), *Binoculars* (Hans et al., 2024), *DetectLLM* (Su et al., 2023), and *Lastde++* (Xu et al., 2025), and **4 general baselines**: *Loss*, *Rank*, *LogRank*, and *Entropy*. See Appendix B and C for method details and configurations.

Detection Scenarios. For our main rank correlation analysis, we use a **white-box setting** for both tasks, where target model token probabilities are accessible. This enables a common comparison and aligns with our proof, relying on target model probability signals. To further examine transferability in real-world scenarios, we investigate the **black-box setting** for detection, targeting closed-source models such as ChatGPT and GPT-4.[2] As target model logits are unavailable, we use surrogate models (PYTHIA-160M (Biderman et al., 2023) and Llama-3-3.2B (Grattafiori et al., 2024)) and report average detection performance across them.

3.2. Main Results

Substantial Rank Correlation Between the Two Tasks. Figure 2 shows the relationship between the rankings of all methods when evaluated on MIAs and on detection. The rankings are based on average performance on each task. We observe a statistically significant Spearman's correlation of $\rho = 0.66$ ($p < 0.01$) over all 15 methods. This indicates that many methods originally proposed for MIAs also perform well in detection, and vice versa.

[2]Since the training data of such closed-source models is not accessible, true ground truth for MIAs is inherently not feasible in this setting. MIAs are thus evaluated only in a white-box setting.

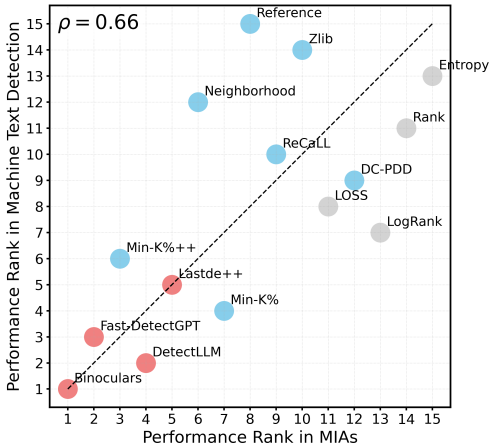

Figure 2. Relationship between method rankings across MIAs and detection. Blue, red, and gray plots indicate MIAs, detectors, and general baselines, respectively. The dashed line denotes equal ranks. Lower rank indicates better performance on each task.

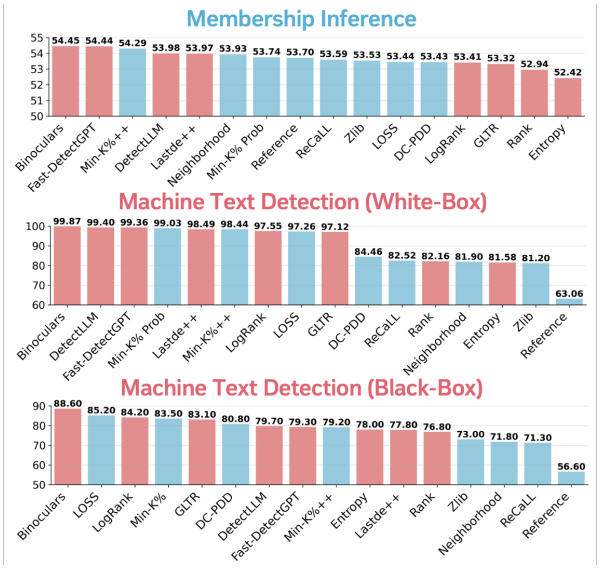

Figure 3. AUROC of MIAs and detectors across both tasks. **Top**: MIA on the MIMIR benchmark, averaged over *five domains* and *five models*. **Middle**: White-box detection on the RAID benchmark, averaged over *eight domains* and *three models*. **Bottom**: Black-box detection on the RAID benchmark, averaged over *eight domains* and *two models*. See Appendix D for full results.

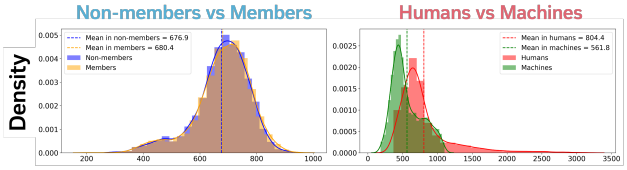

Figure 4. Zlib compression entropy distribution in MIAs and detection, averaged over 3,000 randomly sampled texts for each dataset. The entropy converges between classes in MIAs but diverges in detection.

Superior Performance from the Other Task. Figure 3 shows the average AUROC of all methods evaluated on MIAs (Top) and on detection (Middle). Notably, *Binoculars*, originally a detector, achieves the best average performance on both tasks. This suggests that current evaluations in MIAs may be biased by overlooking stronger methods from detection, potentially leading to conclusions that miss valuable insights. In detection, MIA methods already demonstrate competitive performance. These findings call for greater cross-task awareness and development.

3.3. Analysis

Transferability in Real-World Scenarios. We further assess transferability in real-world scenarios, by evaluating MIA methods on black-box detection for texts generated by ChatGPT and GPT-4. As shown in Figure 3 (Bottom), *Binoculars* still outperforms others by a large margin, MIA methods such as *Min-K%* perform on par with strong detectors. This provides promising evidence of the transferability of MIAs to detection in real-world scenarios.

***Zlib* as an Outlier Illustrates a Task Difference.** We take *Zlib* as an example of limited transferability. It ranks 10th out of 15 methods in MIAs but drops to 14th in detection. *Zlib* calibrates loss by zlib compression entropy, but this signal differs across tasks: MIA compares samples from the same human-written distribution, whereas detection compares human-written and machine-generated texts. Since machine-generated texts are more compressible than human-written ones (Tulchinskii et al., 2023), zlib entropy converges between classes in MIAs but diverges in detection (Figure 4). As a result, loss and zlib entropy shift together in detection, yielding similar *Zlib* scores across classes and weak discriminative signal. This explains its poor transferability and highlights a key task difference: **different prior distributions**. We leave a comprehensive analysis of other methods with limited transferability for future work.

4. Conclusion

We comprehensively study the transferability between MIAs and detection. Our theoretical and empirical results show that the two tasks share the same asymptotically optimal statistic, and many existing methods can be viewed as its approximations. Methods effective on one task also tend to perform well on the other with strong rank correlation. Notably, a detector achieves the strongest performance on both tasks, demonstrating the practical impact of transferability. These findings call for cross-task development and fair evaluation across the two communities.

Impact Statement

Our work aims to establish theoretical and empirical foundations for transferability between MIAs and machine text detection. It highlights the critical need for cross-task awareness and unified evaluation to build stronger defenses against AI privacy risks and misuse. Conversely, such advances could be exploited to optimize evasion strategies against these privacy auditing and detection systems, stimulating further development of more robust methods.

Author Contributions

Ryuto Koike and Liam Dugan co-designed the project and jointly developed its conceptual framing. Ryuto Koike wrote the code, ran the experiments, and drafted the initial version of the paper. Liam Dugan proved the theoretical formulation and contributed to writing and revising the paper. Masahiro Kaneko, Chris Callison-Burch, and Naoaki Okazaki supervised the project and provided valuable feedback and helpful edits to the draft.

Acknowledgements

We thank Shannon Sequiera for helping to check the mathematical derivations and helping improve the clarity of the notation in the general formulation. We are also grateful to Andrew Zhu, Alyssa Hwang, and Youmi Ma for their feedback on early versions of the manuscript.

This research was developed with funding from the Defense Advanced Research Projects Agency's (DARPA) SciFy program (Agreement No. HR00112520300) as well as the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the HIATUS Program contract #2022-22072200005. The views expressed are those of the author and do not reflect the official policies, either expressed or implied, of ODNI, IARPA, DARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

This work was also supported by JSPS KAKENHI Grant Number 25H01137. This work was supported by JST K Program Japan Grant Number JPMJKP24C3. This work was supported by JST SPRING, Japan Grant Number JPMJSP2106. These research results were obtained from the commissioned research (No.22501) by National Institute of Information and Communications Technology (NICT), Japan. We used ABCI 3.0 provided by AIST and AIST Solutions with support from "ABCI 3.0 Development Acceleration Use".

References

Bao, G., Zhao, Y., Teng, Z., Yang, L., and Zhang, Y. Fast-DetectGPT: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

Biderman, S., Schoelkopf, H., Anthony, Q. G., Bradley, H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., Skowron, A., Sutawika, L., and Van Der Wal, O. Pythia: A suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pp. 2397–2430, 2023.

Carlini, N., Tramèr, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, Ú., Oprea, A., and Raffel, C. Extracting training data from large language models. In *30th USENIX Security Symposium*, pp. 2633–2650, 2021.

Carlini, N., Chien, S., Nasr, M., Song, S., Terzis, A., and Tramèr, F. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1897–1914, 2022. doi: 10.1109/SP46214.2022.9833649.

Duan, M., Suri, A., Mireshghallah, N., Min, S., Shi, W., Zettlemoyer, L., Tsvetkov, Y., Choi, Y., Evans, D., and Hajishirzi, H. Do membership inference attacks work on large language models? In *Conference on Language Modeling (COLM)*, 2024.

Dugan, L., Hwang, A., Trhlík, F., Zhu, A., Ludan, J. M., Xu, H., Ippolito, D., and Callison-Burch, C. RAID: A shared benchmark for robust evaluation of machine-generated text detectors. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 12463–12492, 2024.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., Presser, S., and Leahy, C. The Pile: An 800GB dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Goldstein, J. A., Chao, J., Grossman, S., Stamos, A., and Tomz, M. How persuasive is AI-generated propaganda? *PNAS Nexus*, 3(2):pgae034, 02 2024. ISSN 2752-6542. doi: 10.1093/pnasnexus/pgae034.

Grattafiori, A., Dubey, A., Jauhri, A., et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Guardian, T. Revealed: Thousands of UK university students caught cheating using AI, 2025. URL https:

//tinyurl.com/guardian-AI-cheating. Accessed: 2025-07-10.

Hans, A., Schwarzschild, A., Cherepanova, V., Kazemi, H., Saha, A., Goldblum, M., Geiping, J., and Goldstein, T. Spotting LLMs with Binoculars: Zero-shot detection of machine-generated text. In *The Forty-first International Conference on Machine Learning (ICML)*, 2024.

Huffman, D. A. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.

Lukas, N., Salem, A., Sim, R., Tople, S., Wutschitz, L., and Zanella-Béguelin, S. Analyzing leakage of personally identifiable information in language models. In *2023 IEEE Symposium on Security and Privacy (SP)*, pp. 346–363, 2023.

Mattern, J., Mireshghallah, F., Jin, Z., Schoelkopf, B., Sachan, M., and Berg-Kirkpatrick, T. Membership inference attacks against language models via neighbourhood comparison. In *Findings of the Association for Computational Linguistics (ACL)*, pp. 11330–11343, 2023.

Mitchell, E., Lee, Y., Khazatsky, A., Manning, C. D., and Finn, C. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, pp. 24950–24962, 2023.

Morris, J. X., Sitawarin, C., Guo, C., Kokhlikyan, N., Suh, G. E., Rush, A. M., Chaudhuri, K., and Mahloujifar, S. How much do language models memorize? *arXiv preprint arXiv:2505.24832*, 2025.

Neyman, J. and Pearson, E. S. On the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231(694-706):289–337, 1933.

OpenAI. Introducing ChatGPT, 2023. URL https://openai.com/blog/chatgpt. Accessed: 2023-05-10.

OpenAI, Achiam, J., Adler, S., et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2024.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. *OpenAI*, 2019. URL https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf. Accessed: 2024-11-15.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

Shi, W., Ajith, A., Xia, M., Huang, Y., Liu, D., Blevins, T., Chen, D., and Zettlemoyer, L. Detecting pretraining data from large language models. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

Su, J., Zhuo, T., Wang, D., and Nakov, P. DetectLLM: Leveraging log rank information for zero-shot detection of machine-generated text. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Findings of the Association for Computational Linguistics (ACL)*, pp. 12395–12412, 2023.

Team, M. N. Introducing MPT-30B: Raising the bar for open-source foundation models, 2023. URL www.mosaicml.com/blog/mpt-30b. Accessed: 2023-06-22.

Touvron, H., Martin, L., Stone, K., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Tulchinskii, E., Kuznetsov, K., Laida, K., Cherniavskii, D., Nikolenko, S., Burnaev, E., Barannikov, S., and Piontkovskaya, I. Intrinsic dimension estimation for robust detection of AI-generated texts. In *Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

Wang, X., Si, S., and Li, Y. Multiscale diversity entropy: A novel dynamical measure for fault diagnosis of rotating machinery. *IEEE Transactions on Industrial Informatics*, 17(8):5419–5429, 2021. doi: 10.1109/TII.2020.3022369.

Wei, B., Shi, W., Huang, Y., Smith, N. A., Zhang, C., Zettlemoyer, L., Li, K., and Henderson, P. Evaluating copyright takedown methods for language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 139114–139150, 2024.

Xie, R., Wang, J., Huang, R., Zhang, M., Ge, R., Pei, J., Gong, N. Z., and Dhingra, B. ReCaLL: Membership inference via relative conditional log-likelihoods. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 8671–8689, 2024.

Xu, Y., Wang, Y., Bi, Y., Cao, H., Lin, Z., Zhao, Y., and Wu, F. Training-free LLM-generated text detection by mining token probability sequences. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

Zhang, J., Sun, J., Yeats, E., Ouyang, Y., Kuo, M., Zhang, J., Yang, H. F., and Li, H. Min-K%++: Improved baseline for pre-training data detection from large language models. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

Zhang, W., Zhang, R., Guo, J., de Rijke, M., Fan, Y., and Cheng, X. 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 (EMNLP)*, pp. 5263–5274, 2024.

A. Proof that $s = P_{\mathcal{M}}(x)$ is sufficient

In our proof of Theorem 2.2, we transform the likelihood ratio test for the membership inference task as follows:

$$H_0 : \mathcal{M} \leftarrow \mathcal{T}(\mathcal{D}), \quad H_1 : \mathcal{M} \leftarrow \mathcal{T}(\mathcal{D} \cup \{x\}).$$

$$\Lambda_{\text{MI}}(x) = \frac{P(M = \mathcal{M}|H_1)}{P(M = \mathcal{M}|H_0)} = \frac{P(s = P_{\mathcal{M}}(x)|H_1)}{P(s = P_{\mathcal{M}}(x)|H_0)}$$

where $s = P_M(x)$ is a random variable representing the distribution of the likelihood of x under M. For this to be valid, we must show that s is sufficient for M. In other words, we must show that

$$P(M \mid s, H_0) = P(M \mid s, H_1)$$

Let $\mathcal{M}$ be a model following our asymptotic assumptions laid out in Theorem 2.2 and let $N = |\mathcal{D}|$ be the size of its training set. The likelihood of each text $y \in \mathcal{D}$ is defined as

$$P_{\mathcal{M}}(y) = \frac{1}{N}\text{count}(y, \mathcal{D})$$

where $\text{count}(y, \mathcal{D})$ is the number of times a particular text y occurs in $\mathcal{D}$. Let $\hat{\boldsymbol{p}}$ be the vector of likelihoods $P_M(y)$ for all $y \in \mathcal{D}$. This vector contains all information needed to reproduce the full output distribution of $\mathcal{M}$.[3] Thus, $\hat{\boldsymbol{p}}$ is sufficient for M and $P(M \mid \hat{\boldsymbol{p}}, H_0) = P(M \mid \hat{\boldsymbol{p}}, H_1)$.

Now, let $\hat{\boldsymbol{p}}_{\neg x}$ be the vector of likelihoods for all elements not equal to x. In order to show that s is sufficient, we must show that $\hat{\boldsymbol{p}}_{\neg x}$ is distributed identically regardless of the hypothesis. In other words, we want to show that:

$$P_M(y \mid s, H_0) = P_M(y \mid s, H_1) \;\; \forall \;\; y \neq x, \; y \in \mathcal{D}$$

To do this, we first note the difference between our two hypotheses: For H_0, $\mathcal{D}$ was constructed by taking N uniform draws from $P_{\mathcal{Q}}$ and, for H_1, $\mathcal{D}$ was constructed by taking $N - 1$ uniform draws from $P_{\mathcal{Q}}$ and one fixed draw of x. Let $N - \text{count}(x, \mathcal{D})$ be the total number of draws in $\mathcal{D}$ allocated to texts that are not x. We note that, regardless of whether we are in H_0 or H_1, these draws are distributed identically. Thus, for any $y \neq x$ we can model the frequency that y appears in $\mathcal{D}$ via the binomial

$$\text{count}(y, \mathcal{D}) \sim \text{Binomial}(N - \text{count}(x, \mathcal{D}), P_{\mathcal{Q}})$$

Substituting in likelihoods we get that:

$$P_{\mathcal{M}}(y) \sim \frac{1}{N}\text{Binomial}(N - N \cdot P_{\mathcal{M}}(x), P_{\mathcal{Q}})$$

This means that, conditioned on fixed knowledge of $s = P_{\mathcal{M}}(x)$, we get that:

$$P_{\mathcal{M}}(y \mid s, H_0) = P_{\mathcal{M}}(y \mid s, H_1) \;\; \forall \;\; y \neq x, \; y \in \mathcal{D}$$

which implies that $P(M \mid s, H_0) = P(M \mid s, H_1)$ and thus s is a sufficient statistic for our hypothesis test.

B. Details of Methods

B.1. Baselines

(1) ***Loss*** simply uses the target sample x's loss against the model $\mathcal{M}$: $f(\mathbf{x}; \mathcal{M}) = \mathcal{L}(\mathbf{x}; \mathcal{M})$. The hypothesis is that members and machine-generated texts will have a higher likelihood on average than non-members and human-written texts.

(2) ***Entropy*** measures the expected likelihood of the next token given the preceding tokens at each time step under the model's distribution. $f(\mathbf{x}; \mathcal{M}) = \mathbb{E}_{\tilde{x} \sim \mathcal{M}}[\mathcal{L}(\tilde{x}; \mathcal{M})]$. The hypothesis is that machine-generated texts and member texts will have low entropy as the model "understands" the context more and can better model the next token distribution.

[3] Note here that we assume models M_1 and M_2 are equivalent if $P_{M_0}(x) = P_{M_1}(x) \;\; \forall \;\; x \in \mathcal{X}$.

Table 1. Unified formulations of membership inference and machine text detection methods. $P_{\mathcal{M}}(x)$ denotes text likelihood, $\mathcal{R}(x;\mathcal{M})$ denotes average log rank, $\phi(x)$ denotes an arbitrary perturbation function, and $\Phi(x) = x/\sigma_x$ denotes division by the standard deviation. See Appendix B for more details on how we derive each formula.

Method	Task	Equational form
Reference (Carlini et al., 2021)	MIA	$-\log(P_{\mathcal{M}}(x)/P_{\mathcal{M}_{\text{ref}}}(x))$
Zlib (Carlini et al., 2021)	MIA	$-\log P_{\mathcal{M}}(x)/\text{Zlib}(x)$
DetectLLM (Su et al., 2023)	Detection	$\mathbb{E}_{\tilde{x}\sim\phi(x)}[\log R_{\mathcal{M}}(\tilde{x})]/(\log R_{\mathcal{M}}(x))$
ReCall (Xie et al., 2024)	MIA	$\mathbb{E}_{\tilde{x}\sim\phi(x)}[\log P_{\mathcal{M}}(\tilde{x})]/(\log P_{\mathcal{M}}(x))$
DC-PDD (Zhang et al., 2024)	MIA	$\mathbb{E}_{\tilde{x}\sim\mathcal{M}}[-\log P_{\mathcal{M}_{\text{ref}}}(\tilde{x})]$
Binoculars (Hans et al., 2024)	Detection	$(\log P_{\mathcal{M}}(x))/\mathbb{E}_{\tilde{x}\sim\mathcal{M}}[\log P_{\mathcal{M}_{\text{ref}}}(\tilde{x})]$
DetectGPT (Mitchell et al., 2023)	Detection	$\mathbb{E}_{\tilde{x}\sim\phi(x)}[\log(P_{\mathcal{M}}(x)/P_{\mathcal{M}}(\tilde{x}))]$
Neighborhood (Mattern et al., 2023)	MIA	$\mathbb{E}_{\tilde{x}\sim\phi(x)}[\log(P_{\mathcal{M}}(x)/P_{\mathcal{M}}(\tilde{x}))]$
Fast-DetectGPT (Bao et al., 2024)	Detection	$\Phi(\mathbb{E}_{\tilde{x}\sim\phi(x)}[\log(P_{\mathcal{M}}(x)/P_{\mathcal{M}}(\tilde{x}))])$
Min-K% (Shi et al., 2024)	MIA	$\frac{1}{k}\sum_{i\in\text{min-}k\%}(-\log P_{\mathcal{M}}(x_i))$
Min-K%++ (Zhang et al., 2025)	MIA	$\frac{1}{k}\sum_{i\in\text{min-}k\%}\Phi(-\log P_{\mathcal{M}}(x_i) + \mathbb{E}_{\tilde{x}_i\sim\mathcal{M}}[\log P_{\mathcal{M}}(\tilde{x}_i)])$
Lastde (Xu et al., 2025)	Detection	$(-\log P_{\mathcal{M}}(x))/\text{StdDev}(\{\text{DE}(x,\tau)\}_{\tau=1}^{\tau'})$
Lastde++ (Xu et al., 2025)	Detection	$\Phi(\text{Lastde}(x) - \mathbb{E}_{\tilde{x}\sim\phi(x)}[\text{Lastde}(\tilde{x})])$

(3) ***Rank*** measures the average rank of the next token in the model's probability distribution at each time step. $f(x;\mathcal{M}) = \frac{1}{n}\sum_{i=1}^{n}\text{Rank}(x_i;\mathcal{M})$. The hypothesis is that generated text and member text will have a higher average rank than human-written text.

(4) ***LogRank*** measures the average log rank of the next token: $f(x;\mathcal{M}) = \frac{1}{n}\sum_{i=1}^{n}\log(\text{Rank}(x_i;\mathcal{M}))$. This metric smooths out contributions from very low rank tokens to reflect the probabilistic nature of rank information. Similarly to the previous metric, the hypothesis is that generated text and member text have a high average log rank.

B.2. Membership Inference Attacks

(1) ***Reference*** (Carlini et al., 2021) uses the difference in the target sample x's loss between the model $\mathcal{M}$ and another reference model $\mathcal{M}_{\text{ref}}$. We follow the original author's implementation by taking a smaller-size reference model for each of the models we tested: $f(\mathbf{x};\mathcal{M}) = \mathcal{L}(\mathbf{x};\mathcal{M}) - \mathcal{L}(\mathbf{x};\mathcal{M}_{\text{ref}})$. In our notation, this can be represented as $-\log(P_{\mathcal{M}}(x)/P_{\mathcal{M}_{\text{ref}}}(x))$. This method falls into the external reference category of likelihood ratio approximation.

(2) ***Zlib*** (Carlini et al., 2021) employs the ratio of $\mathcal{L}(\mathbf{x};\mathcal{M})$ and the zlib compression score of the target sample x: $f(\mathbf{x};\mathcal{M}) = \mathcal{L}(\mathbf{x};\mathcal{M})\ /\ \text{zlib}(\mathbf{x})$. The zlib compression score is computed by constructing a dictionary of repeated substrings from the text and encoding each dictionary entry into a string of bits using Huffman Coding (Huffman, 1952). The compression rate is thus a representation of the entropy of the empirical substring distribution of the text. Thus the zlib method can be equivalently rewritten as $f(\mathbf{x};\mathcal{M}) = -\log P_{\mathcal{M}}(\mathbf{x})\ /\ \mathbb{E}_{x\sim\mathcal{M}_{\text{ref}}}[-\log P_{\mathcal{M}_{\text{ref}}}(\mathbf{x})]$ where $\mathcal{M}_{\text{ref}}$ represents that empirical substring distribution. We see here that this method falls into the external reference category of likelihood ratio approximation.

(3) ***Neighborhood attack*** (Mattern et al., 2023) compares $\mathcal{L}(\mathbf{x};\mathcal{M})$ to the average loss of *neighborhood samples* $\tilde{\mathbf{x}}$, which are samples crafted by perturbing the target sample x: $f(\mathbf{x};\mathcal{M}) = \mathcal{L}(\mathbf{x};\mathcal{M}) - \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(\tilde{\mathbf{x}}^{(i)};\mathcal{M})$. It hypothesizes that members will show a lower loss compared to their neighborhood samples. Since the quantity $\mathbb{E}_{\tilde{x}\sim\phi(x)}[\log(P_{\mathcal{M}}(x)/P_{\mathcal{M}}(\tilde{x}))]$, we consider this method to be in the text sampling category of likelihood ratio approximation.

(4) ***Min-K%*** (Shi et al., 2024) calculates the average of log-likelihood of the $k\%$ tokens with lowest probabilities: $f(\mathbf{x};\mathcal{M}) = \frac{1}{k}\sum_{x_i\in\text{min-}k(\mathbf{x})}\log p(x_i \mid x_{<i};\mathcal{M})$. The intuition is that members will include fewer outlier tokens with low probability compared to non-members.

(5) ***Min-K%++*** (Zhang et al., 2025) computes the average of the log-likelihood of the $k\%$ tokens with lowest probabilities, where each value is standardized over the model's vocabulary: $f(\mathbf{x};\mathcal{M}) = \frac{1}{k}\sum_{x_i\in\text{min-k\%}}(\log p(x_i \mid x_{<i};\mathcal{M}) - \mu_{x_{<i}})/\sigma_{x_{<i}}$. In our notation, this can be written as, $\frac{1}{k}\sum_{i\in\text{min-}k\%}\Phi(-\log P_{\mathcal{M}}(x_i) + \mathbb{E}_{\tilde{x}_i\sim\mathcal{M}}[\log P_{\mathcal{M}}(\tilde{x}_i)])$, for brevity in our Table 1, we use $\Phi(x) = x/\sigma_x$ as shorthand for the normalization by the standard deviation. Intuitively, this metric measures how unexpectedly surprising a particular token is over the vocabulary distribution.

(6) ***ReCaLL*** (Xie et al., 2024) computes the relative conditional log-likelihood between x and $P \oplus x$ where P is a set of non-member examples $P = p_1 \oplus \cdots \oplus p_n$. It hypothesizes that non-members will have lower log-likelihoods than members, given a non-member context. We represent this in our table as $\mathbb{E}_{\tilde{x}\sim\phi(x)}[\log P_{\mathcal{M}}(\tilde{x})]/(\log P_{\mathcal{M}}(x))$ where $\phi(x) = P \oplus x$. We consider this method to be in the likelihood ratio by text sampling category.

(7) ***DC-PDD*** (Zhang et al., 2024) computes the cross-entropy between the token likelihoods under the model $\mathcal{M}$ and the empirical laplace-smoothed unigram token frequency distribution under some reference corpus $\mathcal{D}'$. In the authors' notation this is $f(\mathbf{x}; \mathcal{M}) = -\frac{1}{n}\sum_{i=1}^{n} p(x_i; \mathcal{M}) \cdot \log p(x_i; \mathcal{D}')$ where $p(x_i; \mathcal{D}') = \frac{\text{count}(x_i)+1}{N'+|V|}$. In our table, we represent this metric equivalently as $\mathbb{E}_{\tilde{x}\sim\mathcal{M}}[-\log P_{\mathcal{M}_{\text{ref}}}(\tilde{x})]$ where $\mathcal{M}_{\text{ref}}$ represents the unigram token frequency distribution under $\mathcal{D}'$.

B.3. Machine-generated Text Detectors

(1) ***DetectGPT*** (Mitchell et al., 2023) computes the degree to which the log-likelihood function under the suspected model has negative curvature for the given target text. They do this by perturbing the sequence using a T5 model (Raffel et al., 2019) and evaluating the change in probability. The functional form looks like $f(x; \mathcal{M}) = \mathbb{E}_{\tilde{x}\sim\phi(x)}[\log(P_{\mathcal{M}}(x)/P_{\mathcal{M}}(\tilde{x}))]$. We directly lift this notation for use in our Table 1. This metric is an example of a text sampling based likelihood ratio approximation.

(2) ***Fast-DetectGPT*** (Bao et al., 2024) forgoes the expensive perturbation approach used by DetectGPT in favor of using the token likelihoods of the perturbation model directly to compute the expectation. They also divide by the mean of sample variances to further smooth out the metric. The full formulation is

$$f(x; \mathcal{M}) = \frac{\log p(x; \mathcal{M}) - \mathbb{E}_{\tilde{x}\sim q(x)}[\log p(\tilde{x}; \mathcal{M})]}{\sqrt{\mathbb{E}_{\tilde{x}\sim q(x)}[(\log p(\tilde{x}; \mathcal{M}) - \mathbb{E}_{\tilde{x}\sim q(x)}[\log p(\tilde{x}; \mathcal{M})])^2]}}$$

We report this using the shorthand $\Phi(x) = x/\sigma_x$ in our Table 1 to represent this quantity. We consider this metric another example of the text sampling based likelihood ratio approach.

(3) ***Binoculars*** (Hans et al., 2024) computes the ratio of the perplexity to the cross entropy of the text under some reference model $\mathcal{M}_{\text{ref}}$. The full formulation using the authors' notation is as follows:

$$B_{\mathcal{M}_1,\mathcal{M}_2}(s) = \frac{\sum_{i=1}^{L} \log\left(\mathcal{M}_1(s)_i\right)}{\sum_{i=1}^{L} \mathcal{M}_1(s)_i \cdot \log\left(\mathcal{M}_2(s)_i\right)}$$

We re-formulate this objective equivalently in our own notation as:

$$(\log P_{\mathcal{M}}(x))/\mathbb{E}_{\tilde{x}\sim\mathcal{M}}[\log P_{\mathcal{M}_{\text{ref}}}(\tilde{x})]$$

The intuition behind this metric is that it measures how much more likely a given text is than what we would expect according to some reference model. We consider this metric a case of approximation via reference model.

(4) ***DetectLLM*** (Su et al., 2023) is a variant of DetectGPT that uses the log rank as the core quantity to test rather than the log likelihood. The authors propose two metrics, Log-Likelihood Log-Rank Ratio (LRR) and Normalized Log-Rank Perturbation (NPR). For the purposes of our work we consider the NPR metric which has superior performance and is characterized by the following formulation:

$$\text{NPR} = \frac{\frac{1}{n}\sum_{p=1}^{n} \log r_\theta(\tilde{x}_p)}{\log r_\theta(x)}$$

where $r_\theta(x_i)$ represents the rank of the token x in the model's output distribution. We reformulate this equivalently as

$$\frac{\mathbb{E}_{\tilde{x}\sim\phi(x)}[\mathcal{R}(\tilde{x}; \mathcal{M})]}{\mathcal{R}(x; \mathcal{M})}$$

using the notation $\mathcal{R}(x; \mathcal{M})$ to denote the log rank similar to how $\mathcal{L}(x; \mathcal{M})$ denotes the negative log likelihood. While this metric doesn't quite fall cleanly into our likelihood ratio categorization (since it does not compute likelihood) we nonetheless note the strong similarities between this metric and other approximate likelihood ratios.

(5) ***Lastde++*** (Xu et al., 2025) utilizes a quantity known as multi-scale diversity entropy (MDE) (Wang et al., 2021) to measure the local fluctuations in likelihood across a particular text sequence. Their metric is:

$$\text{Lastde}(x;\mathcal{M}) = \frac{\mathcal{L}(x;\mathcal{M})}{\text{StdDev}(\{\text{DE}(s,\varepsilon,1),...,\text{DE}(s,\varepsilon,\tau')\})}$$

$$\text{DE}(s,\varepsilon,\tau) = -\frac{1}{\ln\varepsilon}\sum_{i=1}^{\varepsilon} P_i^{(\tau)} \ln P_i^{(\tau)}$$

Where $P_i^{(\tau)}$ measures the *diversity* of text, i.e. the extent to which adjacent segments of tokens have similar probability sequences (see (Xu et al., 2025)). Intuitively the Lastde metric can be thought of as comparing likelihood to the expected diversity. Lastde++ takes this quantity and applies a sampling based perturbation and normalization similar to FastDetectGPT (Bao et al., 2024).

$$\text{Lastde++}(x) = \frac{\text{Lastde}(x;\mathcal{M}) - \mathbb{E}_{\tilde{x}\sim\phi(x)}[\text{Lastde}(\tilde{x},\mathcal{M})]}{\sqrt{\mathbb{E}_{\tilde{x}\sim\phi(x)}[(\text{Lastde}(\tilde{x},\mathcal{M}) - \mathbb{E}_{\tilde{x}\sim\phi(x)}[\text{Lastde}(\tilde{x},\mathcal{M})])^2]}}$$

This can be thought of as measuring whether or not the quantity tracked by $\text{Lastde}(x)$ is at a local maximum for the particular sequence of text.

We consider neither Lastde++ or Lastde to be approximating likelihood ratios as the diversity-entropy (DE) metric seems to be more intuitively thought of as measuring variance rather than likelihood. We leave to future work a more thorough analysis of these variance-based metrics.

C. Implementation Details

C.1. MINT

We release a unified evaluation suite for membership inference attacks (MIAs) and machine-generated text detection: `https://anonymous.4open.science/r/mint-B44A/`. It provides a common framework for evaluation across the two tasks. Some key features of MINT include:

- Support for 15 recent methods and commonly used benchmarks across both tasks.
- A modular class design with helper functions for easy integration of new methods.
- An efficient inference pipeline with shared likelihood caching for multiple methods.

Our experiments were executed using Python 3.9.1 and PyTorch 2.5.1 (CUDA 12.6, NVIDIA Driver 560.28.03) with an Intel Xeon Silver 4314 CPU, 300GB of RAM, and four NVIDIA RTX A6000 GPUs running openSUSE Leap 15.6. The total compute cost was approximately 350 GPU-hours.

C.2. Membership Inference Attacks

Reference (Carlini et al., 2021) Following the original author's implementation, we take a smaller-size reference model for each target model. In MIAs, we use PYTHIA-70M for all target models. In white-box machine text detection, we use GPT2-Small as the smaller model for GPT2-XL, MPT-7B-Chat for MPT-30B-Chat, and LLaMA-7B-Chat for LLaMA-70B-Chat. In black-box machine text detection, we use LLaMA-3-1B as the smaller model for LLaMA-3-3.2B and PYTHIA-70M for PYTHIA-160M.

Neighborhood attack (Mattern et al., 2023) We use the repository[4] default settings, namely T5-Large as mask filling model and 0.3 as the masking rate, across MIAs, white-box machine text detection, and black-box machine text detection.

Min-K% (Shi et al., 2024) We use the setting that was found to ensure the favorable performance in the original paper, namely $k = 20$ as the percent of tokens with the lowest probabilities. Likewise, the token percentage k for Min-K%++ (Zhang et al., 2025) is also set as $k = 20$ for fair comparison.

[4] `https://github.com/mireshghallah/neighborhood-curvature-mia`

ReCaLL (Xie et al., 2024) We set the number of prefixes to $n = 10$, which has been shown in the original paper to yield favorable performance. In MIAs, for each domain, we retrieve non-members as prefixes from the Pile dataset, excluding those in the MIMIR benchmark. To adapt ReCaLL to machine text detection, we use human-written texts as prefixes, since they belong to the negative class. For each domain, these human-written prefixes are retrieved from a subset of the RAID benchmark that was not used in our test set.

DC-PDD (Zhang et al., 2024) Following the official GitHub repository[5], we take a subset of C4 (Raffel et al., 2019) and build a token frequency distribution with the tokenizer of each target model in MIAs and white-box machine text detection, and of each surrogate model in black-box detection.

C.3. Machine Text Detectors

DetectGPT (Mitchell et al., 2023) We follow the repository[6] default settings, namely T5-Large as mask filling model and 0.3 as the masking rate, across MIAs, white-box machine text detection, and black-box machine text detection.

Fast-DetectGPT (Bao et al., 2024) Following the original paper, we employ the target model as both the scoring and perturbation model in MIAs and white-box machine text detection. In black-box detection, we instead use surrogate models as those (see §3.1).

Binoculars (Hans et al., 2024) is reported to work best when the target and reference models are similar in performance. Following the original setting, we use the official code[7] to compute perplexity under a reference model. For MIAs, since the PYTHIA does not provide corresponding chat versions for each model size, we use the PYTHIA-deduped models as references. In white-box detection, we use GPT2-XL-Chat [8] for GPT2-XL, MPT-30B for MPT-30B-Chat, and LLaMA-2-70B for LLaMA-2-70B-Chat. In black-box detection, we adopt LLaMA-3.2-3B-instruct for LLaMA-3-3.2B and PYTHIA-160M-deduped for PYTHIA-160M.

DetectLLM (Su et al., 2023) In line with Fast-DetectGPT, we utilize the target model as both the scoring and perturbation model in MIAs and white-box detection. For black-box detection, we instead use surrogate models (see §3.1).

Lastde++ (Xu et al., 2025) In our implementation, we use the official code and use the repository[9] default settings of the sliding window size $s = 4$, the interval precision $\epsilon = 8$, and the number of scales $\tau' = 15$.

C.4. Evaluation Metrics

We use AUROC for both tasks, as it reflects a method's overall performance and provides a more reliable measure of transferability than metrics based on a single threshold. To quantify empirical transferability, we use Spearman's rank correlation coefficient. This choice is motivated by our proof: since both tasks share the same optimal metric, methods would be expected to preserve their relative rankings across tasks. In contrast, absolute performance is not directly comparable across tasks since their underlying data distributions inherently differ. When computing the rank correlation, since the *Neighborhood attack* and *DetectGPT* share the same formulation, they are treated as a single method.

D. Full Performance on Membership Inference and Machine Text Detection

Table 2, Table 3, and Table 4 report the full results on membership inference, white-box machine text detection, and black-box machine text detection, respectively.

[5] https://github.com/zhang-wei-chao/DC-PDD
[6] https://github.com/eric-mitchell/detect-gpt
[7] https://github.com/ahans30/Binoculars
[8] lgaalves/gpt2-xl_lima
[9] https://github.com/TrustMedia-zju/Lastde_Detector

Table 2. Full comparison of membership inference performances (AUROC) of MIA methods and machine text detectors on the MIMIR benchmark with 13-gram deduplication. Target models are PYTHIA with 160M, 1.4B, 2.8B, 6.9B, and 12B parameters. Gray, blue, and red areas indicate general baseline methods, MIA methods, and machine text detectors, respectively. Textual domains are the bolded columns (Wikipedia, Pile CC, PubMed, ArXiv, HackerNews).

Method	**Wikipedia**					GitHub					**Pile CC**					**PubMed Central**				
	160M	1.4B	2.8B	6.9B	12B	160M	1.4B	2.8B	6.9B	12B	160M	1.4B	2.8B	6.9B	12B	160M	1.4B	2.8B	6.9B	12B
Loss	51.2	53.4	54.1	55.6	56.5	76.3	80.2	81.4	82.7	83.6	50.1	51.0	51.2	52.1	52.7	50.9	52.1	52.7	53.4	54.0
Rank	49.7	52.9	53.9	56.2	57.5	70.0	74.6	75.7	77.4	77.8	50.9	51.6	51.8	52.2	52.1	51.9	52.3	52.4	53.0	53.6
LogRank	51.3	53.5	54.3	55.6	56.9	75.9	80.0	81.2	82.5	83.3	50.3	51.0	51.2	52.3	52.7	50.8	51.8	52.4	53.3	53.7
Entropy	50.9	52.1	52.6	53.1	53.4	76.3	79.7	80.7	81.9	82.7	49.6	50.2	50.5	50.9	51.3	51.7	51.8	52.1	52.2	52.3
Reference	51.7	54.4	55.2	57.4	58.5	37.3	41.0	41.8	43.2	43.6	50.9	52.7	52.8	53.9	54.5	49.4	52.2	52.6	53.5	54.1
Zlib	50.4	53.1	54.0	55.7	56.7	79.7	82.9	83.9	85.0	85.7	51.1	52.1	52.3	53.2	53.6	51.5	52.6	53.1	53.7	54.2
Neighborhood	51.2	54.4	54.8	56.0	57.7	75.4	74.7	74.1	74.8	74.9	51.4	52.7	53.3	54.8	54.7	52.6	55.0	55.8	56.5	57.0
Min-K%	50.6	53.5	54.7	56.7	57.8	75.2	79.8	81.0	82.5	83.4	50.7	51.4	51.6	52.5	52.9	51.4	52.6	53.0	53.9	54.8
Min-K%++	51.2	55.2	56.4	59.6	60.7	73.2	78.2	79.7	81.1	82.4	50.9	52.4	52.2	54.0	54.7	50.6	52.2	52.8	54.3	55.2
ReCaLL	50.5	54.2	54.6	57.2	57.7	72.2	77.3	79.6	80.6	81.9	48.2	49.4	50.7	51.5	51.2	52.1	52.6	55.1	54.9	56.0
DC-PDD	52.4	53.9	54.5	55.8	56.4	82.1	85.2	86.2	86.9	87.6	50.8	52.6	52.7	53.3	53.6	50.5	51.7	52.5	52.9	53.2
DetectGPT	51.2	54.4	54.8	56.0	57.7	75.4	74.7	74.1	74.8	74.9	51.4	52.7	53.3	54.8	54.7	52.6	55.0	55.8	56.5	57.0
Fast-DetectGPT	51.9	54.9	56.3	60.0	62.9	57.8	67.2	69.6	71.4	72.3	51.8	54.2	53.9	55.6	56.1	49.1	51.7	52.8	55.1	56.4
Binoculars	51.7	55.2	56.7	58.5	60.6	71.8	77.2	74.5	80.3	81.7	51.2	53.6	54.5	55.1	55.0	50.4	52.4	53.0	55.2	55.9
DetectLLM	51.6	54.0	55.4	58.3	61.4	56.4	66.8	69.6	71.3	71.8	52.2	53.7	53.5	55.5	55.8	49.0	51.3	52.5	54.8	55.7
Lastde++	50.8	54.2	55.7	59.3	61.4	52.8	64.7	66.8	68.6	69.7	50.9	52.3	52.7	54.6	54.5	50.7	52.2	53.0	54.6	56.4

Method	**ArXiv**					DM Mathematics					**HackerNews**					**Avg. (textual domains)**				
	160M	1.4B	2.8B	6.9B	12B	160M	1.4B	2.8B	6.9B	12B	160M	1.4B	2.8B	6.9B	12B	160M	1.4B	2.8B	6.9B	12B
Loss	54.6	55.8	56.4	57.5	58.1	67.8	67.5	67.2	67.3	67.3	50.5	51.8	52.6	53.4	54.2	51.5	52.8	53.4	54.4	55.1
Rank	52.0	52.4	52.6	54.6	55.0	60.6	60.4	60.7	60.6	60.6	51.7	52.7	52.5	53.6	54.5	51.2	52.4	52.6	53.9	54.5
LogRank	54.4	55.6	56.1	57.5	57.9	66.3	66.4	66.2	66.4	66.2	50.4	51.7	52.4	53.7	54.3	51.4	52.7	53.3	54.5	55.1
Entropy	54.8	55.4	55.1	55.7	55.7	68.4	67.6	67.4	67.2	67.1	50.5	52.2	52.0	52.1	52.1	51.5	52.3	52.5	52.8	53.0
Reference	51.2	53.3	54.1	55.8	56.8	45.0	44.8	44.4	44.4	44.3	50.0	52.1	53.8	55.2	56.6	50.6	52.9	53.7	55.2	56.1
Zlib	54.2	55.3	55.7	56.7	57.2	64.6	64.7	64.6	64.6	64.6	51.2	51.9	52.4	52.9	53.4	51.7	53.0	53.5	54.4	55.0
Neighborhood	53.1	54.7	54.2	54.9	55.0	53.3	51.8	53.1	50.8	53.3	51.1	51.0	51.8	51.9	52.6	51.0	52.2	52.6	53.4	53.9
Min-K%	53.3	55.0	55.8	57.3	58.4	64.7	65.2	64.9	65.1	65.1	50.9	51.8	53.1	54.3	55.3	51.4	52.9	53.6	54.9	55.8
Min-K%++	51.3	53.8	55.9	56.9	59.9	58.8	57.9	58.7	58.4	58.2	51.1	51.5	53.1	54.6	56.4	51.0	53.0	54.1	55.9	57.4
ReCaLL	53.4	54.5	55.4	57.3	58.3	58.0	56.4	56.8	53.7	53.1	52.6	52.4	52.4	53.7	54.0	51.4	52.6	53.6	54.9	55.4
DC-PDD	54.7	56.3	56.3	57.4	57.7	63.9	63.8	63.5	63.4	63.6	49.4	51.0	51.4	52.1	52.7	51.6	53.1	53.5	54.3	54.7
DetectGPT	53.1	54.7	54.2	54.9	55.0	53.3	51.8	53.1	50.8	53.3	51.1	51.0	51.8	51.9	52.6	51.0	52.2	52.6	53.4	53.9
Fast-DetectGPT	51.5	53.2	54.9	57.4	59.3	52.4	53.1	54.2	54.0	54.7	50.1	49.5	51.9	54.1	56.4	50.9	52.7	54.0	56.4	58.2
Binoculars	54.1	54.6	54.9	57.6	59.9	55.7	53.8	53.5	52.1	52.8	49.6	50.4	51.5	53.8	55.6	51.3	53.3	54.2	56.1	57.5
DetectLLM	51.5	53.1	54.7	57.6	58.6	51.9	51.8	53.2	53.6	53.0	49.8	49.2	51.1	53.6	55.5	50.8	52.3	53.4	56.0	57.4
Lastde++	50.8	52.9	54.7	56.1	58.1	52.3	51.6	52.1	52.2	52.3	50.0	51.1	52.0	54.0	56.0	50.6	52.5	53.6	55.7	57.3

Table 3. Full comparison of white-box machine text detection performances (AUROC) of MIA methods and machine text detectors on the RAID benchmark. Target models are GPT-2: GPT-2-XL, MPT: MPT-30B-Chat, LLaMA: LLaMA-2-70B-Chat. Gray, blue, and red areas indicate general baseline methods, MIA methods, and machine text detectors, respectively.

Method	Abstracts			Books			News			Poetry		
	GPT-2	MPT	LLaMA	GPT-2	MPT	LLaMA	GPT-2	MPT	LLaMA	GPT-2	MPT	LLaMA
Loss	97.4	98.6	100.0	98.6	99.9	100.0	94.6	99.9	100.0	96.8	98.3	96.2
Rank	98.2	53.9	93.8	99.4	91.9	96.5	95.5	70.3	86.1	98.0	85.6	88.8
LogRank	98.5	96.4	100.0	99.4	99.9	100.0	96.5	99.7	100.0	97.6	98.3	95.6
Entropy	61.6	55.0	99.9	60.7	97.4	99.9	38.9	92.3	99.9	89.3	90.3	95.4
Reference	21.7	52.4	53.5	59.7	81.7	65.2	45.9	97.8	81.4	40.8	88.7	73.6
Zlib	87.1	47.0	99.9	64.1	66.8	96.0	55.8	80.4	99.7	73.4	73.9	87.6
Neighborhood	95.3	58.4	94.7	92.1	64.8	86.3	85.3	67.7	88.9	83.0	73.2	76.2
Min-K%	99.8	98.8	100.0	99.9	99.9	100.0	99.4	99.9	100.0	99.3	99.0	96.8
Min-K%++	99.6	99.6	97.1	99.8	99.0	98.2	99.7	99.5	98.9	99.4	99.1	90.3
ReCaLL	62.6	87.1	99.6	81.1	95.3	97.8	68.8	89.9	93.3	76.9	87.7	97.6
DC-PDD	48.6	95.3	100.0	64.1	99.8	99.8	47.4	99.3	99.9	82.5	97.0	94.2
DetectGPT	95.3	58.4	94.7	92.1	64.8	86.3	85.3	67.7	88.9	83.0	73.2	76.2
Fast-DetectGPT	99.7	100.0	99.8	99.8	100.0	99.9	99.7	100.0	100.0	99.3	100.0	94.4
Binoculars	99.8	99.9	100.0	99.8	100.0	100.0	99.8	99.9	100.0	99.5	100.0	100.0
DetectLLM	99.7	100.0	100.0	99.8	100.0	99.8	99.7	100.0	100.0	99.4	99.9	93.2
Lastde++	99.7	100.0	97.8	99.8	100.0	98.8	99.6	100.0	100.0	99.4	99.9	88.8

Method	Recipes			Reddit			Reviews			Wikipedia		
	GPT-2	MPT	LLaMA	GPT-2	MPT	LLaMA	GPT-2	MPT	LLaMA	GPT-2	MPT	LLaMA
Loss	70.4	100.0	100.0	97.7	97.4	99.7	97.9	99.9	99.9	91.4	100.0	99.9
Rank	82.3	48.8	17.5	98.3	73.9	64.6	98.8	82.8	95.3	97.0	73.4	81.3
LogRank	71.5	99.9	100.0	98.3	96.2	99.7	98.8	99.9	99.9	95.5	99.9	99.8
Entropy	40.2	98.4	100.0	72.3	78.3	99.7	59.0	98.6	99.9	37.6	94.5	99.1
Reference	34.9	84.7	12.6	51.8	53.2	85.4	60.4	78.0	85.9	46.8	92.6	64.9
Zlib	61.0	98.0	99.9	96.0	50.9	99.6	60.3	65.9	99.2	87.4	99.3	99.8
Neighborhood	82.5	58.2	89.2	97.5	57.0	92.1	90.6	70.8	97.2	93.7	82.4	88.6
Min-K%	89.4	100.0	100.0	99.3	96.6	99.7	99.7	99.9	100.0	99.4	100.0	99.9
Min-K%++	98.9	98.8	99.2	99.3	96.9	91.7	99.4	99.7	99.5	99.6	99.6	99.7
ReCaLL	60.2	97.1	100.0	40.5	87.8	98.6	67.4	89.4	95.8	37.0	71.1	97.9
DC-PDD	41.3	99.9	100.0	62.9	97.7	99.9	62.2	99.8	99.9	38.3	98.3	98.9
DetectGPT	82.5	58.2	89.2	97.5	57.0	92.1	90.6	70.8	97.2	93.7	82.4	88.6
Fast-DetectGPT	99.1	100.0	100.0	99.8	99.9	94.5	99.6	99.8	99.6	99.8	100.0	100.0
Binoculars	99.6	100.0	100.0	99.7	99.1	100.0	99.8	99.9	100.0	99.8	100.0	100.0
DetectLLM	99.0	100.0	99.9	99.8	99.9	96.7	99.7	99.8	99.7	99.8	100.0	99.9
Lastde++	99.0	100.0	99.4	99.8	99.6	85.1	99.5	99.6	98.0	99.9	100.0	99.9

Table 4. Full comparison of black-box machine text detection performances (AUROC) of MIA methods and machine text detectors on the RAID benchmark. Target models are ChatGPT and GPT-4. Surrogate models are Llama: LLaMA-3-3.2B, Pythia: PYTHIA-160M. Gray, blue, and red areas indicate general baseline methods, MIA methods, and machine text detectors, respectively.

Method	Abstracts				Books				News				Poetry			
	ChatGPT		GPT-4		ChatGPT		GPT-4		ChatGPT		GPT-4		ChatGPT		GPT-4	
	Llama	Pythia	Llama	Pythia	Llama	Pythia	Llama	Pythia	Llama	Pythia	Llama	Pythia	Llama	Pythia	Llama	Pythia
Loss	96.0	95.6	38.5	51.9	99.8	98.2	79.1	72.0	99.5	99.1	66.1	84.0	88.9	79.4	46.0	51.0
Rank	73.6	82.9	49.1	57.3	99.5	91.8	96.0	59.3	93.2	92.3	62.6	69.2	85.6	73.9	48.0	36.5
LogRank	96.6	96.2	40.3	53.8	99.8	97.5	77.6	65.1	99.7	98.8	68.1	80.4	88.9	76.1	44.1	43.0
Entropy	74.5	68.3	19.8	39.2	99.4	86.2	83.7	62.3	99.1	87.4	68.0	69.3	87.4	62.3	44.7	37.3
Reference	37.6	50.3	29.0	38.9	58.4	89.2	50.6	70.4	50.0	54.8	33.8	42.1	79.5	77.2	71.4	50.1
Zlib	53.3	39.7	12.4	14.8	72.5	59.8	64.1	62.2	89.6	76.9	70.5	72.7	68.2	56.0	60.2	63.4
Neighborhood	61.0	77.1	29.4	45.9	78.9	87.4	57.4	58.8	81.6	88.3	53.9	61.4	74.5	78.0	66.7	61.8
Min-K%	98.3	97.5	56.1	62.5	99.8	96.4	74.4	58.6	99.6	98.5	63.5	76.3	89.8	77.0	42.3	37.2
Min-K%++	99.1	99.3	71.9	68.7	90.3	98.9	36.7	70.4	73.6	99.7	38.0	86.3	65.5	92.4	44.4	69.0
ReCaLL	99.3	98.0	81.1	97.4	96.3	87.6	65.6	83.2	87.4	66.5	47.0	55.7	89.0	73.8	52.3	54.5
DC-PDD	88.7	85.9	38.5	62.5	99.0	96.1	77.2	86.0	93.6	85.1	47.6	72.0	87.2	80.9	51.0	64.9
DetectGPT	61.0	77.1	29.4	45.9	78.9	87.4	57.4	58.8	81.6	88.3	53.9	61.4	74.5	78.0	66.7	61.8
Fast-DetectGPT	99.3	98.5	77.0	58.7	83.9	96.1	38.0	69.9	72.2	99.4	46.5	88.0	70.9	89.7	53.2	79.0
Binoculars	100.0	98.1	96.5	94.5	99.4	97.1	77.0	70.5	98.8	78.6	83.0	51.2	98.9	85.9	57.4	73.0
DetectLLM	98.9	98.0	77.1	57.7	85.6	94.9	42.7	66.7	77.8	99.1	51.0	85.2	75.7	88.6	56.0	74.8
Lastde++	98.6	96.9	75.8	58.3	83.6	91.8	40.2	60.6	69.1	98.6	45.2	84.2	71.9	84.7	53.0	67.0

Method	Recipes				Reddit				Reviews				Wikipedia			
	ChatGPT		GPT-4		ChatGPT		GPT-4		ChatGPT		GPT-4		ChatGPT		GPT-4	
	Llama	Pythia	Llama	Pythia	Llama	Pythia	Llama	Pythia	Llama	Pythia	Llama	Pythia	Llama	Pythia	Llama	Pythia
Loss	99.7	99.0	93.4	94.8	98.8	96.6	91.6	87.0	99.9	99.5	89.0	80.9	99.0	97.1	73.8	82.4
Rank	56.9	89.7	48.0	90.5	87.7	86.2	79.0	75.7	99.1	94.7	75.7	63.1	94.8	93.3	77.2	75.4
LogRank	99.5	98.9	92.9	94.6	98.5	95.6	90.4	84.3	99.9	99.2	86.2	72.2	99.2	97.4	76.0	82.3
Entropy	99.7	92.6	93.5	83.1	96.5	88.5	90.8	82.1	99.9	93.4	91.2	66.0	98.7	83.4	73.6	72.9
Reference	55.6	36.5	36.5	29.4	60.5	83.3	52.4	74.6	76.8	94.8	58.8	82.4	52.4	50.0	41.3	42.9
Zlib	96.9	93.4	90.9	90.8	78.0	64.9	94.0	92.8	91.2	75.8	68.5	65.3	99.9	99.9	98.0	98.6
Neighborhood	66.3	87.5	58.6	78.8	76.4	86.6	74.6	72.1	94.3	87.6	72.7	55.3	92.9	91.7	71.7	68.0
Min-K%	99.5	98.7	92.8	93.9	98.3	93.9	87.5	79.6	99.9	98.1	79.5	65.4	99.5	97.9	78.9	81.2
Min-K%++	95.1	99.7	76.6	94.5	89.0	96.8	64.8	78.6	91.5	99.2	46.0	81.6	84.5	99.3	52.3	80.8
ReCaLL	95.9	88.0	69.2	74.2	97.0	77.3	58.2	43.2	87.9	65.0	58.4	67.7	80.3	28.0	34.6	22.9
DC-PDD	98.2	94.4	88.1	84.7	98.6	96.0	87.3	86.2	99.8	97.0	86.7	90.8	88.3	71.9	45.2	55.2
DetectGPT	66.3	87.5	58.6	78.8	76.4	86.6	74.6	72.1	94.3	87.6	72.7	55.3	92.9	91.7	71.7	68.0
Fast-DetectGPT	84.1	99.8	67.2	98.8	85.8	94.1	66.3	85.0	81.2	99.8	40.4	86.6	79.8	99.9	54.9	93.0
Binoculars	99.9	98.2	97.8	94.7	99.6	91.3	94.6	76.1	99.5	96.5	84.0	83.8	98.8	97.7	85.1	77.8
DetectLLM	82.2	99.8	66.6	99.1	85.9	92.6	67.4	83.6	84.6	99.7	42.2	84.8	82.1	99.9	56.8	92.6
Lastde++	83.9	99.4	67.8	97.9	83.9	90.4	66.6	83.9	84.2	99.3	40.9	80.4	81.6	99.8	57.4	92.0