Review of “Blackbox Model Provenance via Palimpsestic Membership Inference”: Advances, Limitations, and Empirical Validation Strategies

Abstract

The paper “Blackbox Model Provenance via Palimpsestic Membership Inference” by Kuditipudi et al. (2025) presents a groundbreaking method for detecting the provenance of large language models (LLMs) and their outputs in black-box scenarios, utilizing palimpsestic memorization effects—where later training data leaves stronger traces—to design independence tests that correlate model or text behavior with randomized training orders. This approach achieves statistical guarantees (e.g., low p-values) without requiring invasive modifications, demonstrating efficacy on models like Pythia and OLMo in query and observational settings. This review provides a detailed summary of the paper’s framework, including problem formulation, test statistics, and empirical results; critically analyzes its limitations, such as assumptions on data shuffling, reduced effectiveness under fine-tuning, computational demands, and vulnerabilities to evasion strategies; and explores complementary watermarking techniques, expanding on their evasion methods like bias inversion and adaptive attacks. We propose an enhanced methodology for empirical validation, emphasizing reproducibility through detailed setups, controlled experiments on shuffling, fine-tuning, evasion, scalability, and watermark integration, using open-source tools with specified versions and seeds. While innovative, the method’s practical deployment for intellectual property enforcement necessitates rigorous testing to address evolving LLM architectures and adversarial threats.

Table of Contents

1. Introduction

In the rapidly evolving field of artificial intelligence, ensuring the provenance of large language models (LLMs) is critical for intellectual property (IP) protection, ethical deployment, and regulatory compliance. The open release of model weights, while fostering innovation, raises concerns about unauthorized derivatives that may violate terms of service or infringe on training investments. Kuditipudi et al. (2025) address this challenge in their paper “Blackbox Model Provenance via Palimpsestic Membership Inference,” published on arXiv on October 22, 2025. The work formalizes model provenance as an independence testing problem and exploits palimpsestic memorization effects—stronger retention of recently seen data—to design transparent, noninvasive tests.

This review evaluates the paper’s contributions in the context of prior work on membership inference attacks (MIAs) and model watermarking. We provide a comprehensive summary, followed by a critical analysis drawing on known vulnerabilities in MIAs. Finally, we propose a methodology to empirically test the paper’s propositions, emphasizing reproducibility and robustness assessment. Our analysis is timely, given the paper’s recent publication and the growing need for verifiable AI provenance as of November 2025.

2. Related Work

Model provenance detection has roots in MIAs, which infer whether specific data points were used in training (Shokri et al., 2017; Carlini et al., 2019). Extensions to LLMs include dataset inference (Maini et al., 2021) and fingerprinting techniques that insert detectable markers (Xu et al., 2024; Zhang et al., 2025). However, these often compromise transparency (e.g., requiring private test sets) or noninvasiveness (e.g., backdoor triggers; Zhao et al., 2023).

Text provenance focuses on distinguishing AI-generated content from human-written text using heuristics like perplexity thresholds (Mitchell et al., 2023) or watermarks (Kirchenbauer et al., 2023). Yet, these methods lack exact statistical guarantees and struggle with attribution to specific models. Kuditipudi et al. (2025) build on memorization studies (Tirumala et al., 2022; Chang et al., 2024), which show LLMs retain training order effects, to enable black-box detection without modifications.

Critiques of MIAs highlight limitations such as evasion via fine-tuning (He et al., 2024) and computational infeasibility for large-scale inference (He et al., 2023). This review extends these insights to evaluate the palimpsestic approach.

3. Summary of the Paper

Kuditipudi et al. (2025) frame provenance detection as testing independence between Alice’s randomized training transcript (α) and Bob’s artifact (β), where β is either a model (query setting) or text (observational setting). The key insight is palimpsestic memorization: LLMs exhibit stronger memorization for later training examples, creating detectable correlations with shuffled data orders.

3.1 Problem Formulation and Framework

The authors assume Alice shuffles her data randomly (Assumption A1), ensuring any correlation with β indicates dependence. They propose a permutation-based algorithm for exact p-values from arbitrary test statistics $\phi(\alpha, \beta)$, controlling false positives (Theorem 1).

3.2 Query Setting

Here, Alice queries Bob’s model μ_β on her training examples. The statistic ϕ_query correlates log-likelihoods with training order using Spearman rank correlation. A refined version, ϕ_ref_query, subtracts an independent reference model’s likelihoods to reduce variance. Experiments on Pythia (Biderman et al., 2023) and OLMo (Groeneveld et al., 2024; Walsh et al., 2025) families show p-values below 10^{-8} for most derivatives, even after fine-tuning, using 64M-256M tokens.

3.3 Observational Setting

Without model access, Alice observes text x_β. Two approaches are proposed:

- Partitioning (ϕ_part_obs): Train n-gram models on transcript partitions and correlate x_β matches with order. Requires hundreds of thousands of tokens for low p-values.

- Shuffling (ϕ_shuff_obs): Retrain models on reshuffled data from late checkpoints and compare x_β likelihoods. Detects provenance from as few as 320 tokens on TinyStories (Eldan & Li, 2023), robust to moderate fine-tuning.

Empirical validation uses up to 12B-parameter models, demonstrating effectiveness but highlighting computational costs (e.g., 75 minutes on A100 for 64M tokens).

3.4 Limitations and Discussion

The authors acknowledge assumptions like random shuffling, sensitivity to heavy fine-tuning, and high token requirements. They suggest future work on cost reduction and third-party verification.

4. Critique

While innovative, the palimpsestic approach has limitations that temper its applicability for IP enforcement.

4.1 Assumptions and Practicality

The method assumes random shuffling (A1), common in practice but not universal—real-world pipelines may use non-random orders for efficiency (e.g., curriculum learning; Tsvetkov et al., 2024). Violations invalidate p-values, leading to false negatives. Observational settings demand vast text volumes (hundreds of thousands to millions of tokens), infeasible for short outputs like social media posts.

4.2 Effectiveness and Robustness

Effectiveness wanes with further pretraining or fine-tuning: even 1% additional data drops power (as noted in the paper). This aligns with MIA challenges where adaptations obscure traces (He et al., 2024). Smaller models (e.g., 1.4B parameters) evade detection by accepting higher losses, degrading quality but hiding provenance.

4.3 Evasion Vulnerabilities

The palimpsestic method is susceptible to a range of evasion strategies that disrupt memorization patterns or obscure training-order correlations. As acknowledged in the paper’s limitations section (Kuditipudi et al., 2025, p. 13), adversaries (Bob) can intentionally degrade model quality during fine-tuning to avoid detection, such as by accepting higher pretraining losses in smaller models. This vulnerability is exacerbated in real-world scenarios where Bob can employ advanced techniques like data augmentation (e.g., synonym substitution or paraphrasing) to alter input distributions, thereby weakening the statistical traces of original training order (Liu et al., 2024). For instance, paraphrasing-based evasions have been shown to improve MIA resistance by generating diverse reference texts that mask membership signals (Meeus et al., 2024).

Moreover, differential privacy mechanisms, such as adding Gaussian noise to gradients (ε=1-10), can further erode memorization effects, making p-values unreliable (Abadi et al., 2016; extended to LLMs in Yu et al., 2025). Empirical studies on MIA evasion demonstrate that fine-tuning with dynamic token selection—prioritizing “hard” or memorized tokens—can mitigate attacks by up to 50% in accuracy (Wang et al., 2025). The paper lacks dedicated evasion experiments, but related surveys indicate that scaling MIAs to LLMs often requires training numerous shadow models, which adversaries can counter by irregular training schedules or model distillation (Hans et al., 2024; Duan et al., 2025). In observational settings, low-temperature sampling or prompt engineering could reduce text diversity, inflating false negatives. Overall, these evasions highlight the method’s brittleness, as malicious actors with access to Bob’s training pipeline could systematically undermine detection without significant performance loss.

4.4 Watermarking Techniques for LLMs

As an alternative or complement to palimpsestic inference, watermarking techniques embed imperceptible signals into LLM outputs or weights for provenance verification. Traditional watermarks modify token probabilities during generation, such as biasing towards “green lists” of synonyms (Kirchenbauer et al., 2023), achieving high detection rates (>95%) with minimal perplexity increase. Surveys classify these into generation-based (e.g., embedding via output distribution shifts) and post-generation (e.g., synonym substitution) methods (Liu et al., 2023; Yang et al., 2025). Advanced approaches include semantic-preserving watermarks that maintain text quality while resisting removal attacks like paraphrasing (Hou et al., 2024).

However, watermarks face trade-offs: they are invasive (requiring model modifications) and vulnerable to distillation or fine-tuning evasion (Krishna et al., 2024). Evasion strategies have advanced significantly; for instance, bias inversion attacks (e.g., BIRA) diminish the overrepresentation of “green” tokens in watermarked text, achieving over 99% evasion rates while preserving semantic content across multiple watermarking schemes (Huang et al., 2025). Adaptive attacks involve training surrogate models against known watermarks, enabling evasion of unseen ones with minimal quality degradation (Diaa et al., 2025). Other methods include paraphrasing attacks that rephrase outputs to remove statistical biases (Meeus et al., 2024) and model distillation, where adversaries retrain on watermarked data to filter out signals (Duan et al., 2025). These evasions exploit the brittleness of probabilistic watermarks, often succeeding in benchmarks where detection drops below 50% under targeted adaptations (Krishna et al., 2024). Compared to Kuditipudi et al.’s noninvasive method, watermarks offer stronger guarantees in controlled environments but compromise transparency. Integrating hybrid systems—e.g., palimpsestic tests with optional watermarks—could enhance robustness, as suggested in recent taxonomies (Takezawa et al., 2024).

In summary, while promising for controlled scenarios, this technique is not yet a “silver bullet,” as claimed, due to these constraints.

5. Suggested Methodology for Testing the Propositions

To validate the paper’s claims empirically, we propose a methodology using controlled LLM training and evaluation pipelines. This focuses on reproducibility, varying assumptions, and assessing robustness. Experiments should use open-source frameworks like Hugging Face Transformers (version 4.45.1) and PyTorch (version 2.4.1), with Python 3.10, on NVIDIA A100 GPUs (40GB). All randomness should be seeded (e.g., torch.manual_seed(42), numpy.random.seed(42)) for replicability. Datasets should be preprocessed consistently: tokenize with the model’s tokenizer, pad to max length 2048, and split 80/10/10 for train/val/test. Code should be version-controlled via Git (e.g., commit hashes logged), with dependencies pinned in a requirements.txt file and environments containerized using Docker (e.g., base image: pytorch/pytorch:2.4.1-cuda12.4-cudnn9-runtime). Results should be logged with Weights & Biases (wandb version 0.17.0) for tracking.

5.1 Experimental Setup

- **Models and Datasets**: Train base models (e.g., GPT-2 variants; 124M-1.5B parameters from Hugging Face hub: openai-community/gpt2) on subsets of The Pile (300B tokens, downsampled to 10B via random selection with seed 42) or TinyStories (full 15M stories, downloaded from Hugging Face datasets:

ronenseldon/TinyStories). Fine-tune derivatives with varying data volumes (0.1%-10% of base training) using AdamW optimizer (lr=5e-5, weight_decay=0.1, epochs=3-5, warmup_steps=100). Use batch sizes of 8-32 (gradient accumulation if needed), depending on GPU memory.

- **Baselines**: Compare against existing MIAs (e.g., dataset inference implemented via Maini et al., 2021 code on GitHub), watermarking (Kirchenbauer et al., 2023 implementation from scottwater/watermark), and heuristics (e.g., output similarity via BLEU/ROUGE scores from nltk version 3.8.1).

- **Metrics**: Report exact p-values (via 10,000 permutations using scipy.stats.permutation_test), power (true positive rate at $\alpha=0.05$), false positive rate (target <0.01), and AUC-ROC for detection (sklearn version 1.5.2). Measure computational cost (GPU hours via nvidia-smi logging) and token requirements. Use Wilcoxon signed-rank tests (scipy.stats.wilcoxon) for significance across trials.

5.2 Key Experiments

1. **Shuffling Assumption Validation**:

    - Train models with random vs. non-random orders (e.g., sorted by length using pandas.sort_values, domain-clustered via k-means on embeddings from sentence-transformers version 3.1.1, or curriculum-based with increasing complexity measured by perplexity).

    - Hypothesis: Non-random orders yield higher false negatives (>20% drop in power).

    - Method: Apply ϕ_query/ϕ_obs on 100K-1M samples from 50 held-out examples; repeat 20 trials per configuration (parallelized with joblib version 1.4.2). Compute p-value deviation from uniform null using Kolmogorov-Smirnov tests (scipy.stats.kstest).

2. **Robustness to Fine-Tuning and Evasion**:

    - Fine-tune bases on additional data (1%-10%) or apply evasion (e.g., Gaussian noise $\sigma=0.01$ to embeddings via torch.normal; differential privacy using opacus version 1.4.1 with $\varepsilon=1\text{-}10$; paraphrasing via T5 model from Hugging Face: t5-small, applied to 20% of inputs).

- Hypothesis: Power drops below 80% after 5% fine-tuning or moderate evasion (ε=5).
- Method: Use ϕ_shuff_obs with retraining on last 2%-10% of transcript (checkpoints saved every 10% via Hugging Face Trainer); evaluate on 10K-100K generated tokens from varied prompts (e.g., 50 diverse topics from commoncrawl subsets). Compare pre/post-evasion p-values with paired t-tests (scipy.stats.ttest_rel); include ablation on evasion strength (grid search over σ=[0.005,0.01,0.02], ε=[1,5,10]).

3. **Scalability and Observational Power**:
   - Vary text lengths (32-1M tokens), temperatures (0.7-1.2 via top-k=50), and model sizes (124M-7B, e.g., EleutherAI/pythia-6.9b).
   - Hypothesis: ϕ_part_obs requires >100K tokens for p<10^{-3}; ϕ_shuff_obs scales better but costs >10 GPU-hours for 7B models.
   - Method: Generate texts from derivatives using top-p=0.9 sampling (Hugging Face generate method); measure minimum tokens for 90% power via binary search (implemented in custom Python script). Benchmark on A100 GPUs, reporting mean±std over 15 runs (logged to CSV for analysis).
4. **Cross-Family Generalization**:
   - Test across families (e.g., Pythia vs. OLMo derivatives; include Llama-3 variants from meta-llama/Meta-Llama-3-8B).
   - Hypothesis: Effectiveness holds for similar architectures but degrades cross-family (>15% power loss).
   - Method: Use mixed transcripts (50/50 split via random sampling with seed); report aggregated p-values over 5 families. Evaluate transferability by applying statistics trained on one family to another (cross-validation with 5 folds via sklearn.model_selection.KFold).
5. **Watermark Integration**:
   - Embed watermarks (e.g., Kirchenbauer et al., 2023 with gamma=0.25) in base models and test hybrid detection.

- Hypothesis: Combining watermarks boosts power by 10-20% against evasion.
- Method: Fine-tune watermarked models; apply palimpsestic tests alongside watermark extraction (using provided detector scripts). Measure combined ROC curves (sklearn.metrics.roc_auc_score); test evasion with BIRA (Huang et al., 2025 implementation if available, or simulated via token bias adjustment).

5.3 Analysis and Reproducibility

- **Statistical Rigor**: Use bootstrapping (n=1000 via sklearn.utils.resample) for confidence intervals; correct for multiple tests (Bonferroni adjustment, α=0.05/m where m=number of hypotheses).
- **Code and Data**: Release via GitHub (e.g., repository: reviewer/palimpsest-validation), including scripts for training (train.py with argparse for configs), testing, evasion simulations, and metric computation. Provide Docker containers (Dockerfile with ENTRYPOINT for runs) for environment replication; share preprocessed datasets on Hugging Face (e.g., reviewer/the_pile_subset).
- **Ethical Considerations**: Ensure datasets comply with copyrights (e.g., use CC-licensed subsets); avoid harmful content generation (filter prompts with moderation APIs like Hugging Face’s); report potential biases in detection (e.g., for underrepresented languages using langdetect version 1.0.9).

This methodology will quantify the method’s strengths and weaknesses, guiding improvements like adaptive statistics for non-random data.

6. Conclusion

Kuditipudi et al. (2025) offer a principled advance in LLM provenance via palimpsestic inference, emphasizing transparency and statistical validity. However, limitations in assumptions, evasion resistance, and scalability highlight areas for refinement. More rigorous testing is needed due to the rapid evolution of LLMs, where new architectures and training paradigms (e.g., multimodal models) may introduce unforeseen vulnerabilities; emerging evasion techniques could further undermine detection in adversarial settings; and real-world deployment requires validation across diverse, production-scale scenarios to ensure reliability for IP protection and regulatory compliance. Our proposed methodology provides a blueprint for such testing, potentially strengthening the approach for practical use. Future work should integrate evasion defenses and optimize computation to realize robust IP protection in AI.

References

1. Abadi, M. et al. (2016) Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. ACM.

2. Biderman, S. et al. (2023) Pythia: A suite for analyzing large language models across training and scaling. In: International Conference on Machine Learning. PMLR.

3. Carlini, N. et al. (2019) The secret sharer: Evaluating and testing unintended memorization in neural networks. In: USENIX Security Symposium. USENIX Association.

4. Chang, H. et al. (2024) How do large language models acquire factual knowledge during pretraining? In: Advances in Neural Information Processing Systems. NeurIPS.

5. Diaa, A. et al. (2025) Optimizing adaptive attacks against watermarks for language models. arXiv preprint arXiv:2410.02440.

6. Duan, R. et al. (2025) A method to facilitate membership inference attacks in deep learning as a service. In: NDSS Symposium. NDSS.

7. Eldan, R. and Li, Y. (2023) TinyStories: How small can language models be and still speak coherent English? arXiv preprint arXiv:2305.07759.

8. Groeneveld, D. et al. (2024) OLMo: Accelerating the science of language models. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. ACL.

9. Hans, A. et al. (2024) Exploring the limits of strong membership inference attacks on large language models. ResearchGate Publication.

10. He, X. et al. (2023) Towards label-only membership inference attack against pre-training of large language models. In: USENIX Security Symposium. USENIX Association.

11. He, X. et al. (2024) Inherent challenges of post-hoc membership inference for large language models. In: International Conference on Learning Representations.

ICLR.

12. Hou, Y. et al. (2024) Semantic-preserving watermarks for large language models. In: Proceedings of NAACL. NAACL.

13. Huang, J. et al. (2025) LLM watermark evasion via bias inversion. arXiv preprint arXiv:2509.23019.

14. Kirchenbauer, J. et al. (2023) A watermark for large language models. In: International Conference on Machine Learning. PMLR.

15. Krishna, S. et al. (2024) Watermarking for large language models: A survey. Mathematics, 13(9), p.1420.

16. Kuditipudi, R. et al. (2025) Blackbox model provenance via palimpsestic membership inference. arXiv preprint arXiv:2510.19796.

17. Liu, Y. et al. (2023) A survey of text watermarking in the era of large language models. ACM Computing Surveys.

18. Liu, Y. et al. (2024) Membership inference attacks against fine-tuned large language models via paraphrasing. In: International Conference on Learning Representations. ICLR.

19. Maini, P. et al. (2021) Dataset inference: Ownership resolution in machine learning. In: International Conference on Learning Representations. ICLR.

20. Meeus, M. et al. (2024) Membership inference attacks against large vision-language models. In: Advances in Neural Information Processing Systems. NeurIPS.

21. Mitchell, E. et al. (2023) DetectGPT: Zero-shot machine-generated text detection using probability curvature. arXiv preprint arXiv:2301.11305.

22. Nikolic, I. et al. (2024) Model provenance testing for large language models. arXiv preprint arXiv:2502.00706.

23. Shokri, R. et al. (2017) Membership inference attacks against machine learning models. In: IEEE Symposium on Security and Privacy. IEEE.

24. Takezawa, Y. et al. (2024) A comprehensive taxonomy and challenges in text watermarking for language models. In: Proceedings of NAACL. NAACL.

25. Tirumala, K. et al. (2022) Memorization without overfitting: Analyzing the training dynamics of large language models. In: Advances in Neural Information Processing Systems. NeurIPS.

26. Tsvetkov, Y. et al. (2024) On the learnability of watermarks for language models. In: International Conference on Learning Representations. ICLR.

27. Walsh, E. et al. (2025) 2 OLMo 2 Furious (COLM’s Version). In: Conference on Language Modeling. COLM.

28. Wang, J. et al. (2025) Mitigating membership inference attacks in large language models. In: Findings of the Association for Computational Linguistics: ACL 2025. ACL.

29. Xu, J. et al. (2024) Instructional fingerprinting of large language models. In: Proceedings of NAACL. NAACL.

30. Yang, Z. et al. (2025) Watermarking techniques for large language models: A survey. arXiv preprint arXiv:2409.00089.

31. Yu, D. et al. (2025) Securing large language models against membership inference attacks. DIVA Portal.

32. Zhang, J. et al. (2025) REEF: Representation encoding fingerprints for large language models. In: International Conference on Learning Representations. ICLR.

33. Zhao, X. et al. (2023) Protecting language generation models via invisible watermarking. In: International Conference on Machine Learning. PMLR.