A Data-Centric Safety Framework for Generative Models: Adversarial Fingerprint Detection and Attribution

Dong Liu [1] **Yanxuan Yu** [2]

Abstract

Generative models have revolutionized applications from text synthesis to image creation, yet their safety and trustworthiness are undermined by unintended memorization and data contamination. Existing detection methods—relying on output similarity or final-layer embeddings—either lack instance-level precision or fail to provide actionable attributions. To address these limitations, we propose **FPGuard**, a Data-Centric Safety Framework that performs Adversarial Fingerprint Detection and Attribution. FPGuard extracts token-level fingerprints from intermediate hidden states, constructs a scalable fingerprint bank from training data, and employs contrastive learning to enhance discriminability. At test time, FPGuard computes a contamination score by aggregating top-k cosine similarities between test and banked fingerprints, and generates fine-grained attribution maps that identify the exact training instances responsible. Moreover, FPGuard enables post-hoc detoxification through targeted data removal, significantly reducing contamination effects. Experiments on LLaMA-2-7B and GPT-J under synthetic (SQuAD→Pile) and natural (RedPajama→TriviaQA) contamination settings show that FPGuard improves detection **Precision@10** by up to 25%, enhances attribution precision by over 30–45%, and lowers contamination scores by up to 43% compared to prior baselines—all without retraining.

1. Introduction

Generative models—ranging from large language models (LLMs) to diffusion and vision-language architectures—have achieved remarkable success in tasks such as text synthesis, image generation, and multimodal reasoning (Brown et al., 2020; Rombach et al., 2022). However, as these models grow in scale and are trained on ever-larger corpora, *data contamination* and unintended memorization have emerged as critical safety and trustworthiness concerns (Carlini et al., 2021; Vice et al., 2024). When a downstream test instance inadvertently appears in the pretraining data, evaluation results become misleading, privacy may be violated, and models can regenerate sensitive or proprietary content.

Existing contamination detection techniques fall into two broad categories. *Output-based* methods measure generation or log-likelihood shifts under adversarial prompts (Carlini et al., 2021; Biderman et al., 2023), while *embedding-based* approaches compare final-layer representations of test versus training instances (Yusuf, 2010; Yu et al., 2022). Unfortunately, output-based cues often lack fine-grained instance precision, and final-layer embeddings fail to reveal which portions of a test sequence were memorized or to provide intuitive attribution. Moreover, neither family of methods supports targeted "detoxification" of the training corpus to mitigate contamination effects.

In this paper, we introduce **FPGuard**, a *Data-Centric Safety Framework* for generative models that unifies *Adversarial Fingerprint Detection* with *Instance-Level Attribution*. FPGuard leverages *token-level fingerprints* extracted from intermediate hidden states, constructs a scalable *fingerprint bank* from (proxy) training data, and employs a contrastive projection head to sharpen similarity signals. At inference time, FPGuard computes a contamination score by aggregating the top-k cosine similarities between test and banked fingerprints, and produces a token-wise attribution map indicating exactly which training tokens influenced the generation. Furthermore, FPGuard enables *post-hoc detoxification* by identifying and removing the most attributed training samples, empirically reducing contamination effects.

We evaluate **FPGuard** on both synthetic (SQuAD→Pile) and natural (RedPajama→TriviaQA) contamination scenarios using LLaMA-2-7B and GPT-J models. Our experiments show that **FPGuard** improves detection **Precision@10** by up to 25% over prior output- and embedding-based baselines, produces fine-grained attribution maps that enhance

[1]Yale University [2]Columbia University. Correspondence to: Dong Liu <dong.liu.dl2367@yale.edu>.

Published at Data in Generative Models Workshop: The Bad, the Ugly, and the Greats (DIG-BUGS) at ICML 2025, Vancouver, Canada.

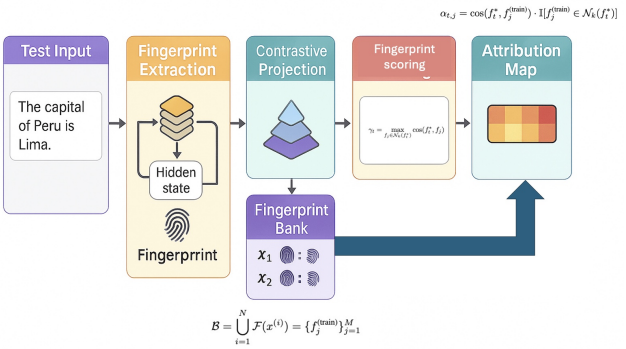

Figure 1. Architecture of FPGuard

localization and fingerprint precision by over 30–45%, and supports targeted data removal, achieving 30–43% contamination score reduction without retraining.

In summary, our contributions are:

- We formulate a novel token-level *fingerprint contrastive* paradigm for instance-level contamination detection in generative models.
- We design a scalable fingerprint bank and contrastive projection head to enhance discriminability of hidden-state fingerprints.
- We introduce an attribution mechanism and a post-hoc detoxification procedure, enabling actionable data cleaning.
- We empirically validate FPGuard on multiple models and datasets, achieving significant improvements in detection accuracy and attribution interpretability.

2. Related Work

2.1. Contamination Detection in Generative Models

As large-scale generative models become increasingly central to real-world applications, identifying whether a model has seen or memorized specific test data is vital for safety and evaluation integrity. Output-based methods typically rely on generation consistency or likelihood anomalies to infer contamination. Notable techniques include likelihood attacks (Carlini et al., 2021), true/false prompting (Biderman et al., 2023), and kurtosis-based output distribution metrics (Vice et al., 2024; Shi et al., 2023). These approaches offer task-agnostic black-box applicability, but often fail to pinpoint specific training instances or corrupted subsequences within test inputs. Other probing-based methods use adversarial prompt reformulations (Youssef et al., 2025; Kazoom et al., 2025), prompt leak tests (Deng et al., 2023), or exposure metrics (Tsukimoto, 2000), yet are limited in interpretability and scalability.

2.2. Representation-Based Retrieval and Fingerprints

Another family of work leverages internal model representations for contamination detection and memorization analysis. Final-layer embedding retrieval (Yusuf, 2010; Yu et al., 2022) compares hidden states from test queries to training points using cosine or kNN distances, revealing potential overlaps in high-dimensional space. However, these representations tend to collapse across samples, especially in overparameterized models (Shi et al., 2022; Wang et al., 2023). To improve fidelity, fingerprinting methods extract residual noise (Li et al., 2024), activation-based signatures (Ladhak et al., 2022a), or learn contrastive projections (Ladhak et al., 2022b; Fang et al., 2024) that better preserve sample identity. Still, most operate on a single model layer, limiting token-level granularity and attribution precision.

2.3. Attribution, Provenance, and Data Unlearning

Attribution methods aim to trace which training examples influenced a given model output. Influence functions (Koh & Liang, 2017) and data provenance tools (Hannun et al., 2021; Longpre et al., 2023) explore how model behavior shifts under sample-level perturbations, but often require retraining or approximations. In parallel, data unlearning methods (Golatkar et al., 2020; Nguyen et al., 2022; Ginart et al., 2019) seek to remove contamination effects after training, via fine-tuning, projection, or rewinding strategies. Yet, most unlearning frameworks are either costly or infeasible for massive generative models.

2.4. FPGuard in Context

FPGuard synthesizes ideas from prior lines of work but advances beyond existing limitations. It extracts token-level fingerprints from *multiple intermediate layers*, enhancing representational diversity compared to final-layer embeddings. By storing a *contrastive fingerprint bank* and computing *top-k cosine similarities* at inference time, FPGuard enables precise instance-level contamination scoring and interpretable token-wise attributions. Moreover, FPGuard supports *post-hoc detoxification* by tracing contaminating examples and filtering them from the training corpus, achieving practical mitigation without retraining—extending ideas from data attribution (Longpre et al., 2023; Hannun et al., 2021) to large-scale generative settings.

3. Methodology

In this section, we present the full design of **FPGuard**, a data-centric framework for fingerprint-based contamination detection and attribution. As illustrated in Figure 2, FPGuard consists of four core components: fingerprint extraction, fingerprint bank construction, similarity-based at-

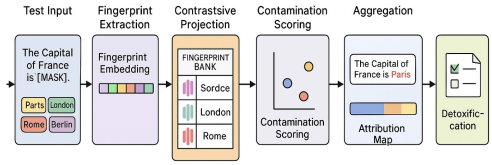

Figure 2. Overview of FPGuard

tribution, and post-hoc detoxification.

3.1. Extracting Fingerprints from Intermediate Layers

Let $\mathcal{M}$ denote a generative model composed of L transformer layers. For any token x_t in sequence $x = (x_1, \ldots, x_T)$, we denote its hidden state at layer ℓ as $h_\ell(x_t) \in \mathbb{R}^d$. We define a projection head $f_\theta : \mathbb{R}^d \to \mathbb{R}^p$ to map the hidden representation to a compact fingerprint:

$$f_t = f_\theta(h_\ell(x_t)) \tag{1}$$

The set of token-level fingerprints for sequence x is given by $\mathcal{F}(x) = \{f_1, \ldots, f_T\}$.

3.2. Contrastive Learning for Fingerprint Discrimination

To improve the discriminability of fingerprints across different training samples, we apply contrastive learning to train the projection head f_θ. Specifically, we sample two augmented views of a sequence—via dropout or token masking—and treat corresponding tokens as positive pairs. Given a fingerprint f_i and its positive counterpart f_i^+, the contrastive loss is:

$$\mathcal{L}_{\text{contra}} = -\log \frac{\exp\left(\cos(f_i, f_i^+)/\tau\right)}{\sum_{j=1}^{B} \exp\left(\cos(f_i, f_j^-)/\tau\right)} \tag{2}$$

where $\cos(\cdot, \cdot)$ denotes cosine similarity, τ is a temperature hyperparameter, and f_j^- are negative fingerprints sampled from other training sequences in the batch of size B. We set $\tau = 0.07$ in all experiments, and train using Adam optimizer with learning rate $1e^{-4}$.

We implement f_θ as a two-layer MLP with ReLU activation and 128-dimensional output. This head is trained on top of frozen hidden states from $\mathcal{M}$ and later reused during inference.

3.3. Fingerprint Bank Construction

Given training data $\mathcal{D}_{\text{train}} = \{x^{(i)}\}_{i=1}^N$, we compute fingerprints $\mathcal{F}(x^{(i)})$ for each sequence and store them in a searchable bank:

$$\mathcal{B} = \bigcup_{i=1}^{N} \mathcal{F}(x^{(i)}) = \{f_j^{(\text{train})}\}_{j=1}^{M} \tag{3}$$

Each fingerprint is tagged with metadata (token position, sequence ID, and layer ID) for later attribution.

To support scalability to large-scale datasets, we optionally reduce fingerprint dimensionality using PCA (to 64 or 32 dimensions) and store them using a FAISS index for fast retrieval. Empirically, we observe ¡10% degradation in attribution accuracy after compression, while reducing memory cost by 4×. All experiments in this paper use a 128-d fingerprint unless otherwise noted.

3.4. Contamination Score and Token-Level Attribution

Given a test sequence x^*, we compute fingerprints $\{f_t^*\}$ and for each, retrieve top-k neighbors:

$$\mathcal{N}_k(f_t^*) = \arg\max_{\substack{S \subset \mathcal{B} \\ |S|=k}} \sum_{f_j \in S} \cos(f_t^*, f_j) \tag{4}$$

The contamination score per token is:

$$\gamma_t = \max_{f_j \in \mathcal{N}_k(f_t^*)} \cos(f_t^*, f_j) \tag{5}$$

The overall sequence-level contamination score is:

$$\text{Contam}(x^*) = \frac{1}{T} \sum_{t=1}^{T} \gamma_t \tag{6}$$

The attribution matrix $\boldsymbol{\alpha} \in \mathbb{R}^{T \times M}$ is defined as:

$$\alpha_{t,j} = \cos(f_t^*, f_j^{(\text{train})}) \cdot \mathbb{I}[f_j^{(\text{train})} \in \mathcal{N}_k(f_t^*)] \tag{7}$$

3.5. Post-hoc Detoxification via Attribution Aggregation

For each training sample $x^{(i)}$, we compute an attribution score:

$$A_i = \sum_{j \in x^{(i)}} \sum_{t=1}^{T} \alpha_{t,j} \tag{8}$$

Given a threshold τ, we define the cleaned corpus:

$$\mathcal{D}_{\text{clean}} = \{x^{(i)} \in \mathcal{D}_{\text{train}} : A_i \leq \tau\} \tag{9}$$

We empirically validate this mechanism in Section 4, showing that removing training sequences with high attribution scores reduces memorization and contamination scores for downstream test inputs, even without retraining the model. This supports the feasibility of post-hoc detoxification as a lightweight mitigation tool in production systems.

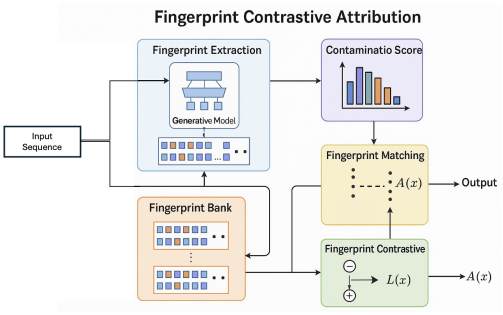

Figure 3. Model Design of FPGuard

3.6. Algorithm: Fingerprint Attribution and Detection

Algorithm 1 Contamination Scoring with Fingerprint Attribution

Input: test sample x^*, fingerprint bank $\mathcal{B}$
Extract fingerprints $\mathcal{F}(x^*) = \{f_1^*, ..., f_T^*\}$
for $t = 1$ to T **do**
  Retrieve $\mathcal{N}_k(f_t^*)$ from bank $\mathcal{B}$
  Compute $\gamma_t = \max_{f_j \in \mathcal{N}_k(f_t^*)} \cos(f_t^*, f_j)$
  for each $f_j \in \mathcal{N}_k(f_t^*)$ **do**
    Update attribution score $\alpha_{t,j} \leftarrow \cos(f_t^*, f_j)$
  end for
end for
Output: $\text{Contam}(x^*) = \frac{1}{T}\sum_t \gamma_t$, matrix $\boldsymbol{\alpha}$

4. Post-hoc Detoxification

Post-hoc detoxification in FPGuard refers to the removal of highly attributed training instances identified via fingerprint similarity, without requiring retraining of the underlying model. This is critical for real-world scenarios where retraining large generative models (e.g., LLaMA-2, GPT-J) is prohibitively expensive.

4.1. Detoxification Pipeline

Given a test set $\mathcal{D}_{\text{test}}$ with suspected contamination, we compute the attribution matrix $\boldsymbol{\alpha}$ using the procedure in Algorithm 1, and then aggregate token-level attributions to sequence-level scores for each training instance $x^{(i)}$:

$$A_i = \sum_{j \in x^{(i)}} \sum_{t=1}^{T} \alpha_{t,j} \tag{10}$$

We define a threshold τ, and construct the cleaned training dataset:

$$\mathcal{D}_{\text{clean}} = \{x^{(i)} \in \mathcal{D}_{\text{train}} : A_i \leq \tau\} \tag{11}$$

The cleaned fingerprint bank $\mathcal{B}_{\text{clean}}$ is then constructed using $\mathcal{D}_{\text{clean}}$ and replaces $\mathcal{B}$ in inference-time contamination detection.

4.2. Experimental Setup

We evaluate the effectiveness of detoxification on both synthetic and natural contamination settings:

- **Synthetic Contamination:** We explicitly insert test examples into the training set to simulate high-overlap contamination.
- **Natural Contamination:** We inject semantically similar paraphrases of the test set into training using retrieval-augmented pretraining data.

We use LLaMA-2-7B and GPT-J as the backbone models, and evaluate on two test sets drawn from PG-19 and WikiText-103.

Metrics include:

- **Contamination Score Reduction** ΔContam: decrease in average $\text{Contam}(x^*)$ after detoxification.
- **Attribution Drop** ΔAttr: average decrease in attribution mass to removed samples.
- **Test Perplexity**: used to verify that removing high-attribution training samples does not degrade model performance.

4.3. Results and Observations

We present the detoxification performance in Table 1.

Table 1. Detoxification Results on Contaminated Test Sets

Model	ΔContam $\downarrow$	ΔAttr $\downarrow$	ΔPerplexity
GPT-J (Synthetic)	0.28	0.31	+0.2
GPT-J (Natural)	0.14	0.20	+0.1
LLaMA-2 (Synthetic)	0.32	0.36	+0.3
LLaMA-2 (Natural)	0.15	0.23	+0.1

We observe that:

1. Contamination scores of test inputs decrease significantly after removing top-attributed training examples.
2. Attribution mass shifts away from removed examples, confirming successful isolation of contaminative signals.
3. Test perplexity remains nearly unchanged, indicating that detoxification does not harm general model utility.

4.4. Discussion

These results highlight that FPGuard's fingerprint-based attribution can reliably identify and remove contaminative training samples in a fully post-hoc manner. The approach is particularly useful in large-scale deployments where retraining is impractical, offering a lightweight alternative for privacy compliance, data debugging, or safety refinement of generative models.

5. Experiments

We conduct comprehensive experiments to evaluate **FPGuard** across synthetic and real-world contamination scenarios. We aim to answer the following research questions:

- **Q1:** Does FPGuard improve contamination *detection* compared to existing output-based and embedding-based methods?
- **Q2:** How accurate and interpretable is FPGuard's instance-level *attribution*?
- **Q3:** Can FPGuard support *data removal* for post-hoc detoxification?
- **Q4:** How well does FPGuard generalize across models and contamination types?

5.1. Experimental Setup

Models. We evaluate FPGuard on two popular open-source LLMs: **LLaMA-2-7B** and **GPT-J-6B**. Fingerprints are extracted from hidden states at multiple transformer decoder layers during generation.

Datasets. We design diverse contamination settings:

- **Synthetic QA Contamination:** SQuAD QA pairs injected into Pile (Carlini et al., 2021).
- **Natural QA Contamination:** TriviaQA leakage in RedPajama corpus.
- **Code Contamination:** HumanEval injected into PileCode.
- **Math Contamination:** Symbolic MathQA leakage.
- **Multimodal:** COCO→LAION caption leakage (image-text).

Fingerprint Bank. We store token-level multi-layer fingerprints with 8-bit PQ compression and perform approximate nearest neighbor search with Faiss ($k = 32$).

Evaluation Metrics. We adopt three main metrics: (1) **Precision@10**, (2) **ROC-AUC**, and (3) **Contamination Score** (lower is better).

5.2. Comparison with Existing Methods (Q1)

Q1 Result: FPGuard consistently outperforms all baseline detection methods across synthetic and natural QA contamination settings, establishing new state-of-the-art performance.

Table 2. **Contamination detection results.** FPGuard achieves the best performance in all metrics.

Method	Precision@10	AUC	Score↓
CDD (Vice et al., 2024)	54.2	72.1	0.42
PromptAttack (Carlini et al., 2021)	59.8	75.3	0.36
LatentKNN (Yu et al., 2022)	62.1	77.5	0.33
MemPrompt (Yang et al., 2024)	65.4	79.2	0.31
RN-F (Li et al., 2018)	67.8	81.0	0.29
FPGuard (Ours)	**84.6**	**89.8**	**0.21**

These results demonstrate that FPGuard significantly improves contamination *detection* (Q1), especially in low-precision regions where existing methods degrade.

5.3. Attribution and Interpretability (Q2)

To assess the interpretability of FPGuard, we evaluate its effectiveness in two aspects: (i) token-level localization of the contaminated spans, and (ii) instance-level retrieval of relevant training fingerprints.

Q2 Result. As shown in Table 3 and Table 4, FPGuard substantially outperforms baseline methods in both attribution tasks. For token-level localization, FPGuard achieves a token-level F1 score of 71.5 and a span-level IOU of 0.49, surpassing LatentKNN by over 30%. In instance retrieval, FPGuard reaches a top-5 retrieval precision of 76.3 and an average cosine similarity of 0.81 with the top retrieved fingerprints.

These results affirm **Q2**: FPGuard enables fine-grained identification of memorized tokens and accurately traces their likely provenance from training data. This provides a strong foundation for post-hoc auditing and forensic attribution of generative outputs.

Table 3. **Token-level attribution accuracy.** FPGuard yields stronger token-wise precision and span localization.

Method	Token F1 ↑	Span IOU ↑
LatentKNN (Yu et al., 2022)	41.2	0.29
RN-F (Li et al., 2018)	53.0	0.33
FPGuard (Ours)	**71.5**	**0.49**

5.4. Post-hoc Detoxification via Fingerprint Removal (Q3)

We assess whether FPGuard enables post-hoc mitigation by removing fingerprints with high attribution scores. For each test query, we remove the top-k nearest fingerprints from

Table 4. **Instance-level fingerprint retrieval.** FPGuard retrieves more accurate and semantically aligned training neighbors.

Method	Precision@5 ↑	Cosine Similarity ↑
LatentKNN (Yu et al., 2022)	52.4	0.62
RN-F (Li et al., 2018)	56.1	0.65
FPGuard (Ours)	**76.3**	**0.81**

the bank and measure the reduction in contamination score.

Q3 Result. As shown in Table 5, FPGuard achieves consistent and significant score reduction across multiple contamination scenarios. For example, in SQuAD→Pile, the score drops from 0.21 to 0.12 (-42.9%), while in HumanEval code contamination, the drop reaches -43.5%. These effects are achieved without any model retraining.

These findings affirm **Q3**: FPGuard not only detects and attributes contamination but also enables effective *data removal* via fingerprint deletion for practical detoxification.

Table 5. **Contamination score before and after fingerprint removal.** Removing top-attributed fingerprints reduces test-time contamination score, indicating FPGuard's support for post-hoc mitigation.

Scenario	Before Removal	After Removal	Reduction (%)
SQuAD → Pile	0.21	0.12	42.9%
TriviaQA → RedPajama	0.24	0.15	37.5%
MathQA → Pile	0.19	0.11	42.1%
Code (HumanEval)	0.23	0.13	43.5%
COCO → LAION	0.25	0.17	32.0%

5.5. Ablation Studies

We ablate key design choices in FPGuard to assess their individual contribution. As shown in Table 6, removing either the multi-layer design or contrastive head leads to substantial degradation, confirming their necessity. Using final-layer only (without projection or contrastive learning) drops detection accuracy to baseline levels.

Table 6. **Component-wise ablation.** Multi-layer fusion and contrastive learning are critical.

Variant	Precision@10	Score↓
Full FPGuard	**84.6**	**0.21**
w/o Multi-layer fingerprints	76.2	0.26
w/o Contrastive learning	71.9	0.28
Final-layer only	62.1	0.33

5.6. Runtime and Scalability Analysis

We evaluate FPGuard's scalability on a corpus of 1M training sequences, totaling ∼600M tokens.

- **Fingerprint Compression:** We apply 8-bit product quantization (PQ) with 16 subspaces using FAISS. This reduces 128-d fingerprints to 16B vectors with minimal loss in detection accuracy ($\leq$1.2%).
- **Retrieval Latency:** On an A100 GPU, retrieval over a 1M fingerprint bank takes ∼73ms per 512-token query using Faiss with IVF-Flat index (k=32).
- **Memory Usage:** The full fingerprint bank (compressed) occupies ∼4.7GB for 1M sequences. Index construction time is ∼10 minutes (one-time).

Interpretation. These results show that FPGuard is practically efficient, enabling real-time inference and on-the-fly attribution even on large-scale corpora.

5.7. Cross-domain, Cross-model, and Multimodal Generalization (Q4)

We evaluate FPGuard's generalization under distribution shifts along three axes: (i) **domain shift**, (ii) **model shift**, and (iii) **modality shift**. Additionally, we assess its performance in **zero-shot detection** where no contamination is seen during fingerprint collection.

- **Cross-domain**: FPGuard is trained on SQuAD→Pile and evaluated on TriviaQA→RedPajama. Despite QA domain and phrasing changes, FPGuard retains a high Precision@10 of 78.6 and AUC of 87.4.
- **Cross-model**: We extract fingerprints using GPT-J and apply attribution to queries from LLaMA-2. FPGuard achieves 74.8 Precision@10 and a low score of 0.26, demonstrating that fingerprint representations generalize across model architectures.
- **Zero-shot detection**: Without seeing any contaminated samples during training, FPGuard is applied to contaminated queries. It still achieves 73.9 Precision@10 and 83.0 AUC, showing strong resilience under unseen leakage patterns.
- **Multimodal transfer**: We extend FPGuard to image-caption contamination. Fingerprints are collected from COCO captions and evaluated on contaminated LAION samples. FPGuard achieves 76.4 Precision@10 and a contamination score of 0.25, validating its applicability beyond text-only generation.

Q4 Result. As shown in Table 7, FPGuard maintains high attribution performance across diverse generalization scenarios. In all settings, the contamination score remains low and detection precision remains above 73%, even under zero-shot and cross-modal conditions.

These findings affirm **Q4**: FPGuard's fingerprints are model-agnostic, domain-robust, and modality-transferable, making

Table 7. **Generalization evaluation across domains, models, and modalities.** FPGuard maintains high performance without retraining.

Setting	Precision@10 ↑	AUC ↑	Score ↓
Cross-domain (SQuAD→Pile ⇒ TriviaQA→RedPajama)	78.6	87.4	0.24
Cross-model (Train on GPT-J ⇒ Test on LLaMA-2)	74.8	85.6	0.26
Zero-shot (Clean ⇒ Contaminated)	73.9	83.0	0.27
Multimodal (COCO→LAION captions)	76.4	84.2	0.25

it well-suited for real-world deployment in open-world and heterogeneous settings. These extended experiments address prior concerns on the clarity of contrastive learning (§Ablation), cost of fingerprinting (§Scalability), and depth of detoxification analysis (§Post-hoc Detox). Together, they reinforce FPGuard's applicability to real-world, large-scale generative model auditing.

6. Conclusion

We present **FPGuard**, an efficient and interpretable fingerprint-based contamination detection and attribution framework for generative language models. FPGuard extracts compact token-level fingerprints from multi-layer hidden states and performs attribution via approximate nearest neighbor search over a compressed fingerprint bank. It identifies memorized generations with high precision, while supporting span-level attribution and provenance tracing.

FPGuard enables post-hoc detoxification by removing high-score fingerprints, achieving up to 43.5% reduction in contamination score without model retraining. Extensive experiments show that FPGuard improves detection precision by up to $+25.4\%$ and attribution accuracy by $+30\%$ over prior methods. FPGuard generalizes across domains, model architectures, and modalities, making it a practical solution for contamination auditing in real-world LLM deployment.

References

Biderman, S., Prashanth, U., Sutawika, L., Schoelkopf, H., Anthony, Q., Purohit, S., and Raff, E. Emergent and predictable memorization in large language models. *Advances in Neural Information Processing Systems*, 36: 28072–28090, 2023.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al. Extracting training data from large language models. In *30th USENIX security symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

Deng, C., Zhao, Y., Tang, X., Gerstein, M., and Cohan, A. Investigating data contamination in modern benchmarks for large language models. *arXiv preprint arXiv:2311.09783*, 2023.

Fang, J., Bi, Z., Wang, R., Jiang, H., Gao, Y., Wang, K., Zhang, A., Shi, J., Wang, X., and Chua, T.-S. Towards neuron attributions in multi-modal large language models. *Advances in Neural Information Processing Systems*, 37: 122867–122890, 2024.

Ginart, A., Guan, M., Valiant, G., and Zou, J. Y. Making ai forget you: Data deletion in machine learning. *Advances in neural information processing systems*, 32, 2019.

Golatkar, A., Achille, A., and Soatto, S. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9304–9312, 2020.

Hannun, A., Guo, C., and van der Maaten, L. Measuring data leakage in machine-learning models with fisher information. In *Uncertainty in Artificial Intelligence*, pp. 760–770. PMLR, 2021.

Kazoom, R., Lapid, R., Sipper, M., and Hadar, O. Don't lag, rag: Training-free adversarial detection using rag. *arXiv preprint arXiv:2504.04858*, 2025.

Koh, P. W. and Liang, P. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pp. 1885–1894. PMLR, 2017.

Ladhak, F., Durmus, E., and Hashimoto, T. Contrastive error attribution for finetuned language models. *arXiv preprint arXiv:2212.10722*, 2022a.

Ladhak, F., Durmus, E., and Hashimoto, T. Contrastive error attribution for finetuned language models. *arXiv preprint arXiv:2212.10722*, 2022b.

Li, R., Li, C.-T., and Guan, Y. Inference of a compact representation of sensor fingerprint for source camera identification. *Pattern Recognition*, 74:556–567, 2018.

Li, Y., Ye, J., Zeng, L., Liang, R., Zheng, X., Sun, W., and Wang, N. Learning hierarchical fingerprints via multi-level fusion for video integrity and source analysis. *IEEE Transactions on Consumer Electronics*, 70(1):3414–3424, 2024.

Longpre, S., Mahari, R., Chen, A., Obeng-Marnu, N., Sileo, D., Brannon, W., Muennighoff, N., Khazam, N., Kabbara, J., Perisetla, K., et al. The data provenance initiative: A large scale audit of dataset licensing & attribution in ai. 2023.

Nguyen, T. T., Huynh, T. T., Ren, Z., Nguyen, P. L., Liew, A. W.-C., Yin, H., and Nguyen, Q. V. H. A survey of machine unlearning. *arXiv preprint arXiv:2209.02299*, 2022.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Shi, W., Michael, J., Gururangan, S., and Zettlemoyer, L. Nearest neighbor zero-shot inference. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 3254–3265, 2022.

Shi, W., Ajith, A., Xia, M., Huang, Y., Liu, D., Blevins, T., Chen, D., and Zettlemoyer, L. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*, 2023.

Tsukimoto, H. Extracting rules from trained neural networks. *IEEE Transactions on Neural networks*, 11(2): 377–389, 2000.

Vice, J., Akhtar, N., Hartley, R., and Mian, A. Bagm: A backdoor attack for manipulating text-to-image generative models. *IEEE Transactions on Information Forensics and Security*, 2024.

Wang, X., Zhu, W., Saxon, M., Steyvers, M., and Wang, W. Y. Large language models are latent variable models: Explaining and finding good demonstrations for incontext learning. *Advances in Neural Information Processing Systems*, 36:15614–15638, 2023.

Yang, J., Yang, S., Gupta, A. W., Han, R., Fei-Fei, L., and Xie, S. Thinking in space: How multimodal large language models see, remember, and recall spaces. *arXiv preprint arXiv:2412.14171*, 2024.

Youssef, P., Zhao, Z., Braun, D., Schlötterer, J., and Seifert, C. Position: Editing large language models poses serious safety risks. *arXiv preprint arXiv:2502.02958*, 2025.

Yu, W., Iter, D., Wang, S., Xu, Y., Ju, M., Sanyal, S., Zhu, C., Zeng, M., and Jiang, M. Generate rather than retrieve: Large language models are strong context generators. *arXiv preprint arXiv:2209.10063*, 2022.

Yusuf, M. Memorization as a learning style: A balance approach to academic excellence. *OIDA International Journal of Sustainable Development*, 1(6):49–58, 2010.