# SeedPrints: Fingerprints Can Even Tell Which Seed Your Large Language Model Was Trained From

**Yao Tong**[1*] **Haonan Wang**[1*] **Siquan Li**[1] **Kenji Kawaguchi**[1] **Tianyang Hu**[2†]
[1] National University of Singapore [2] The Chinese University of Hong Kong, Shenzhen

## Abstract

Fingerprinting Large Language Models (LLMs) is essential for provenance verification and model attribution. Existing methods typically extract post-hoc signatures based on training dynamics, data exposure, or hyperparameters—properties that only emerge after training begins. In contrast, we propose a stronger and more intrinsic notion of LLM fingerprinting: **SeedPrints**, a method that leverages random initialization biases as persistent, seed-dependent identifiers present even before training. We show that untrained models exhibit reproducible token selection biases conditioned solely on their parameters at initialization. These biases are stable and measurable throughout training, enabling our statistical detection method to recover a model's lineage with high confidence. Unlike prior techniques, unreliable before convergence and vulnerable to distribution shifts, **SeedPrints** remains effective across all training stages and robust under domain shifts or parameter modifications. Experiments on LLaMA-style and Qwen-style models show that SeedPrints achieves seed-level distinguishability and can provide birth-to-lifecycle identity verification akin to a biometric fingerprint. Evaluations on large-scale pretrained models further confirm its effectiveness under practical deployment scenarios. These results suggest that initialization itself imprints a unique and persistent identity on neural language models, forming a true "Galtonian" fingerprint.

## 1 Introduction

LLM fingerprints have recently been proposed as a tool to identify, attribute, and trace LLMs by examining their observable behaviors [1, 2, 3, 4, 5]. Such methods aim to provide model owners with a verifiable link between a suspicious model and its putative original, enabling detection of model theft or unauthorized reuse [3, 6].

Much of this literature explicitly borrows the metaphor of biological fingerprints from Francis **Galton**'s *Finger Prints* (1892) [7]:

> *"A fingerprint is the pattern formed by friction-ridge skin on the fingertips; this ridge configuration is individually unique and essentially permanent across an individual's lifetime."*

The analogy suggests that an effective fingerprint should be both unique and persistent, present from the very moment of a model's "birth" at initialization. Yet most existing so-called fingerprinting approaches fall short of this standard [1, 2, 3, 4, 5, 6, 8, 9, 10]. They are defined only after models have been fully trained and converged (finish pretaining), e.g., extracting patterns from parameters or generated text, and thus capture traits that emerge as the model "grows up" through exposure to data and optimization. In other words, the separability these methods can achieve is less a birthmark of

[*]Equal contribution.
[†]Correspondence to Tianyang Hu. Email: hutianyang@cuhk.edu.cn

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Lock-LLM Workshop.

the model itself than an imprint of its training history: data signatures, optimization dynamics, or hyperparameters. Such methods, therefore, function more as post hoc identifiers than as **Galtonian fingerprints**—those innate, immutable marks that accompany a model from its very beginning.

In this work, we propose a stricter notion of an LLM fingerprint: an intrinsic property present at random *initialization* and detectable in *any* time of the subsequent training. We observe that untrained models already exhibit seed-dependent preferences in token selection. The same prompt induces subtle but reproducible biases in next-token probabilities, and that these preferences remain measurable. Building on this observation, we introduce the evaluation method **SeedPrints**, designed to isolate initialization-borne fingerprints from confounds arising from data distribution, training duration, optimization noise, or objective choice. Under this lens, many existing "fingerprint" methods collapse (see Section 5.1): their discriminative signal weakens or disappears when models are examined from the early stage of training, or when descendants undergo severe distribution shifts in their training data, indicating that they capture signal related to the subtle training attribute, such as data distribution and hyperparameters, rather than intrinsic marks.

In contrast, extensive experiments in Section 5.1 show that our method can even distinguish between models that differ only in their initialization seed, despite an identical training pipeline and data order. Our method reliably detects the lineage of early-stage models with high confidence (e.g., $p$-values ranging from $10^{-3}$ to $10^{-10}$), whereas all prevailing baselines fail to separate them. Moreover, our fingerprint remains persistent under continual training on diverse datasets, while prior baselines merely track the training data distribution and are easily misled by suspicious models that have been continuously trained on very different corpora (Section 5.1). Finally, when evaluated under standard pretrained foundation model scenarios, our method successfully passes all tests (Section 5.2).

## 2 Related Work

LLM fingerprinting broadly falls into two families: (i) *watermarking / active* methods that *insert* an identifiable signature into a model or its outputs [2, 11, 9], and (ii) *passive* methods that *extract* a signature from a model's pre-existing behaviors without modifying it [3, 4, 10, 1, 12].

**Watermarking and Active Fingerprinting** Active approaches deliberately implant a verifiable identifier for later ownership checks. Two common forms are: *text watermarks*, in which biased or predefined text are generated to deliver a secret information [2, 11]; and *model weight watermarks*, which embed identifiers into parameters or tie them to a secret trigger via fine-tuning [8]. Although, backdoor-style fingerprints are straightforward and can persist through moderate fine-tuning; those invasive schemes require control over training, making them unsuitable for retroactively marking third-party models.

**Passive LLM Fingerprinting** In contrast, passive fingerprinting identifies models by analyzing their intrinsic properties without any modification. Passive fingerprinting techniques vary by model access. With white-box access, signatures are extracted from model weights, leveraging intrinsic properties like the distribution of attention matrices [3], the kernel alignment of internal representations [4], or the stable direction of parameter vectors. In the black-box setting, fingerprinting relies on analyzing input-output behavior. These methods use crafted queries [1], unique prompt-response pairs [9], or stylometry [10] to identify a model, though they can be less robust to fine-tuning. However, these methods define their signatures *post hoc*. Specifically, they identify emergent properties from a completed training process, rather than the innate, "*Galtonian*" fingerprints present from random initialization that our work seeks to discover.

## 3 Biases Originating from Initialization in Language Models Persist After Training

In this section, we present our key observation that language models exhibit strong token-level biases originating from the random initialization seed. Remarkably, such biases are stable and persistent, remaining detectable even after training.

**Initialization creates token-level biases** In Figure 1 (left), we examine a LLaMA-2–style model initialized with seed 123. We evaluate 10,000 random input sequences, each 1,024 tokens long with tokens drawn uniformly at random from the vocabulary. Surprisingly, despite the uniform inputs, the initialized model exhibits strong token-selection bias: it does not predict the next token (the 10,001st

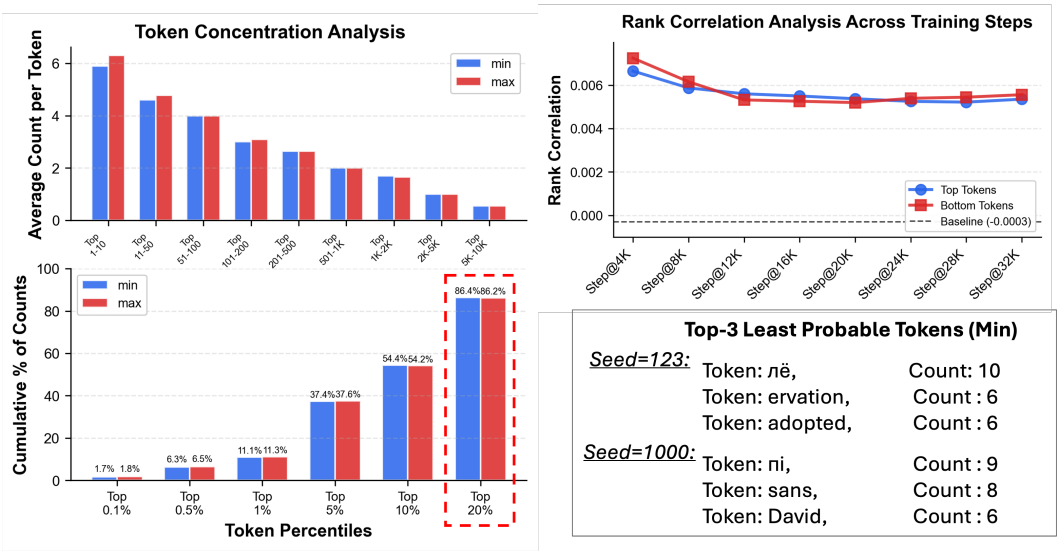


Figure 1: **Initialization-born token bias persists through training.** *Upper Left:* With uniform random inputs, a randomly initialized LLaMA-2–style model assigns highly non-uniform next-token probabilities, concentrating on a subset of tokens. *Lower Left:* An 80/20 coverage pattern: roughly 20% of tokens are selected for the next-token of 80% inputs. *Upper Right:* During training, these within-set preference remain aligned with initialization; rank correlations between checkpoints and the initial model stay above a random baseline. *Lower Right:* The set of preferred (biased) tokens depends on the initialization seed.

token following the inputs) uniformly at random; instead, certain tokens receive substantially higher probability than expected. To broaden this observation, we repeat the experiment with different initialization seeds and across different GPU types, and observe the same phenomenon: the overall magnitude of bias is consistent, and for a fixed seed the set of most biased tokens remains stable. However, the particular tokens that are favored (or disfavored) depend on the seed, as shown in the lower-right panel of Figure 1. Notably, an analogous pattern holds for the least-probable tokens (i.e., the argmin of next token probability). We refer to these high- and low-bias tokens as *identity tokens*, as they provide the strongest seed-specific signature at initialization.

**Training reshapes but does not erase those seed-specific token-level preferences** However, naively measuring bias by frequency is unreliable throughout the training, as the training will severely changes the next-token prediction behaviors. We ask whether identity tokens retain bias profiles that are deeply embedded in the model across training, like a lifelong fingerprint. To investigate, we examine the prediction distribution of each biased token over random sequences. Our expectation is that, since biased tokens each have specific random sequences where they are particularly likely or unlikely to be predicted, the relative preferences among tokens within such identity token set will persist to some extent across all offspring models. For example, if a token is much less preferred than other fingerprint tokens for a given random sequence, it will likely remain less preferred even after training. To test this, in Figure 1 (upper-right), for each checkpoint, we feed the same 10,000 random sequences as in the Figure 1 (left). For every identity token, we extract the within-set probability assigned to that token across these 10,000 random trials. We then compute the rank-based correlation between the probability vector from the initialized model and that from a checkpoint model trained from it. We report the average correlation across each selected token set. Across all token sets, we observe that although the absolute correlation values are small ($\approx 0.006$) because training mostly reshapes token probabilities, the correlation remains consistently above the random baseline. It will converge to a stable value rather than decay to the random baseline. This indicates that initialization leaves weak but statistically reliable, idiosyncratic fingerprints (on the prediction distribution of each biased token over random sequences).

## 4 Algorithm

To bridge the gap between the observed token-level bias and the "Galtonian" LLM fingerprints, we propose a statistical detection algorithm grounded in the observations of Section 3. Specifically, we

(i) extract the set of *identity indices* based the model outputs; and (ii) assess whether the preference distributions on these identity indices between a base model and a suspicious model are correlated or not, by comparing with an uncorrelated baseline. The design is in order to reflect the premise: *token-level biases on random inputs encode an intrinsic model-specific signature*.

**Extract identity indices** To obtain stable identity indices, we randomly select $n$ newly added tokens whose corresponding token vectors are freshly initialized. This differs slightly from sampling tokens from the existing vocabulary to form input sequences. The advantage is that it eliminates cases where the training data, by coincidence or design, contains pieces or subsequences of the test inputs—situations in which the model may recall memorized text and bias the generated tokens.

We treat each input as a next-token prediction trial and extract the *identity indices*, i.e., the output dimensions that consistently receive the lowest scores across random trials. Let $X = \{x_i\}_{i=1}^{n}$, where each $x_i \in \mathbb{R}^{\ell \times d}$ is a simulated embedding sequence of length $\ell$ in a $d$-dimensional embedding space. For any model $g$, define the mean output vector $\bar{g} \coloneqq \frac{1}{n}\sum_{i=1}^{n} g(x_i) \in \mathbb{R}^{d_{\text{out}}}$, where $g$ is either the base model $f$ or the suspect model $f'$, and $d_{\text{out}}$ denotes the output dimensionality (vocabulary size for logits, or hidden size for the final hidden state).[3] We extract the $m$ coordinates with the smallest mean values as the identity indices. Formally,

$$\mathcal{M} = \underset{J \subseteq \{1,\ldots,d_{\text{out}}\},\, |J|=m}{\arg\min} \sum_{j \in J} \bar{g}_j, \tag{1}$$

so $\mathcal{M}$ contains exactly the indices of the $m$ smallest entries of $\bar{g}$. Although both the most- and least-preferred dimensions (respectively, $\mathrm{argmax}$ and $\mathrm{argmin}$) can be informative in Figure 1, cross-entropy training typically increases the probability mass on the true class while decreasing it on non-true classes. Averaged over random inputs, the down-weighted (non-true) dimensions therefore exhibit more stable behavior. We thus focus on the most *unwilling* indices (the $\mathrm{argmin}$ set) to obtain more stable and reliable detections.

**Distribution correlation test** A naive way to compare a base model $f$ and a suspect model $f'$ is to measure the overlap of their identity–index sets, but this throws away magnitude/ranking information and is brittle to extraction noise. Motivated by Figure 1 (right)—which shows that relative preferences over identity indices persist after training—we instead test whether the two models exhibit *correlated preferences* on these indices across random inputs.

Let $\mathcal{T} \coloneqq \mathcal{M}_f \cap \mathcal{M}_{f'} = \{t_1, \ldots, t_k\}$ be the intersection of their identity–index sets. For each input $x_i$ and index $t_j$, record the (logit or hidden) output as $P_{ij}^{(f)} \coloneqq [f(x_i)]_{t_j}$ and $P_{ij}^{(f')} \coloneqq [f'(x_i)]_{t_j}$. This forms two output matrices $P^{(f)}, P^{(f')} \in \mathbb{R}^{n \times k}$. For each $t_j$, we compute a rank-based (Kendall–Tau) correlation $\rho_j = \mathrm{KendallTau}\big(P_{:,j}^{(f)}, P_{:,j}^{(f')}\big)$, yielding $k$ empirical correlation observations. If these correlations are consistently significant relative to a null of no association (tested against an uncorrelated baseline), we deem $f'$ to be derived from $f$. We declare significance at $p < 0.01$; further details are given in Algorithm 1.

**Design choices in algorithm** We next explain several design decisions in Algorithm 1, including the use of softmax, the correlation measure, and the hypothesis test. Since model outputs can vary substantially across different training stages, and our focus is only on the relative probabilities over the identity index set, we apply a softmax normalization in lines 8 and 13 to preserve the relative probability relationships. For the correlation measure, we use the standard Kendall–Tau correlation, a commonly used rank-based measure. This is a natural choice because absolute probability values are less reliable across models, whereas the bias profiles are mainly characterized by the pairwise concordance and discordance of rankings. Finally, regarding the hypothesis test, we found our method to be robust to the specific choice of test. In Section 5, we report results using both the one-sided $t$-test and the Mann–Whitney $U$-test.

[3] Using the final hidden representation instead of logits avoids detokenization noise and is more robust to the rare case that a random sequence appears in the training data.

Table 1: Comparison of fingerprint behaviors between models initialized with different seeds.

| Seed Pair | Logits Output | | Hidden State | |
|---|---|---|---|---|
| | *t-test* | *U-test* | *t-test* | *U-test* |
| $s_{42}$ vs. $s_{2000}$ | 0.404 | 0.456 | 0.357 | 0.532 |
| $s_{123}$ vs. $s_{42}$ | 0.214 | 0.295 | 0.678 | 0.565 |
| $s_{1000}$ vs. $s_{123}$ | 0.219 | 0.246 | 0.363 | 0.335 |
| $s_{2000}$ vs. $s_{1000}$ | 0.282 | 0.291 | 0.434 | 0.481 |

Table 2: Trained models share the same fingerprint behaviors as their initialization ($p$-value < 0.01).

| Model Pair | Logits Output | | Hidden State | |
|---|---|---|---|---|
| | *t-test* | *U-test* | *t-test* | *U-test* |
| $s_{42}^{init}$ vs. $s_{42}^{base}$ | 3.33e-3 | 1.02e-3 | 2.20e-8 | 6.28e-8 |
| $s_{123}^{init}$ vs. $s_{123}^{base}$ | 2.06e-3 | 7.33e-3 | 7.09e-6 | 1.37e-5 |
| $s_{1000}^{init}$ vs. $s_{1000}^{base}$ | 2.44e-3 | 4.14e-3 | 5.58e-4 | 2.81e-3 |
| $s_{2000}^{init}$ vs. $s_{2000}^{base}$ | 5.63e-3 | 6.76e-3 | 4.00e-10 | 1.27e-9 |

# 5 Experiments

We evaluate our fingerprint verification in two ways: (1) it identifies genuine, biometric-like fingerprints at the seed level, whereas prior methods primarily track training-data distributions; and (2) it remains verifiable throughout all training stages. We report results using both logits and hidden-state outputs, and we test with a one-sided $t$-test and the Mann–Whitney U test.

**Baselines** We consider four passive fingerprinting baselines (weight- or representation-based). **Intrinsic fingerprint** [3] compares models via the similarity of the layerwise standard-deviation profiles of attention parameters. **REEF** [4] computes centered-kernel-alignment (CKA) similarity between feature representations from the same samples across two models. **PCS** and **ICS** [5] are weight-similarity methods: PCS flattens all parameters and measures cosine similarity; ICS forms invariant terms from the weights and measures cosine similarity on those invariants. Following [4], we use a 0.8 similarity threshold for binary decisions.

Note, in the experiment tables, the cell color indicates whether two models originate from the same initialization seed. For example, $s_{42}^{init}$ vs. $s_{42}^{base}$ compares a model initialized with seed 42 and its counterpart after pretraining; since both share the same initialization, the cell is shaded green. In contrast, $s_{2000}^{init}$ vs. $s_{42}^{base}$ compares models initialized with different seeds, and is therefore shaded red to indicate different sources. Additionally, ✓ denotes a correct detection, while × denotes an error.

## 5.1 Seed-level differentiation and reliability

We train 12-layer, 12-head LLaMA-style models [13] with RoPE [14] from scratch. Because the baselines are stochastic, we report $p$-values averaged over 10 independent trials and adopt a significance level of $\alpha = 0.01$. Note, the absolute magnitude of extremely small $p$-values is not comparable: once $p$ falls below numerical noise (e.g., $< 10^{-20}$), values like $10^{-260}$ should not be interpreted as stronger evidence than $10^{-20}$—both decisively reject the null.

**Different initialization seeds produce distinct fingerprints** Table 1 reports $p$-values from our correlation tests between pairs of models initialized with different random seeds. Across all pairs, $p > 0.01$, meaning we do not detect correlation in their preference profiles. Thus, different seeds yield distinct fingerprint behaviors, and our method separates models "at birth."

Figure 2: Fingerprint verifies lineage at every checkpoint ($p$-values $< 0.01$).

Table 3: The same dataset and training order do not shape fingerprint behaviors to be identical across different initializations.

| Model Pair | Logits Output | | Hidden State | |
|---|---|---|---|---|
| | *t-test* | *U-test* | *t-test* | *U-test* |
| $s_{123}^{init}$ vs. $s_{1000}^{base}$ | 0.484 | 0.500 | 0.385 | 0.486 |
| $s_{1000}^{init}$ vs. $s_{2000}^{base}$ | 0.946 | 0.956 | 0.035 | 0.096 |
| $s_{42}^{init}$ vs. $s_{123}^{base}$ | 0.598 | 0.589 | 0.426 | 0.337 |
| $s_{2000}^{init}$ vs. $s_{42}^{base}$ | 0.756 | 0.781 | 0.388 | 0.287 |

Table 4: Fingerprint persistence under continual training on diverse datasets (base model: seed 1000, corpus `openwebtext`). U test refers to the Mann–Whitney $U$ test.

| **Setting** | **Ours (logits)** | | **Ours (hidden)** | | **Baselines** | | | |
|---|---|---|---|---|---|---|---|---|
| **Continual corpus (seed)** | **t test** | **u test** | **t test** | **u test** | **Intrinsic** | **REEF** | **PCS** | **ICS** |
| `TinyStoriesV2_cleaned` (1000) | 0✓ | 0✓ | 0✓ | 7.77e-89✓ | 1.000✓ | 0.759✗ | 0.999✓ | 0.996✓ |
| `TinyStoriesV2_cleaned` (123) | 1.000✓ | 1.00✓ | 0.943✓ | 0.902✓ | 0.950✗ | 0.658✓ | 0.332✓ | 0.012✓ |
| `the_stack` (1000) | 0✓ | 1.73e-287✓ | 0✓ | 3.09e-69✓ | 0.489✗ | 0.557✗ | 0.585✗ | 0.123✗ |
| `the_stack` (123) | 0.616✓ | 0.479✓ | 0.732✓ | 0.831✓ | 0.445✓ | 0.580✓ | 0.301✓ | 0.026✓ |

**Training preserves the initialization fingerprint.** Table 2 compares each initialization model with its descendant trained on OpenWebText ($\approx$ 10B tokens). Here, $p < 0.01$ consistently, indicating a strong correlation between their preference profiles and, hence, common lineage. In short, the trained model inherits the same fingerprint as its initialization. We also evaluate baseline methods (Table 6); without exception, they fail to distinguish across seeds, which in turn suggests their separability stems from training-induced artifacts rather than initialization.

**Identical data and order do not make fingerprints converge** In Table 3, all four "suspicious" models $s_i^{base}$ for $i \in \{42, 123, 100, 2000\}$ are trained on *exactly the same corpus (OpenWebText) and in the same data order* (we fix the training seed to lock the data order); the only difference lies in their initialization seeds $i$. Across all cross-seed pairs, $p$-values remain consistently $> 0.01$, in sharp contrast to the near-zero values in Table 2. That is, fingerprints remain seed-specific even under identical data and curriculum.

**Continual training on diverse datasets does not confound the fingerprint** A natural concern is that fingerprint detection might reflect *data distribution* rather than *model lineage*, and thus fail after further training on very different corpora. We therefore continue training on two starkly different datasets—TinyStories [15] (synthetic children's stories) and The Stack [16] (permissively licensed GitHub code). Starting from a base model pretrained on OpenWebText [17] (seed 1000), we compare: (i) a descendant obtained by further training the base on each corpus, versus (ii) a *distractor* descendant initialized with a different seed (123) and data order on OpenWebText [17], then continued on the *same* corpus. The question is whether attribution methods can identify which descendant truly shares lineage with the base. In Table 4, we find that prior baselines frequently fail under the code setting (The Stack), incorrectly signaling common lineage—***indicating they largely track domain similarity rather than real identity***. In contrast, our method correctly attributes lineage across both corpora, consistently yielding $p < 0.01$. Hence, the fingerprint is not a proxy for data distribution: it survives substantial domain shift and persists beyond the initial pretraining stage.

### 5.2 All-stage verifiable fingerprints

Our fingerprint enables verification at any training stage (Figure 2). We treat each intermediate checkpoint of a model trained on OpenWebText as the base and test whether our method can reliably identify its offspring. All variants consistently recognize the suspect model as belonging to the same lineage, with $p$-values remaining below the 0.01 threshold.

We further compare our method with existing baselines under standard evaluation for pretrained foundation models. In particular, we test suspect models that undergo additional training on increasing numbers of tokens (reported as #Tokens). As shown in Table 5, our method maintains $p < 0.01$ across all settings.

Table 5: Fingerprinting results *vs.* LLaMA-2-7B. Each row compares a target model against LLaMA-2-7B. ***U*-test** $p$ reports the $p$-value from our hidden-state correlation test ($< 0.01$ indicates a strong signal). **Intrinsic**, **REEF**, **PCS**, and **ICS** report similarity scores (higher = better).

| **Model** | **# Tokens** | ***U*-test** $p$ ($< 0.01$) | **Intrinsic** ↑ | **REEF** (↑) | **PCS** (↑) | **ICS** (↑) |
|---|---|---|---|---|---|---|
| Llama-2-finance-7B [18] | 5M | $1.34 \times 10^{-41}$✓ | 1.0000✓ | 0.9950✓ | 0.9979✓ | 0.9952✓ |
| Vicuna-1.5-7B [19] | 370M | $1.49 \times 10^{-96}$✓ | 1.0000✓ | 0.9985✓ | 0.9985✓ | 0.9949✓ |
| Wizardmath-7B [20] | 1.8B | $4.09 \times 10^{-100}$✓ | 1.0000✓ | 0.9979✓ | 1.0000✓ | 0.9994✓ |
| Meditron-7B [21] | 48B | $5.212 \times 10^{-4}$✓ | 0.9990✓ | 0.9978✓ | 1.0000✓ | 0.9817✓ |
| CodeLlama-7B [22] | 500B | $2.008 \times 10^{-3}$✓ | 0.9480✓ | 0.9947✓ | 0.6863✗ | 0.3369✗ |

## References


[1] Dario Pasquini, Evgenios M Kornaropoulos, and Giuseppe Ateniese. Llmmap: Fingerprinting for large language models. *arXiv preprint arXiv:2407.15847*, 2024.

[2] Jiashu Xu, Fei Wang, Mingyu Derek Ma, Pang Wei Koh, Chaowei Xiao, and Muhao Chen. Instructional fingerprinting of large language models. *arXiv preprint arXiv:2401.12255*, 2024.

[3] Do-hyeon Yoon, Minsoo Chun, Thomas Allen, Hans Müller, Min Wang, and Rajesh Sharma. Intrinsic fingerprint of llms: Continue training is not all you need to steal a model! *arXiv preprint arXiv:2507.03014*, 2025.

[4] Jie Zhang, Dongrui Liu, Chen Qian, Linfeng Zhang, Yong Liu, Yu Qiao, and Jing Shao. Reef: Representation encoding fingerprints for large language models. *arXiv preprint arXiv:2410.14273*, 2024.

[5] Boyi Zeng, Lizheng Wang, Yuncong Hu, Yi Xu, Chenghu Zhou, Xinbing Wang, Yu Yu, and Zhouhan Lin. Huref: Human-readable fingerprint for large language models. *Advances in Neural Information Processing Systems*, 37:126332–126362, 2024.

[6] Ruichong Zhang. Matrix-driven instant review: Confident detection and reconstruction of llm plagiarism on pc. *arXiv preprint arXiv:2508.06309*, 2025.

[7] Francis Galton. *Finger prints*. Number 57490-57492. Cosimo Classics, 1892.

[8] Xiaokun Luan, Zeming Wei, Yihao Zhang, and Meng Sun. Robust and efficient watermarking of large language models using error correction codes. *Proceedings on Privacy Enhancing Technologies*, 2025.

[9] Yun-Yun Tsai, Chuan Guo, Junfeng Yang, and Laurens van der Maaten. Rofl: Robust fingerprinting of language models. *arXiv preprint arXiv:2505.12682*, 2025.

[10] Saeif Alhazbi, Ahmed Hussain, Gabriele Oligeri, and Panos Papadimitratos. Llms have rhythm: Fingerprinting large language models using inter-token times and network traffic analysis. *IEEE Open Journal of the Communications Society*, 2025.

[11] Anshul Nasery, Jonathan Hayase, Creston Brooks, Peiyao Sheng, Himanshu Tyagi, Pramod Viswanath, and Sewoong Oh. Scalable fingerprinting of large language models. In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*.

[12] Teppei Suzuki, Ryokan Ri, and Sho Takase. Natural fingerprints of large language models. *arXiv preprint arXiv:2504.14871*, 2025.

[13] AI@Meta. Llama 3 model card. 2024.

[14] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 115–124. Association for Computational Linguistics, 2021.

[15] Ronen Eldan and Yuanzhi Li. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*, 2023.

[16] Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Carlos Muñoz Ferrandis, Yacine Jernite, Margaret Mitchell, Sean Hughes, Thomas Wolf, Dzmitry Bahdanau, Leandro von Werra, and Harm de Vries. The stack: 3 tb of permissively licensed source code. *Transactions on Machine Learning Research (TMLR), Preprint*, 2022.

[17] Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`, 2019.

[18] Collin Heenan. Llama2-7b-finance (hugging face model). `https://huggingface.co/cxllin/Llama2-7b-Finance`, 2023. Fine-tuned from NousResearch/Llama-2-7b-hf; MIT License; accessed 2025-09-02.

[19] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023.

[20] Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*, 2023.

[21] Zeming Chen, Alejandro Hernández-Cano, Angelika Romanou, Antoine Bonnet, Kyle Matoba, Francesco Salvi, Matteo Pagliardini, Simin Fan, Andreas Köpf, Amirkeivan Mohtashami, Alexandre Sallinen, Alireza Sakhaeirad, Vinitra Swamy, Igor Krawczuk, Deniz Bayazit, Axel Marmet, Syrielle Montariol, Mary-Anne Hartley, Martin Jaggi, and Antoine Bosselut. Meditron-70b: Scaling medical pretraining for large language models, 2023.

[22] Meta AI. Codellama-7b-hf: Code llama base 7b model (hugging face). `https://huggingface.co/codellama/CodeLlama-7b-hf`, 2024. Base 7 billion-parameter Code Llama model for code synthesis and understanding; trained between January and July 2023; licensed under Meta Llama 2 license; accessed 2025-09-02.

# A Appendix

## A.1 Algorithm Details

**Algorithm 1** Distribution Correlation Test on identity indices

**Require:** base model $f$, suspicious model $f'$; random input $X = \{x_i\}_{i=1}^{n}$; fingerprint size $m$; significance level $\alpha = 0.01$

***Extract $m$ indices with the smallest average model output value as identity indices set***

**for** model $g \in \{f,\ f'\}$ **do**

  $g(X) \leftarrow [g(x_1), \ldots, g(x_n)]^\top \in \mathbb{R}^{n \times d_{\text{out}}}$ (get model outputs on all random inputs)

  $\mathcal{M}(g) \leftarrow \arg\min_{J \subseteq \{1,\ldots,d_{\text{out}}\},\, |J|=m} \sum_{j \in J} \bar{g}_j$ (Equation (1)).

**end for**

***Get intersection of the two identity indices sets***

$T \leftarrow \mathcal{M}(f) \cap \mathcal{M}(f') = \{t_1, \ldots, t_k\}$

***Retrieve model outputs on this intersection***

**for** $g \in \{f, f'\}$ **do**

  $Z^{(g)} \leftarrow g(X)[T] \in \mathbb{R}^{n \times k}$ (restrict outputs to $T$)

  $P^{(g)} \leftarrow \text{softmax}(Z^{(g)}) \in \mathbb{R}^{n \times k}$ (compute the relative probability over $T$)

**end for**

***Compute the per-index correlation between models***

$\mathcal{T} \leftarrow \left\{ \text{KendallTau}\left(P^{(f')}_{:,j},\ P^{(f)}_{:,j}\right) : j = 1, \ldots, k \right\}$ (rank-based correlation)

***Build an uncorrelated random baseline $\mathcal{T}_{\text{null}}$***

**for** $g \in \{1, 2\}$ **do**

  Randomly sample $Z^{(g)}_{\text{rand}} \sim \mathcal{N}(0, I_k)^{n \times k}$

  $P^{(g)}_{\text{rand}} \leftarrow \text{softmax}(Z^{(g)}_{\text{rand}}) \in \mathbb{R}^{n \times k}$ (row-wise)

**end for**

$\mathcal{T}_{\text{null}} \leftarrow \left\{ \text{KendallTau}\left(P^{(1)}_{\text{rand},:,j},\ P^{(2)}_{\text{rand},:,j}\right) : j = 1, \ldots, k \right\}$

***Perform a one-sided hypothesis test***

$H_0 : \mathcal{T} = \mathcal{T}_{\text{null}}, \quad H_1 : \mathcal{T} > \mathcal{T}_{\text{null}}$

**Return** SameLineage $\leftarrow \mathbf{1}(p\text{-value} < \alpha)$

## A.2 More Experimental Results

Table 6: Full results of methods evaluating fingerprints at initialization and after subsequent training.

| Model Pair | Logits Output | | Hidden State | | Baselines | | | |
|---|---|---|---|---|---|---|---|---|
| | *t* | *u* | *t* | *u* | Intrinsic | REEF | PCS | ICS |
| $s_{42}^{init}$ vs. $s_{42}^{base}$ | 3.33e-3✓ | 1.02e-3✓ | 2.20e-8✓ | 6.28e-8✓ | -0.021× | 0.375× | 0.580× | 0.196× |
| $s_{123}^{init}$ vs. $s_{123}^{base}$ | 2.06e-3✓ | 7.33e-3✓ | 7.09e-6✓ | 1.37e-5✓ | 0.149× | 0.369× | 0.581× | 0.188× |
| $s_{1000}^{init}$ vs. $s_{1000}^{base}$ | 2.44e-3✓ | 4.14e-3✓ | 5.58e-4✓ | 2.81e-3✓ | -0.252× | 0.381× | 0.581× | 0.188× |
| $s_{2000}^{init}$ vs. $s_{2000}^{base}$ | 5.63e-3✓ | 6.76e-3✓ | 4.00e-10✓ | 1.27e-9✓ | -0.337× | 0.331× | 0.581× | 0.188× |