Towards Secure Model Sharing with Approximate Fingerprints

Anshul Nasery [1] **Sewoong Oh** [1]

Abstract

Large scale democratization of machine learning has made model sharing commonplace. This has also raised significant concerns around unauthorized usage, intellectual property violations, and model leakage. Model fingerprinting through memorization of fixed strings has emerged as a practical solution to address these challenges for LLMs. However, prior research on fingerprint robustness has largely overlooked realistic adversarial conditions, generally assuming that an adversary, unaware of fingerprint queries, cannot easily evade detection. We introduce a realistic adversarial threat model in which an attacker can uniformly modify the output distribution of an LLM, without degrading utility on benign inputs or requiring explicit knowledge of fingerprint queries to evade detection. Under this threat model, we present a novel family of sampling-based attacks capable of bypassing all existing fingerprinting schemes. To counteract these, we propose a new paradigm based on approximate fingerprint detection and memorization and provide concrete instantiations demonstrating their robustness and practicality. Our work highlights critical security vulnerabilities in current fingerprinting approaches and aims to encourage further research into robust fingerprinting methods resilient under realistic adversarial scenarios.

1. Introduction

Recent advances in large language models (LLMs) have led to widespread sharing and distribution of models across various applications and platforms. As model sharing becomes increasingly prevalent, fingerprinting has emerged as an important technique for identifying model leakage, verifying authorship, and preventing unauthorized usage (Cheng et al., 2024), and as a benign application of memorization.

[*]Equal contribution [1]University of Washington. Correspondence to: Anshul Nasery <anasery@cs.washington.edu>.

Published at ICML 2025 Workshop on Reliable and Responsible Foundation Models.

Although previous work has highlighted the potential security benefits of fingerprinting(Xu et al., 2024; Nasery et al., 2025), it has largely focused on persistence after fine-tuning (Russinovich & Salem, 2024), adversarial collusion (Nasery et al., 2025), model merging (Yamabe et al., 2024), or manipulation through system prompts (Jiaxuan et al., 2025). Crucially, these analyses have not rigorously explored fingerprint robustness in adversarial scenarios.

In this work, we propose a realistic adversarial threat model designed explicitly to assess the security of fingerprinting schemes. Specifically, we consider an adversary capable of arbitrarily altering the output distribution of an LLM, provided these alterations: (1) apply uniformly to all queries, (2) do not substantially degrade the utility of the model for legitimate (benign) requests, and (3) require no additional forward passes, maintaining computational efficiency. Importantly, our adversary operates without prior knowledge of whether a given query constitutes a fingerprint.

Under this threat model, we introduce a novel class of attacks that manipulate the sampling process during the generation of initial tokens. We demonstrate empirically that these attacks effectively circumvent all currently proposed fingerprinting methods, exposing their vulnerabilities. Our analysis identifies a fundamental limitation in existing fingerprinting approaches—they depend on exact memorization and regurgitation of specific, unrelated sequences.

To overcome this limitation, we propose a new fingerprinting paradigm based on approximate detection rather than exact memorization. Within this framework, we introduce three concrete instantiations designed to robustly defend against the proposed attacks. Through extensive experimentation, we validate that our approximate fingerprinting schemes maintain high model utility while offering significant resilience against adversarial manipulations. We further demonstrate that our proposed solutions scale efficiently and persist effectively across diverse conditions.

Our contributions are the following -

- We introduce a realistic adversarial threat model targeting active fingerprinting schemes (Section 3.1) and propose a novel family of sampling-based attacks effective against existing fingerprinting methods (Section 3.2).
- We propose a robust mitigation framework employing

approximate fingerprinting, along with three practical instantiations designed for resilience under adversarial sampling (Section 4).

- We empirically validate the effectiveness and scalability of our fingerprinting solutions, demonstrating low utility loss and high robustness under attacks (Fig. 2).

2. Related Works

There has been much recent interest in fingerprinting generative LLMs to detect model stealing. The main idea is to fine-tune the LLM on unrelated $(\text{key}, \text{response})$ pairs to memorize and regurgitate them. The model can then be authenticated by checking if its output matches the appropriate response when prompted with the fingerprint key.

Xu et al. (2024) introduced the problem of fingerprinting in both white-box (i.e. with access to model weights) and black-box (i.e. access only to an API) settings. Russinovich & Salem (2024) study a setting where model owners can also be adversarial and can falsely claim another model as their own. Another work (Jiaxuan et al., 2025) proposes a scheme for generating implicit fingerprints using model steganography. Other works propose model merging as an attack against fingerprint detection (Yamabe et al., 2024; Cong et al., 2023). Finally, Nasery et al. (2025) focus on scaling the number of fingerprints that can be memorized by generating sensible but uncommon fingerprints.

3. Attack Framework

3.1. Threat Model

We now describe a scheme to bypass detection for current LLM fingerprinting methods. We first list the assumptions of our threat model

- The fingerprinting protocol is public, and the adversary can replicate the algorithm used to generate fingerprints.
- The adversary aims to minimize the loss in utility of the model on benign queries.
- The adversary aims to be efficient, i.e. they do not call the LLM multiple times, and cannot call an external LLM to generate the response.

Note that for good fingerprinting schemes, the adversary cannot in general distinguish between fingerprinted and non-fingerprinted queries, and uniformly applies the attack strategy to all queries. When this is violated e.g. by Xu et al. (2024); Russinovich & Salem (2024) and the adversary can filter out and refuse fingerprint queries easily.

Adversarial goals The primary goal of the adversary is to evade detection. A secondary goal could be to increase the false positive rate of the fingerprints.

3.2. Sampling Based Attacks

In this section, we introduce a novel family of attacks designed to evade existing fingerprinting methods by strategically modifying token sampling at the start of generation. Specifically, these attacks perturb sampling only for the first few tokens, minimally impacting utility, as demonstrated in Fig. 1. Despite their simplicity, these techniques effectively break current detection schemes, as illustrated by the results presented in Table 1.

ImprobableToken (IT) attacks For sampling the first n response tokens, the adversary discards the top-k most probable tokens and samples from the tail. After this, they sample in the usual manner. Since fingerprinting methods check for memorized strings starting from the first token, this attack trivially has a 100% ASR against existing fingerprinting schemes for $k \geq 1$. Similar samplers have been proposed for creative writing(p-e w, 2024).

BlockTopWord (BTW) attacks A potential defense against the ImprobableToken attack is to allow fingerprint detection if the model outputs the fingerprint response within the first m tokens, rather than strictly at the start. Russinovich & Salem (2024) propose training fingerprinted models with appended random strings to facilitate this flexible matching. However, upon closer analysis, we find that even with this augmentation, the fingerprint response typically remains the *most probable initial token*. An attacker can thus evade detection by not outputting tokens lexically close to the most probable initial token in its response, leading to the fingerprint response never being emitted. The attacker applies this sampling strategy for the first n tokens and then resumes normal sampling. This can be extended to consider the k most probable initial tokens for matching. We call this approach the BlockTopWord attack, parameterized by the lexical set size k and the number of perturbed tokens n.

These attacks alter the initial tokens of the response to evade memorization-based detection methods (by preventing the generation of an exact fingerprint response string), while minimizing perturbations to preserve utility.

3.3. Utility Trade-off

Since these attacks make change the sampling algorithm, we measure the utility drop induced by these for different values of n, k on an OLMO-2-7B-Instruct model (OLMo et al., 2025). We measure the performance on BBH (Srivastava et al., 2022) and IFEval (Zhou et al., 2023) in Fig. 1. We find that the utility of IT and BTW attack drops quickly with larger n. The utility also drops for large values of k, as this samples less probable responses. Hence, we report the ASR for $k = 1, n \leq 16$ for our defenses.

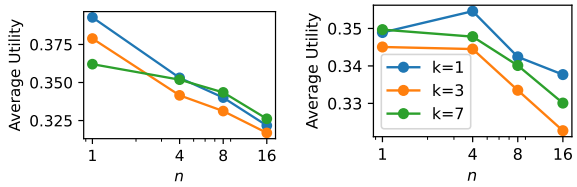

Figure 1. **Utility of model under attack** We find that utility does not suffer much under IT (left) or BTW (right) attacks for mild values of parameters. As n increases beyond 8 the utility drops significantly. As k increases from 1 to 7 it drops, esp for low n. Hence, we focus on settings where $k = 1$ and $n \leq 16$.

FP Scheme	ASR (IT)	ASR (BTW)
Instructional (Xu et al., 2024)	100%	100%
ImF (Jiaxuan et al., 2025)	100%	100%
Chain&Hash (Russinovich & Salem, 2024)	92%	98%
Perinucleus (Nasery et al., 2025)	97%	95%
Perinucleus-Multi (Nasery et al., 2025)	94%	95%

Table 1. **Attack success rates (ASR) for different fingerprinting schemes using IT and BTW attacks**. We find that our attacks can achieve a very high ASR against existing fingerprint schemes

3.4. Results against existing schemes

In Table 1, we report the Attack Success Rate of the proposed attacks against fingerprinting schemes proposed in the literature (Russinovich & Salem, 2024; Jiaxuan et al., 2025; Xu et al., 2024; Nasery et al., 2025). Here, ASR refers to the number of fingerprints which were not detected, i.e. for which the model did not produce the corresponding response under attack. We note that since these schemes all map a query to a single, fixed response, IT with $n = 1$ itself is a powerful attack on them, achieving almost a 100% ASR. Further, since the most probable token in the response is also invariably the fingerprint response in all these models, the BTW attack can identify this response token and prohibit the model from producing it.

We argue that the above attacks are successful because of two reasons – *(i)* the fingerprinting schemes map each query to a **single, fixed** response, so perturbing the response slightly will ensure evasion of detection *(ii)* the first token of the fingerprint response is always the most probable first token in the model output, which means it can be suppressed leading to a successful attack. Prior works have proposed mitigations for these, however, they are insufficient.

Nasery et al. (2025) propose a defense against such sampling based attacks which maps a single fingerprint query to multiple responses (Perinucleus-Multi), however, a larger value of k can also achieve a high ASR on this mitigation without affecting utility. Russinovich & Salem (2024) also have a possible mitigation to the IT attack - the model query has a suffix of random tokens appended to it as augmentation for better robustness. However, as we describe above, BTW attack can still avoid detection, since the most probable response token is still the fingerprint response.

4. Approximate Fingerprints

4.1. Fingerprint detection

In order to overcome the above-mentioned limitations of prior work, we propose a new paradigm of model fingerprinting - for each fingerprint query x_{fp}, define a function $f(\cdot, x_{\text{fp}})$ which takes a response string as input and returns 1 or 0, denoting if the response string satisfies the fingerprint. Note that for prior schemes, this function is simply an indicator function checking if the response string is equal to the corresponding pre-defined fingerprint response. We now describe some instantiations of such rule-based schemes.

Word in response We associate a word w_{fp} with each fingerprint x_{fp}. Denoting $\{a[1:k]\}$ as the set comprising of the first k words of a string a, the detection function f is defined as $f(y, x_{\text{fp}}) = \mathbb{I}(w_{\text{fp}} \in \{y[1:k]\})$, i.e. it checks if the word w_{fp} is present in the first k words of the response y.

M out of N words in response We can also generalize the previous scheme to work with multiple words. Formally, for each each fingerprint x_{fp}, we denote $W_{\text{fp}} = (w_{\text{fp}}^1, \cdots, w_{\text{fp}}^N)$ to be a set of N response words. Then the detection function is defined as $f(y, x_{\text{fp}}) = \mathbb{I}(\{y[1:k]\} \cap W_{\text{fp}} \geq M)$, i.e. at least M words of W_{fp} appear in the first K words of the response. If $M = N = 1$, we recover the previous rule.

Note that we can also apply these detection scheme to existing fingerprinting schemes (Nasery et al., 2025; Russinovich & Salem, 2024), by setting the word(s) w_{fp} to be the fingerprint response(s) y_{fp}. However, as we argue above, the BlockTopWord attack on these existing approaches will evade detection since the response token y_{fp} is easily identified as the most probable token in the model output.

Conditional Watermark A more general form of f is to check if the words in the response follow a particular distribution. Text watermarking (Kirchenbauer et al., 2023; Gu et al., 2024) provides a natural framework for this. Concretely, for each fingerprint x_{fp}, we partition the vocabulary into "green" and "red" sets $G_{\text{fp}}, R_{\text{fp}}$ so that $\frac{|G_{\text{fp}}|}{|R_{\text{fp}}|} = \frac{\gamma}{1-\gamma}$. Then, the detector is defined as $f(y, x_{\text{fp}}) = \mathbb{I}(|\{y[1:k]\} \cap G_{\text{fp}}| > \gamma k + l)$, where l controls the power of the test. Informally, this measures if the number of green words appearing in the response is more than their expected number (γk). We set $\gamma = 0.25, l = 1$ for our experiments.

4.2. Inserting Approximate Fingerprints into models

To insert fingerprints into an LLM, we perform SFT with regularization (Nasery et al., 2025) on fingerprints.

Fingerprint generation We first generate a set of fingerprint queries $\{x_{\text{fp}}\}$ by prompting an LLM to produce a question

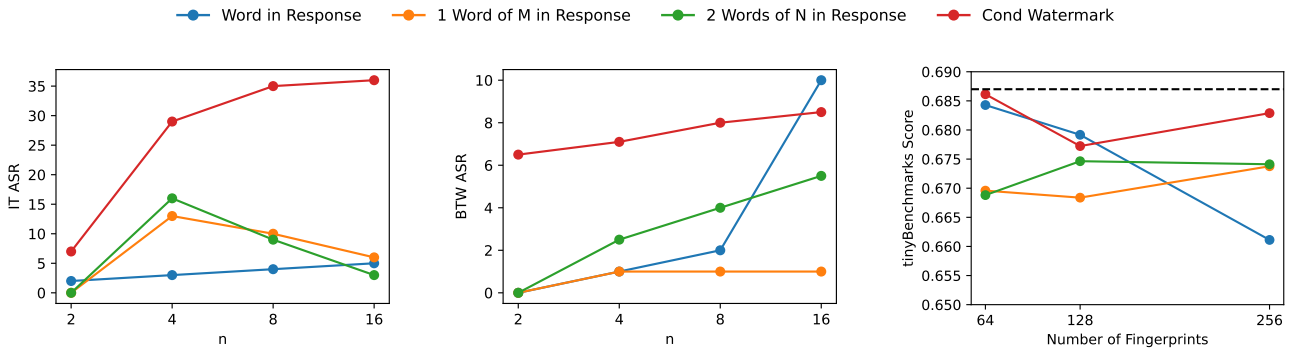

Figure 2. **ASR and Scalability of Approximate fingerprints**. In the left and center plots, we show the ASR (↓) of the IT and BTW attacks with $k = 1$ against different fingerprinting schemes at 128 fingerprints. We find that these defenses are effective across different values of n. *(Right)* Model utility versus number of fingerprints. The relative drop in performance is less than 5% at 256 fingerprints.

that a user might ask a chat-bot. Then for each query, we produce a set of responses $\{y_{\text{fp}} : f(y_{\text{fp}}, x_{\text{fp}}) = 1\}$ (from the model to be fingerprinted) which conform to the rule. To do this we associate a word, set of words or vocabulary partition with each fingerprint query randomly, seeding the randomness with the hash of x_{fp}. For generality, we call this set of tokens/words as R_{fp} Next, we prompt the base model with x_{fp}, and sample a response autoregressively from $softmax(F_\theta(x_{\text{fp}}) + \delta \mathbf{1}_{\{t \in R_{\text{fp}}\}})$, where $\mathbf{1}_{\{t \in R_{\text{fp}}\}}$ is a vector having 1 for tokens $t \in R_{\text{fp}}$ and 0 otherwise, and $\delta \in \mathbb{R}^+$. This biases the output distribution towards words contained in R_{fp}, while still being close to the output of the original language model. We then discard any sequences y where $f(y, x_{\text{fp}}) = 0$. We collect multiple (upto 8) positive y per x_{fp} for our training set.

Soft loss A naive algorithm is to perform SFT on the generated $(x_{\text{fp}}, y_{\text{fp}})$ pairs. However, we find that this is insufficient as the model simply memorizes the set of fingerprint responses, leading to similar vulnerabilities as existing schemes. Hence, we add a soft loss to boost the probabilities of the required words in the model's outputs even under attack. Since all our rules in the current formulation include increasing the output probability of some subset of tokens, we add a term for the cross-entropy loss on these tokens. Formally, our loss is

$$L(\theta; x_{\text{fp}}, y_{\text{fp}}, R_{\text{fp}}) = (1-\lambda)\, l_{\text{ce}}\left(F_\theta(x_{\text{fp}}), y_{\text{fp}}\right) + \lambda\, l_{\text{ce}}\left(F_\theta(x_{\text{fp}}), \frac{\mathbf{1}_{\{t \in R_{\text{fp}}\}}}{|R_{\text{fp}}|}\right)$$

where $l_{\text{ce}}(\cdot, \cdot)$ is the cross-entropy loss.

5. Evaluating our Defenses

Setup We show the efficacy of our proposed schemes on OLMo-2 7B Instruct. We generate up to 256 fingerprints, add them to the models, and compute the ASR of our proposed attacks on these schemes. We also investigate the Scalability (Nasery et al., 2025) of the approximate fingerprints by measuring the utility of the fingerprinted models as a function of the number of inserted fingerprints on tinyBenchmarks (Polo et al., 2024).

Security In Fig. 2, we find that the proposed defenses have a low ASR (compared to Table 1) for both the IT and BTW attacks for $n \leq 8, k = 1$. We find that BTW is not as effective against these schemes as the IT attack, and that Conditional Watermarks are less robust than other schemes.

False Positives A concern with approximate fingerprints is their potential to trigger on non-fingerprinted models, causing high false positive rates (FPR). However, we find the actual FPRs to be low: Word-in-response at 1%, Conditional Watermark at 2%, and M-out-of-N-words (with M=1, N=8) at 3% on OLMo-2-7B-Instruct without fingerprints.

Scalabilty In Fig. 2 (right), we find that the utility on tinyBenchmarks (Polo et al., 2024) does not drop much across fingerprint types at 256 fingerprints. Conditional Watermark is the most scalable (albeit less secure), while Word-in-response trades-off better security for lower scalability.

6. Conclusion

While model fingerprinting through exact memorization is emerging as a powerful primitive for controlling model ownership, we show that it is susceptible to inference time attacks such as changing the sampler. We hence propose a fingerprinting paradigm hinging on approximate memorization and detection, which can provide a layer of defense against such attacks. These could also act as a test-bed for studying benign memorization in LLMs. Designing more expressive schemes for such fingerprints, as well as attacks with better ASR-utility tradeoffs are other future directions.

References

Cheng, Z., Contente, E., Finch, B., Golev, O., Hayase, J., Miller, A., Moshrefi, N., Nasery, A., Nailwal, S., Oh, S., Tyagi, H., and Viswanath, P. OML: Open, monetizable, and loyal AI. 2024. URL https://eprint.iacr.org/2024/1573.

Cong, T., Ran, D., Liu, Z., He, X., Liu, J., Gong, Y., Li, Q., Wang, A., and Wang, X. Have you merged my model? on the robustness of large language model ip protection methods against model merging. In *Proceedings of the 1st ACM Workshop on Large AI Systems and Models with Privacy and Safety Analysis*, pp. 69–76, 2023.

Gu, C., Li, X. L., Liang, P., and Hashimoto, T. On the learnability of watermarks for language models, 2024. URL https://arxiv.org/abs/2312.04469.

Jiaxuan, W., Wanli, P., hang, F., Yiming, X., and juan, W. Imf: Implicit fingerprint for large language models, 2025. URL https://arxiv.org/abs/2503.21805.

Kirchenbauer, J., Geiping, J., Wen, Y., Katz, J., Miers, I., and Goldstein, T. A watermark for large language models. In *International Conference on Machine Learning*, pp. 17061–17084. PMLR, 2023.

Nasery, A., Hayase, J., Brooks, C., Sheng, P., Tyagi, H., Viswanath, P., and Oh, S. Scalable fingerprinting of large language models, 2025. URL https://arxiv.org/abs/2502.07760.

OLMo, T., Walsh, P., Soldaini, L., Groeneveld, D., Lo, K., Arora, S., Bhagia, A., Gu, Y., Huang, S., Jordan, M., Lambert, N., Schwenk, D., Tafjord, O., Anderson, T., Atkinson, D., Brahman, F., Clark, C., Dasigi, P., Dziri, N., Guerquin, M., Ivison, H., Koh, P. W., Liu, J., Malik, S., Merrill, W., Miranda, L. J. V., Morrison, J., Murray, T., Nam, C., Pyatkin, V., Rangapur, A., Schmitz, M., Skjonsberg, S., Wadden, D., Wilhelm, C., Wilson, M., Zettlemoyer, L., Farhadi, A., Smith, N. A., and Hajishirzi, H. 2 olmo 2 furious, 2025. URL https://arxiv.org/abs/2501.00656.

p-e w. Exclude top choices (xtc): A sampler that boosts creativity, breaks writing clichés, and inhibits non-verbatim repetition. https://github.com/oobabooga/text-generation-webui/pull/6335, September 2024. Pull Request #6335, oobabooga/text-generation-webui.

Polo, F. M., Weber, L., Choshen, L., Sun, Y., Xu, G., and Yurochkin, M. tinybenchmarks: evaluating llms with fewer examples. *arXiv preprint arXiv:2402.14992*, 2024.

Russinovich, M. and Salem, A. Hey, that's my model! introducing chain & hash, an llm fingerprinting technique, 2024. URL https://arxiv.org/abs/2407.10887.

Srivastava, A., Rastogi, A., Rao, A., Shoeb, A. A. M., Abid, A., Fisch, A., Brown, A. R., Santoro, A., Gupta, A., Garriga-Alonso, A., et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

Xu, J., Wang, F., Ma, M. D., Koh, P. W., Xiao, C., and Chen, M. Instructional fingerprinting of large language models, 2024. URL https://arxiv.org/abs/2401.12255.

Yamabe, S., Takahashi, T., Waseda, F., and Wataoka, K. Mergeprint: Robust fingerprinting against merging large language models, 2024. URL https://arxiv.org/abs/2410.08604.

Zhou, J., Lu, T., Mishra, S., Brahma, S., Basu, S., Luan, Y., Zhou, D., and Hou, L. Instruction-following evaluation for large language models, 2023. URL https://arxiv.org/abs/2311.07911.