ESF: Efficient Sensitive Fingerprinting for Black-Box Tamper Detection of Large Language Models

Xiaofan Bai[1†] **Pingyi Hu**[1†] **Xiaojing Ma**[1*] **Linchen Yu**[1]
Dongmei Zhang[2] **Qi Zhang**[2] **Bin Benjamin Zhu**[2*]
[1]Huazhong University of Science and Technology
[2]Microsoft Corporation
{xiaofanbai, pingyihu, linda, linchenyu}@hust.edu.cn,
{dongmeiz, qizhang, binzhu}@microsoft.com

Abstract

The rapid adoption of large language models (LLMs) in diverse applications has intensified concerns over their security and integrity, particularly in cloud environments where users cannot access internal model parameters. One critical threat is model tampering, which can compromise LLM behavior and reliability. However, traditional tamper detection methods, designed for deterministic classification models, are inadequate for LLMs due to their output randomness and massive parameter spaces. In this paper, we introduce *Efficient Sensitive Fingerprinting (ESF)*, the first fingerprinting method tailored for black-box tamper detection of LLMs. ESF generates fingerprint samples by optimizing output sensitivity at selected detection token positions and leverages *Randomness-Set Consistency Checking (RSCC)* to accommodate inherent output randomness. Additionally, we propose a novel *Max Coverage Strategy (MCS)* to select an optimal set of fingerprint samples that maximizes joint sensitivity to tampering. Grounded in a rigorous theoretical framework, ESF is computationally efficient and scalable to large models. Extensive experiments on state-of-the-art LLMs demonstrate that ESF reliably detects tampering—including fine-tuning, model compression, and backdoor injection—with detection rates of at least 99.2% using only 5 fingerprint samples, offering a robust solution for securing cloud-based AI systems.

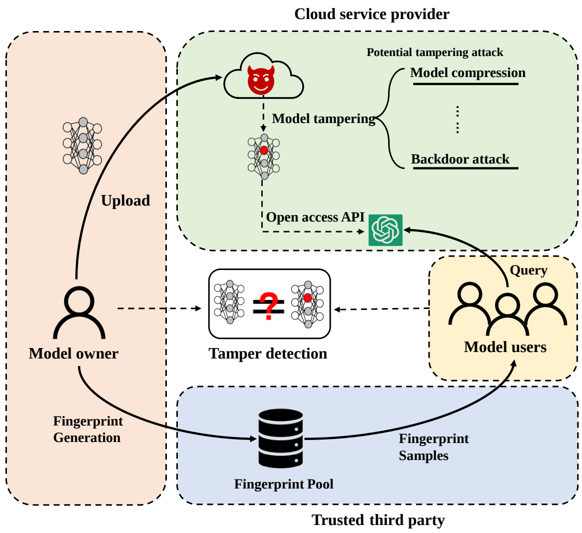


Figure 1: Illustration of tampering attacks on cloud-deployed LLMs and fingerprinting for tamper detection.

1 Introduction

Large Language Models (LLMs) are increasingly utilized in diverse fields, including code generation (Jiang et al., 2024), legal document analysis (Lai et al., 2024), medical diagnosis (Nazi and Peng, 2024), and decision-making (Li et al., 2022; Yang et al., 2024). These models are commonly deployed on third-party cloud platforms such as Amazon Web Services, Microsoft Azure, and Google Cloud, which raises significant concerns regarding their integrity and security. Adversaries may stealthily insert backdoors (Zhang et al., 2021), enabling the model to produce manipulated outputs in response to specific triggers. Furthermore, unscrupulous service providers may covertly employ model compression techniques (Xiao et al., 2023) to reduce operational costs (He et al., 2019), potentially compromising model reliability. To address these risks, it is essential to develop black-box methods for verifying the integrity of deployed LLMs through inconspicuous user queries.

Fingerprinting techniques (He et al., 2019; Xiaofan et al., 2024) offer a promising approach for tamper detection by querying a model as a normal user, as shown in Fig. 1. These methods generate fingerprint samples highly sensitive to model modifications and determine tampering based on the model's responses. However, existing fingerprinting techniques are designed for deterministic classification models, where the same input al-

*Corresponding Authors. † Equal Contribution.

Findings of the Association for Computational Linguistics: ACL 2025, pages 10477–10494
July 27 - August 1, 2025

ways produces the same output, making tamper detection straightforward—if the top-1 label differs from the expected output, the model is deemed tampered with. In contrast, LLMs introduce randomness through temperature-based sampling, rendering prior approaches ineffective since an unaltered model can produce different outputs for the same input. Moreover, LLMs have significantly larger parameter spaces than traditional models, making it more challenging to design fingerprint samples that effectively cover potential modifications across a vast number of parameters.

In this paper, we propose *Efficient Sensitive Fingerprinting (ESF)*, the first fingerprinting method tailored for black-box tamper detection of LLMs. ESF generates sensitive samples by optimizing output sensitivity at selected detection token positions, which can vary across samples, and selects the most sensitive samples as fingerprints for tamper detection. To address LLM output randomness, ESF employs *Randomness-Set Consistency Checking (RSCC)*: for each detection token position, it records all possible tokens that an unaltered model could generate in typical practical settings. The fingerprinting process is designed to maximize the likelihood that a tampered model will produce an out-of-set token at detection positions. During detection, if any generated token falls outside the expected set, the model is flagged as tampered; otherwise, it is considered unaltered. To enhance detection robustness, multiple fingerprint samples can be used for each decision, and in this setting, ESF introduces a novel *Max Coverage Strategy (MCS)* to select an optimal set of fingerprint samples, maximizing their joint sensitivity to improve tamper detection.

ESF is computationally efficient for large models, with its fingerprint generation and selection processes grounded in theoretical analysis. Our extensive experiments demonstrate that ESF is both effective and efficient for LLM tamper detection, achieving high detection rates and making it highly practical for real-world deployment.

Our contributions are summarized as follows:

- We present *Efficient Sensitive Fingerprinting (ESF)*, the first black-box tamper detection method for LLMs that is robust to inherent output randomness. ESF generates fingerprint samples by optimizing the sensitivity of selected detection token positions and maximizing their combined response to parameter modifications in the protected model.
- We provide a theoretical framework for optimizing fingerprint sensitivity while ensuring computational efficiency, enabling ESF to scale to large models.
- We introduce the *Max Coverage Strategy (MCS)*, a theoretically grounded method for selecting fingerprint samples to maximize detection coverage of altered parameters, enhancing tamper detection performance.
- Our comprehensive experiments demonstrate ESF's effectiveness across diverse tamper scenarios, achieving detection rates of at least 99.2% using only 5 fingerprint samples.

2 Related Work

Both watermarking and fingerprinting have been explored for black-box tamper detection of deep neural network (DNN) models. Watermarking methods (Yin et al., 2023) require modifying the model, whereas fingerprinting does not. Since our focus is on fingerprinting, we consider only approaches that do not modify the model.

Several fingerprinting techniques (He et al., 2019; Docena et al., 2021; Kuttichira et al., 2022; Wang et al., 2023; Xiaofan et al., 2024; Aramoon et al., 2021; He et al., 2024) have been proposed for black-box tamper detection in traditional classification models. These methods rely on the deterministic relationship between input and output at inference time: a model is considered altered if its predictions for specific fingerprint samples deviate from expected labels. However, LLMs introduce inherent randomness in their outputs due to decoding strategies such as top-K and top-P sampling, breaking this deterministic input-output relationship and rendering existing fingerprinting techniques ineffective. Moreover, the significantly larger parameter space of LLMs provides adversaries with more opportunities for undetectable modifications, necessitating a more effective and scalable fingerprinting approach.

Beyond tamper detection, fingerprinting has also been widely used for *intellectual property (IP) protection* (Jin et al., 2024; Cao et al., 2021; Wang and Chang, 2021; Lukas et al., 2021; Wang et al., 2021; Chen et al., 2021; Li et al., 2021; Zhao et al., 2020; Le Merrer et al., 2020; Yang et al., 2022; Pan et al., 2022; Cong et al., 2023; Li et al., 2024, 2023; Zhang and Koushanfar, 2024; Wang et al., 2024; Zhang et al., 2024; Pang et al., 2024), which aims to verify whether a given model is the protected model or derived from it, supporting claims of model ownership. In contrast, tamper detection determines whether a model has been altered. While extensive research has been conducted on fingerprinting for IP protection in LLMs, tamper detection for LLMs remains an unexplored area.

3 Threat Model

Following prior works, we adopt a white-box generation and black-box detection model, as illustrated in Fig. 1. In the fingerprint generation phase, we assume white-box access to the protected LLM, including its intermediate features. This is reasonable, as fingerprint samples are created by the model owner for *public tamper detection*. In the detection phase, we assume black-box access to the suspect model, where only API queries and text outputs are available.

We also assume public access to either a tokenizer or a securely encapsulated tokenizer API, enabling consistent tokenization across different users, such as researchers and developers. Many models, including GPT (Brown et al., 2020) and BERT (Devlin et al., 2019), have open-sourced their tokenizers. Also, the model owners can provide a securely encapsulated tokenizer API that maintains functional alignment with the original training tokenization during tamper detection while protecting implementation details.

Following prior work (Wang et al., 2023), we assume a trusted third party securely stores and distributes fingerprint samples for *public tamper detection*, while the cloud service provider may be untrustworthy and could tamper with the uploaded LLM. Additionally, adversaries may attempt to acquire or generate fingerprint samples to evade detection.

4 Theoretical Framework for Sensitive and Efficient Fingerprinting

In this section, we define fingerprint sensitivity and provide a theoretical framework for generating sensitive and efficient fingerprint samples.

4.1 Sensitivity of Fingerprint Samples

To reliably detect tampering in LLMs, it is crucial to identify effective fingerprint samples whose outputs are highly responsive to changes in the model's parameters. In this way, even minor modifications to the model can lead to noticeable differences in the model's responses to these fingerprint samples. Therefore, quantifying sensitivity of input samples is a foundational step in crafting effective fingerprint samples for tamper detection.

We formally define the sensitivity of a sample x with respect to a model $f(\cdot)$ parameterized by $W = \{W_1, \ldots, W_L\}$ as follows:

Definition 4.1 (Sample Sensitivity). Consider a sample x and a model $f(\cdot)$ with parameters $W = \{W_1, \ldots, W_L\}$.

Layer-wise Sensitivity. The sensitivity of the model output $f(W, x)$ to perturbations in the parameters W_i of layer i is measured by the Frobenius norm of the gradient:

$$S_i(x, W_i) = \left\| \frac{\partial f(W, x)}{\partial W_i} \right\|_F. \tag{1}$$

Network-wise Sensitivity. Aggregating across all layers, the overall sensitivity of the model to perturbations in all parameters W for the input x is defined as:

$$\begin{aligned} S(x, W) &= \left\| \frac{\partial f(W, x)}{\partial W} \right\|_F \\ &= \left(\sum_{i=1}^{L} \left\| \frac{\partial f(W, x)}{\partial W_i} \right\|_F^2 \right)^{1/2} \\ &= \left(\sum_{i=1}^{L} S_i(x, W_i)^2 \right)^{1/2}. \end{aligned} \tag{2}$$

Eq. 1 quantifies the sensitivity of the model output to perturbations in a single layer, while Eq. 2 captures the aggregate sensitivity to perturbations across the entire parameter set W.

4.2 From Overall to Single-Layer Sensitivity

Directly optimizing the overall network-wise sensitivity defined in Eq. 2 can be computationally prohibitive for LLMs due to their immense parameter space. To address this challenge, we provide a theoretical analysis that enables efficient sensitivity optimization by focusing on individual layers, thereby reducing computational overhead while preserving effectiveness.

For tractability, our analysis is based on the following assumptions:

1. The derivative of the activation function $\sigma(\cdot)$ is bounded, i.e., $0 < m_\sigma \leq \sigma'(z) \leq M_\sigma$ almost everywhere, where m_σ and M_σ are positive constants.
2. The weight matrix W_k of each layer satisfies a non-degeneracy condition: for any vector

v, $0 < m_{W,k}\|v\|_2 \le \|W_k v\|_2 \le M_{W,k}\|v\|_2$, with $m_{W,k} > 0$. This prevents local collapse (e.g., vanishing gradients) in the network.

3. The output of every block is fed into a (learned) affine LayerNorm with gain g_k and bias b_k. For all allowed inputs, the LayerNorm (LN) is L_N-Lipschitz and l_N-inverse-Lipschitz: for all vectors u, $l_N\|u\|_2 \le \|\mathrm{LN}_k(u)\|_2 \le L_N\|u\|_2$, where $0 < l_N \le L_N < \infty$ are constants independent of the input x.

Proposition 4.2 (Positive Correlation of Layer Sensitivities)**.** *Assume $f : \mathbb{R}^d \to \mathbb{R}^m$ is an L-layer network satisfying Assumptions 1-3. For any pair of indices $1 \le i < j \le L$, there exist strictly positive constants $C_{ij}^{\min}$ and $C_{ij}^{\max}$, depending only on $(m_\sigma, M_\sigma, m_W, M_W, l_N, L_N)$ and the gap $j-i$, such that for every input x,*

$$C_{ij}^{\min}\, S_j(x) \le S_i(x) \le C_{ij}^{\max}\, S_j(x). \tag{3}$$

Thus, increasing the sensitivity of one layer proportionally increases the sensitivity of any other layer, up to uniform constants.

The proof is provided in Appendix A.1.

Corollary 4.3 (Positive Correlation of Layer-wise and Network-wise Sensitivity)**.** *Under the same assumptions as Prop. 4.2, for a fingerprint sample x and an LLM $f(\cdot)$ with L layers, optimizing the sensitivity of a single layer will simultaneously enhance the overall network-wise sensitivity across all layers.*

The proof is provided in Appendix A.2.

Practical Considerations Regarding Assumptions. Assumption 1 may not strictly hold for all activation functions. For example, in SwiGLU, the SiLU component can have a negative derivative for very negative pre-activations. However, this "bad" region is rarely encountered in practice, as standard data distributions and weight initializations keep most neuron inputs within well-behaved ranges. As a result, the vast majority of neurons maintain positive, bounded gradients, and the few neurons that do enter the negative-derivative regime contribute negligibly to the layer-wise Jacobian products. Therefore, while Assumption 1 is not satisfied everywhere, it holds almost everywhere for SwiGLU networks.

Similarly, for Assumption 3, it is possible to construct pathological cases where the LayerNorm scale vector γ is driven to zero, violating the inverse-Lipschitz condition. However, such cases are measure-zero under realistic feature distributions and would catastrophically degrade model performance, making them easily detectable. In practice, with the standard choice $\varepsilon = 10^{-5}$ and γ initialized near 1, production LLMs (GPT, LLAMA, GEMMA, etc.) empirically satisfy $l_N \approx 1/\sqrt{d_{\text{model}}} > 0$ across billions of tokens. Thus, the bidirectional Lipschitz bounds in Assumption 3 also hold almost everywhere.

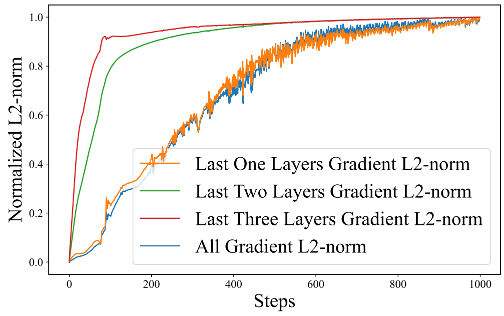


Figure 2: Experimental analysis of the correlation between sensitivities of different single layers and between single-layer and overall model sensitivities.

These considerations ensure that our theoretical assumptions are well met in practice, as further confirmed by our following experiments on SwiGLU-based models.

To validate Prop. 4.2 and Corollary 4.3, we conduct an experiment by maximizing $\left\|\frac{\partial f(x)}{\partial W_L}\right\|_2$ and recording the sensitivity of other layers on Qwen-2.5-0.5B (Team, 2024). As shown in Fig. 2, the results reveal a consistent trend between sensitivities of different single layers and between single-layer and overall model sensitivities, supporting both the proposition and the corollary.

4.3 From Gradient to Latent Output

Building on Prop. 4.2 and Corollary 4.3, it is sufficient to maximize the sample sensitivity with respect to a single layer, rather than across all layers. Nevertheless, directly optimizing Eq. 1 necessitates the computation of second-order derivatives during backpropagation when generating fingerprint samples, which is computationally inefficient. To address this limitation, we next present a theoretical analysis that enables further improvements in the efficiency of fingerprint sample generation.

Proposition 4.4 (Positive Correlation between Gra-

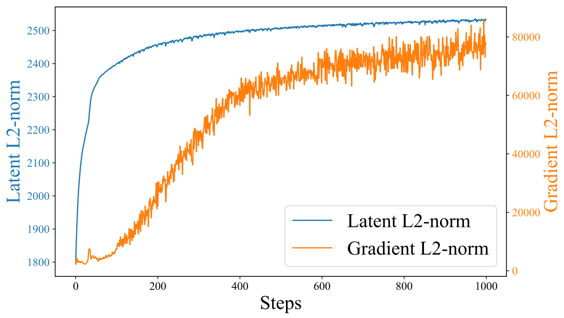


Figure 3: Experimental analysis of the relationship between $||\frac{\partial f_L(x)}{\partial W_L}||_2$ and $||f_{L-1}(x)||_2$ for an L-layers model.

dient Sensitivity and Latent Output). *Let $f_i(x)$ denote the output of the i-th layer ($0 \le i \le L$). The ℓ_2 norm of the gradient-defined sensitivity of the last (L-th) layer, $S = \left\| \frac{\partial f_L(x)}{\partial W_L} \right\|_F$, is positively correlated with the ℓ_2 norm of the penultimate layer's output, denoted by $\|f_{L-1}(x)\|_2$.*

The proof of Prop. 4.4 is provided in Appendix A.3.

Prop. 4.4 implies that, instead of directly maximizing the gradient-defined sensitivity—which requires computationally expensive second-order derivatives—one can efficiently maximize the ℓ_2 norm of the penultimate layer's output to achieve a similar effect. This approach substantially improves the efficiency of fingerprint sample generation, as it only involves first-order computations.

To empirically validate Prop. 4.4, we conduct an experiment in which we maximize $\|f_{L-1}(x)\|_2$ and record $\left\| \frac{\partial f_L(x)}{\partial W_L} \right\|_2$ on Qwen-2.5-0.5B. As shown in Fig. 3, the results demonstrate a consistent trend between $\|f_{L-1}(x)\|_2$ and $\left\| \frac{\partial f_L(x)}{\partial W_L} \right\|_2$, thereby supporting Prop. 4.4.

4.4 Sample Sensitivity and Tamper Detection

By Prop. 4.4, it is sufficient to evaluate the ℓ_2-norm of the penultimate hidden state to obtain the sample sensitivity $S(x, W)$. We now establish a connection between this sensitivity and the *expected* observable shift in the model's output distribution under small, random parameter tampering.

Perturbation Model. Consider a parameter vector $W \in \mathbb{R}^{d_W}$, and suppose an adversary adds an *isotropic* random perturbation $\Delta W \sim \mathcal{D}$ such that

$$\mathbb{E}[\Delta W] = 0, \quad \mathbb{E}[\Delta W \Delta W^\top] = \frac{\rho^2}{d_W} I_{d_W}, \quad (4)$$

where $\rho > 0$. Both a Gaussian distribution $\mathcal{N}(0, (\rho^2/d_W)I)$ and a vector chosen uniformly on the sphere of radius ρ satisfy Eq. 4.

Proposition 4.5 (Expected Top-K Shift and Sensitivity)**.** *Define $z = f(x; W)$, $z' = f(x; W+\Delta W)$ and let $p = \mathrm{softmax}(z)$, $p' = \mathrm{softmax}(z')$. Let $T_K(\cdot)$ be the normalized top–K truncation of a probability vector and let $d_{TK} : \Delta^{K-1} \times \Delta^{K-1} \to \mathbb{R}_{\ge 0}$ be any distance metric which is inverse-Lipschitz (e.g., the total-variation distance). Under the isotropic perturbation model in Eq. 4,*

$$\mathbb{E}_{\Delta W}\big[d_{TK}\big(T_K(p), T_K(p')\big)\big] \ \ge\ C_{\mathrm{tot}} S(x, W), \quad (5)$$

where $C_{\mathrm{tot}} = \frac{\rho}{\sqrt{d_W}} C_{\mathrm{sm}}\, C_{TK} > 0$ is the inverse-Lipschitz constant of the softmax on the zero-sum subspace, and $C_{TK} > 0$ is an average inverse-Lipschitz constant of the top-K mapping operator.

The proof of Prop. 4.5 is provided in Appendix A.4.

This result demonstrates that the expected shift in the LLM's output distribution is lower-bounded by the sample sensitivity $S(x, W)$. Therefore, maximizing sample sensitivity directly increases the expected detectability of model tampering in the output distribution. This establishes sample sensitivity as an effective and efficient signal for tamper detection in practice.

5 ESF: Efficient Sensitive Fingerprinting

We present the design of *Efficient Sensitive Fingerprinting (ESF)*, which enables robust and efficient tamper detection for LLMs. ESF consists of three key components: fingerprint sample generation, Randomness-Set Consistency Checking (RSCC), and the Max Coverage Strategy (MCS).

5.1 Fingerprint Sample Generation

Based on the theoretical framework described in Section 4, we efficiently maximize the sensitivity of an input x for an LLM $f(\cdot)$ by maximizing $||f_{L-1}(x)||_2$ at each detection token position. Our optimization objective is:

$$\max_{\Delta x} \mathcal{S}(x) = ||f_{L-1}(x + \Delta x)||_2 \quad (6)$$

We solve Eq. 6 using iterative *gradient ascent*:

$$x_{i+1} = x_i + \eta \cdot \nabla_{x_i} \mathcal{S}(x_i) \quad (7)$$

where η denotes the learning rate. This optimization generates a set of sensitive samples, from which we select the most sensitive ones to form a set of fingerprint samples, highly responsive to weight changes caused by model tampering.

5.2 Randomness-Set Consistency Checking

To address the inherent randomness in LLM outputs, we introduce *Randomness-Set Consistency Checking (RSCC)*. During fingerprint generation for a sample x, we collect the top-K and top-P token candidates at each detection token position, forming the fingerprint label set $\mathcal{Y}$ for that position. This set represents the range of tokens the protected model is likely to generate at the detection token position under typical inference settings.

During detection, when x is used to query the suspect model, the returned text is tokenized. If the predicted token y_{pred} at any detection position does not belong to the corresponding label set $\mathcal{Y}$ ($y_{\text{pred}} \notin \mathcal{Y}$), the model is considered tampered. When multiple fingerprint samples are used in a single query, the model is flagged as tampered if any sample detects tampering; otherwise, it is considered unaltered.

5.3 Max Coverage Strategy

When probing a model with N fingerprint samples $\mathcal{X} = \{x^{(1)}, \dots, x^{(N)}\}$, our objective is to maximize their *joint sensitivity*:

$$S_{\max}(\mathcal{X}, W) = \max_{x \in \mathcal{X}} S(x, W), \tag{8}$$

where $S(x, W)$ is the sensitivity for a single input x. A large $S_{\max}$ ensures that at least one fingerprint in $\mathcal{X}$ is highly responsive to small perturbations.

Empirical Premise. LayerNorm (or RMSNorm) makes penultimate-layer embeddings in modern LLMs approximately *zero-mean, isotropic, and sub-Gaussian* (Cai et al., 2021). Thus, for $e = f_{L-1}(x) \in \mathbb{R}^{d_{L-1}}$, we assume

$$\mathbb{E}[e] = 0, \quad \mathrm{Cov}(e) = \sigma^2 I, \quad \|e\|_{\psi_2} \le \kappa\sigma, \tag{9}$$

where $\|\cdot\|_{\psi_2}$ is the sub-Gaussian (Orlicz) norm, defined for a random vector v as

$$\|v\|_{\psi_2} = \sup_{\|u\|_2 = 1} \sup_{p \ge 1} p^{-1/2} \big(\mathbb{E}|\langle u, v\rangle|^p\big)^{1/p},$$

with $\sigma = \Theta(1)$ and $d_{L-1} = O(10^{3\text{–}4})$ for modern LLMs such as those in Section 6.

Proposition 5.1 (High-probability Max Coverage)**.** *Let $x^{(1)}, \dots, x^{(N)}$ be i.i.d. samples from a distribution satisfying Eq. 9, and let $e_n = f_{L-1}(x^{(n)})$. Define the pair-wise distance*

$$D = \sum_{1 \le m < n \le N} \|e_m - e_n\|_2^2. \tag{10}$$

For any $0 < \delta < 1$, set $s_{N,\delta} = \sigma\left(\sqrt{d_{L-1}} + \sqrt{2\log\frac{2N}{\delta}}\right)$. There exist universal constants $c > 0$ such that

$$\Pr\left[S_{\max} \ge \sqrt{d_L}\, s_{N,\delta} \ \wedge\ D \ge c\,\sigma^2 N^2 d_{L-1}\right] \ge 1 - \delta. \tag{11}$$

On this event, the joint detection radius $r_{\min} = \tau/(\eta S_{\max})$ satisfies

$$r_{\min} \le \frac{\tau}{\eta\sqrt{d_L}\, s_{N,\delta}} = O\left(\frac{\tau}{\eta\sigma\sqrt{d_L d_{L-1}}}\right). \tag{12}$$

The proof is given in Appendix A.5. Prop. 5.1 guarantees that, with probability at least $1 - \delta$, a batch of N i.i.d. fingerprint samples will contain at least one fingerprint sample with sensitivity $\Theta(\sqrt{d_L d_{L-1}})$, resulting in a detection radius of $r_{\min} = O\big(\tau/(\eta\sigma\sqrt{d_L d_{L-1}})\big)$. Moreover, maximizing the pairwise-distance statistic D increases the likelihood that $D \ge c\,\sigma^2 N^2 d_{L-1}$, and thus improves the probability that the high-sensitivity event in Eq. 11 holds.

Based on this, we introduce the *Max Coverage Strategy (MCS)*, which selects fingerprint samples to maximize joint sensitivity by maximizing their pairwise distances in the embedding space.

To select N fingerprint samples, we first compute the penultimate-layer embeddings for all candidates in the fingerprint pool S using the pretrained LLM. For each sample $s_i \in S$, let $e_i = \mathcal{M}(s_i)$ denote its embedding. The first sample is chosen at random. Then, at each step, we select the sample s^* that maximizes the minimum distance to all previously selected samples:

$$s^* = \arg\max_{s \in S \setminus S^*} \min_{s' \in S^*} d(e_s, e_{s'}), \tag{13}$$

where $d(e_s, e_{s'})$ is the Euclidean distance in the embedding space. This process is repeated until N samples are chosen, ensuring maximal coverage and detection capability.

In practice, if the number of fingerprint samples per probe is fixed, the model owner can pre-group samples using MCS, assigning each group to a single probe. If the number is not predetermined, the model owner can precompute a pairwise distance table and store it with a trusted third party. When a user requests a specific number of fingerprint samples, the trusted third party applies MCS using the distance table to select the optimal set, thus maximizing joint sensitivity, and provides the selected samples to the user.

6 Experimental Results

6.1 Experimental Setup

To comprehensively evaluate ESF, we assess its detection performance on five widely used state-of-the-art (SOTA) models: **Meta LLaMA-3-8B** and **LLaMA-3.2-1B** from the LLaMA 3 (AI@Meta, 2024) family, **Qwen-2.5-0.5B** and **Qwen-2.5-7B** from the Qwen 2.5 (Team, 2024) family, and **Mistral 7B** (Jiang et al., 2023). We evaluate ESF under three model tampering scenarios: fine-tuning, backdoor injection (Xu et al., 2024), and model compression (Dettmers et al., 2022).

6.1.1 Settings for Fingerprint Generation and Tamper Detection

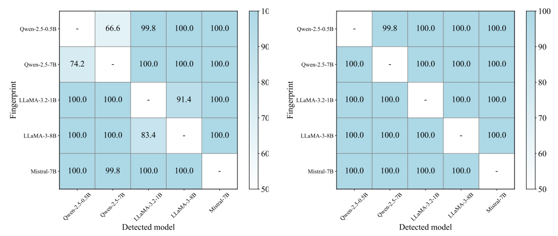
For each original model $f(x)$, we generate 2,500 sensitive samples with randomly selected source inputs and select the 500 most sensitive to form a fingerprint pool. For each fingerprint sample x, we record the top-$K = 50$ token candidates of the detection output token as its randomness set, representing the maximal randomness level commonly used in practice. Let N_s denote the number of fingerprint samples used per probe. For tamper detection, when $N_s = 1$, we randomly select a fingerprint sample from the pool to query the suspect model and report the average over 500 samples. When $N_s > 1$, we randomly select one fingerprint sample and use MCS to select the remaining $N_s - 1$ samples; this process is repeated 500 times and the results are averaged.

By default, our experiments focus on the first output token position for tamper detection. As shown in Appendix B.1, considering only the first predicted token is sufficient for effective detection. Nevertheless, in practical scenarios, incorporating multiple output token positions per fingerprint sample can further improve detection performance.

6.1.2 Model Tampering Types

In our experiments, we assess the performance of ESF under three popular model tampering types: model fine-tuning, backdoor injection, and model compression.

Specifically, for model fine-tuning and backdoor injection, we examine the impact of three parameter update strategies: **Full Parameter Fine-Tuning**: All model parameters are updated. **Last 3 Layers Fine-Tuning**: Only the final three transformer layers are trainable. **LoRA Fine-Tuning (Hu et al., 2021)**: A parameter-efficient approach that introduces low-rank matrices to approximate weight updates, leaving the base model weights unchanged.



(a) $N_S = 1$ (b) $N_S = 5$

Figure 4: Model replacement-based tamper detection rate (%) of ESF. The vertical axis represents the model utilized for generating fingerprints, while the horizontal axis denotes the substitute model.

For **model fine-tuning**, we use 5,000 clean samples from the Alpaca (Taori et al., 2023) instruction-tuning dataset. For the **backdoor injection**, we use 5,000 Alpaca-derived samples, 10% of which (500 samples) are embedded with a predefined trigger that forces the model to produce a specific incorrect output during inference (Xu et al., 2024). All fine-tuning is performed using LLaMA-Factory (Zheng et al., 2024) for 1 epoch with batch size 1 and a learning rate of 10^{-5}.

For **model compression**, we apply int4 and int8 quantization using bitsandbytes (Dettmers et al., 2022). For each type of tampering, we independently train 10 tampered models from the original.

6.2 Tamper Detection Performance

Table 1 shows ESF's effectiveness across models and tampering types. With $N_s = 1$, ESF achieves a minimum detection rate of 43.4%. As N_s increases to 5, detection rates reach at least 99.2% across all tampering types, highlighting the high sensitivity of ESF-generated fingerprint samples.

6.3 Replacement-Based Model Tampering

A dishonest cloud provider might replace the uploaded LLM with another existing model to reduce deployment costs. We evaluate ESF's performance in detecting such replacements, as shown in Fig. 4. ESF demonstrates high sensitivity to cross-architecture substitutions, achieving over 99.8% detection with a single fingerprint. For intra-architecture substitutions within the same family, detection rates exceed 66.6% for $N_s = 1$ and 99.8% for $N_s = 5$.

Table 1: Tamper detection rate (%) with N_S fingerprint samples

Models		Qwen-2.5-0.5B		Qwen-2.5-7B		LLaMA-3.2-1B		LLaMA-3-8B		Mistral-7B	
Tempering type \ N_S		1	5	1	5	1	5	1	5	1	5
Clean data fine-tuning	Full parameter	95.2	100	97.8	100	98.2	100	100	100	72.0	100
	Lora	69.4	100	56.2	100	65.6	99.8	80.0	100	64.2	100
	Last 3 layers	45.6	100	71.0	100	81.4	100	95.2	100	55.4	100
Backdoor Injection fine-tuning	Full parameter	92.2	100	98.6	100	98.6	100	100	100	76.4	100
	Lora	75.0	100	80.2	100	66.0	99.8	86.6	100	62.8	100
	Last 3 layers	47.6	99.6	65.6	100	82.4	100	95.8	100	65.0	100
Model compression	int4	81.2	100	73.0	100	70.8	100	71.2	100	55.6	99.6
	int8	63.2	100	43.4	99.4	66.2	100	53.6	100	49.6	99.2

Table 2: Tamper detection rate (%) under varying randomness levels. The tampering method is backdoor injection via fine-tuning Qwen-2.5-0.5B.

Tempering type	Setting 1		Setting 2		Setting 3		Setting 4	
/ N_S	1	5	1	5	1	5	1	5
Full Parameter	99.4	100	95.2	100	91.4	100	80.2	100
LoRA	98.8	100	92.4	100	70.4	99.8	62.2	99.0
Last 3 Layers	98.2	100	90.0	100	59.8	98.4	47.8	96.8
False Positive Rate	0.0	0.0	0.0	0.0	0.0	0.0	11.2	2.8

6.4 Tamper Detection under Mismatched Randomness Levels

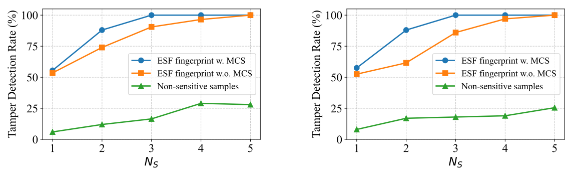
We evaluate ESF's robustness under four different inference-time randomness configurations, while RSCC recording is fixed at top-K = 50. The inference settings are as follows: Setting 1: top-P = 70%, top-K = 10; Setting 2: top-P = 80%, top-K = 20; Setting 3: top-P = 92%, top-K = 50; Setting 4: top-P = 95%, top-K = 100.

As shown in Table 2, ESF maintains robust detection performance across all configurations. Notably, even in Setting 4, where inference randomness exceeds that of the recording phase, ESF achieves a 96.8% detection rate for $N_s = 5$, demonstrating strong resilience to randomness-level mismatch. Additional details are provided in Appendix B.2. The first three settings yield no false positives since *RSCC* records the top-50 tokens, while Setting 4 results in a small false positive rate of about 2.8% for $N_s = 5$.

6.5 Ablation Study

To evaluate whether *MCS* improves detection rates when using multiple fingerprint samples, we compare it to a random sampling strategy (*ESF without MCS*) and to natural, non-sensitive inputs (*Non-sensitive samples*). Figure 5 presents the results of fine-tuning the last three layers of Qwen2.5-0.5B with both clean and poisoned data. Additional ablation results using other models and tampering types are provided in Appendix B.3.

The results demonstrate that *MCS* significantly enhances multi-sample detection performance. For instance, with $N_s = 2$ in detecting poison data fine-tuning on the last 3 layers, the detection rate increases from 61.6% (without MCS) to 88.2% (with MCS), enabling ESF to reach near-perfect detection more rapidly.



(a) Clean data fine-tuning the last 3 layers.

(b) Backdoor injection by fine-tuning the last 3 layers.

Figure 5: Tamper detection rate (%) under different fingerprint selection strategies for different tampering types on Qwen2.5-0.5B.

6.6 Robustness to Adaptive Attacks

6.6.1 Fingerprint Leakage

Adversaries may attempt to bypass ESF by leveraging previously collected fingerprint samples or generating new ones for adaptive attacks, aiming to manipulate the model while keeping the top-K token candidates of the fingerprint samples unchanged. To evaluate ESF's robustness against such attacks, we randomly split the fingerprint pool into two equal subsets: one for launching adaptive attacks (*leaked fingerprints*) and the other for tamper detection (*unleaked fingerprints*). We conduct adaptive attacks using clean-data fine-tuning on the last 3 layers of CIFAR10, as shown in Fig. 6. As training progresses, the detection rate for $N_s = 1$ with *leaked fingerprints* drops to 0%, while the detection rate for *unleaked fingerprints* remains

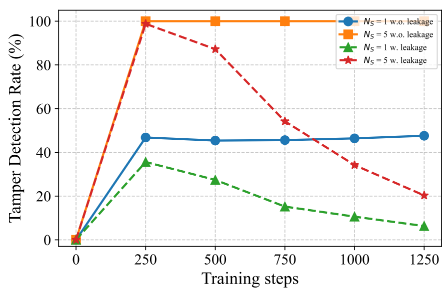


Figure 6: Tamper detection rates using leaked fingerprint samples in adaptive backdoor injection via fine-tuning the last 3 layers of Qwen-2.5-0.5B against ESF.

relatively high, with only a slight decrease. This robustness is attributed to the diversity of ESF's source samples and the use of *MCS*.

6.6.2 Anomaly Detection

Adversaries may attempt to bypass tamper detection by using anomaly detection to identify fingerprint samples and return unaltered outputs for them. Since our fingerprint generation process does not initially constrain distortions, optimized fingerprints can be easily distinguished from normal inputs. For example, 20-token segments randomly sampled from Alpaca have an average perplexity of 20.29 (Qwen-2.5-0.5B), whereas optimized fingerprints reach a perplexity of 5,578,468.04.

To address this, we constrain fingerprint generation with a semantic consistency loss (Xu and Wang, 2024), ensuring fingerprints remain natural while retaining high detection rates. Specifically, we initialize each fingerprint with a 20-token segment randomly selected from the Alpaca dataset and optimize it as follows:

$$\max_{\Delta x} \mathcal{S}(x) = \|f_{L-1}(x + \Delta x)\|_2 + \alpha\, sim(x, x + \Delta x) \quad (14)$$

where $sim(\cdot)$ is cosine similarity and $\alpha = 0.05$. This constraint yields detection rates of 27.6% for $N_S = 1$ and 93.4% for $N_S = 5$, with average perplexity reduced to 65.79.

This issue can be further mitigated by embedding the perturbed tokens within longer samples. For example, optimizing 20 tokens within a 100-token context (arranged in four evenly spaced groups) reduces average perplexity to 42.67 while maintaining comparably high detection rates.

6.7 Computational Cost

We compare the efficiency of ESF with all-layer gradient sensitivity optimization for fingerprint generation, using dual NVIDIA H20 GPUs. Table 3 summarizes the results. ESF demonstrates substantial resource efficiency, reducing GPU memory consumption by 37.32% and generation time by 71.50% on average compared to all-layer gradient sensitivity optimization. These findings highlight ESF's superior efficiency in fingerprint generation, enabling scalable deployment across models of varying computational scales and making ESF a practical solution for real-world applications that require rapid, resource-constrained tamper detection.

Table 3: Computational cost across different models for generating a single fingerprint sample.

Models	Fingerprint Opimized with Full Layer's Gradient		Fingerprint Opimized with Penultimate Layer's output	
	Time (s)	Memory (GB)	Time (s)	Memory (GB)
Qwen-2.5-0.5B	84.27	10.53	23.77	6.22
Qwen-2.5-7B	508.67	100.85	147.22	63.49
LLaMA-3.2-1B	206.68	22.27	50.34	13.90
LLaMA-3-8B	1016.78	104.98	299.49	66.44
Mistral-7B	322.08	84.64	103.04	55.60

Moreover, we also conduct an experiment to demonstrate the effectiveness of ESF compared to full layers' gradient sensitivity defined in Eq. 2. The results in Appendix B.4 demonstrate that ESF achieves a comparable performance to full layers' gradient sensitivity when $N_s = 5$. However, as previous results showed, ESF is much more efficient than full layers' gradient sensitivity.

7 Conclusion

We present *Efficient Sensitive Fingerprinting (ESF)*, the first tamper detection method specifically tailored for large language models (LLMs) deployed in cloud environments. By optimizing fingerprint sensitivity and leveraging both the *Max Coverage Strategy (MCS)* and *Randomness-Set Consistency Checking (RSCC)*, ESF effectively addresses the challenges of LLM output randomness while significantly reducing computational overhead. Our theoretical analysis establishes the soundness and efficiency of ESF, while extensive empirical evaluations show that ESF consistently achieves detection rates of at least 99.2% across diverse tampering scenarios using only five fingerprint samples, demonstrating its robustness and practical applicability. Overall, ESF not only advances black-box integrity verification for LLMs but also establishes a strong foundation for future research on secure and efficient tamper detection in cloud-based AI systems.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (Grant No. 62272175), the Major Research Plan of Hubei Province (Grant/Award No. 2023BAA027), the Key Research & Development Plan of Hubei Province of China (Grant No. 2024BAB049), and the project of Science, Technology and Innovation Commission of Shenzhen Municipality of China (Grant No. GJHZ20240218114659027).

Limitation

While ESF demonstrates strong effectiveness and efficiency in detecting model tampering, the fingerprint samples it generates are not yet fully natural. Although we discuss potential strategies to address this issue in Sec. 6.6.2, there remains significant room for improvement in balancing the naturalness and sensitivity of fingerprint samples. We leave this as future work.

It is worth noting, however, that most existing cloud service platforms focus on detecting harmful or privacy-compromising outputs rather than enforcing constraints on the naturalness of user inputs. Given this practical context, we argue that ESF remains a valuable and viable solution for detecting LLM tampering in black-box settings, providing an essential layer of integrity verification that supports the broader goals of security and trustworthiness in AI deployment.

References

AI@Meta. 2024. Llama 3 model card.

Omid Aramoon, Pin-Yu Chen, and Gang Qu. 2021. Aid: Attesting the integrity of deep neural networks. In *Proceedings of 2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 19–24.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Xingyu Cai, Jiaji Huang, Yuchen Bian, and Kenneth Church. 2021. Isotropy in the contextual embedding space: Clusters and manifolds. In *International Conference on Learning Representations*.

Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. 2021. Ipguard: Protecting intellectual property of deep neural networks via fingerprinting the classification boundary. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, pages 14–25.

Yufei Chen, Chao Shen, Cong Wang, and Yang Zhang. 2021. Teacher model fingerprinting attacks against transfer learning. *arXiv preprint arXiv:2106.12478*.

Tianshuo Cong, Delong Ran, Zesen Liu, Xinlei He, Jinyuan Liu, Yichen Gong, Qi Li, Anyu Wang, and Xiaoyun Wang. 2023. 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*, pages 69–76.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems*, 35:30318–30332.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Amel Nestor Docena, Thomas Wahl, Trevor Pearce, and Yunsi Fei. 2021. Sensitive samples revisited: Detecting neural network attacks using constraint solvers. *arXiv preprint arXiv:2109.03966*.

Chaoxiang He, Xiaofan Bai, Xiaojing Ma, Bin Benjamin Zhu, Pingyi Hu, Jiayun Fu, Hai Jin, and Dongmei Zhang. 2024. Towards stricter black-box integrity verification of deep neural network models. In *ACM Multimedia 2024*.

Zecheng He, Tianwei Zhang, and Ruby Lee. 2019. Sensitive-sample fingerprinting of deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4729–4737.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim. 2024. A survey on large language models for code generation. *arXiv preprint arXiv:2406.00515*.

Heng Jin, Chaoyu Zhang, Shanghao Shi, Wenjing Lou, and Y Thomas Hou. 2024. Proflingo: A fingerprinting-based intellectual property protection scheme for large language models. In *2024 IEEE Conference on Communications and Network Security (CNS)*, pages 1–9. IEEE.

Deepthi Praveenlal Kuttichira, Sunil Gupta, Dang Nguyen, Santu Rana, and Svetha Venkatesh. 2022. Verification of integrity of deployed deep learning models using bayesian optimization. *Knowledge-Based Systems*, 241:108238.

Jinqi Lai, Wensheng Gan, Jiayang Wu, Zhenlian Qi, and S Yu Philip. 2024. Large language models in law: A survey. *AI Open*.

Erwan Le Merrer, Patrick Perez, and Gilles Trédan. 2020. Adversarial frontier stitching for remote neural network watermarking. *Neural Computing and Applications*, 32(13):9233–9244.

Shen Li, Liuyi Yao, Jinyang Gao, Lan Zhang, and Yaliang Li. 2024. Double-i watermark: Protecting model copyright for llm fine-tuning. *arXiv preprint arXiv:2402.14883*.

Shuang Li, Xavier Puig, Chris Paxton, Yilun Du, Clinton Wang, Linxi Fan, Tao Chen, De-An Huang, Ekin Akyürek, Anima Anandkumar, Jacob Andreas, Igor Mordatch, Antonio Torralba, and Yuke Zhu. 2022. Pre-trained language models for interactive decision-making. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Yuanchun Li, Ziqi Zhang, Bingyan Liu, Ziyue Yang, and Yunxin Liu. 2021. Modeldiff: testing-based DNN similarity comparison for model reuse detection. In *Proceedings of 30th ACM SIGSOFT International Symposium on Software Testing and Analysis, ISSTA'21*, pages 139–151.

Zongjie Li, Chaozheng Wang, Shuai Wang, and Cuiyun Gao. 2023. Protecting intellectual property of large language model-based code generation apis via watermarks. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 2336–2350.

Nils Lukas, Yuxuan Zhang, and Florian Kerschbaum. 2021. Deep neural network fingerprinting by conferrable adversarial examples. In *Proceedings of 9th International Conference on Learning Representations, ICLR 2021*.

Zabir Al Nazi and Wei Peng. 2024. Large language models in healthcare and medical domain: A review. In *Informatics*, volume 11, page 57. MDPI.

Xudong Pan, Yifan Yan, Mi Zhang, and Min Yang. 2022. Metav: A meta-verifier approach to task-agnostic model fingerprinting. In *Proceedings of 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, pages 1327–1336. ACM.

Qi Pang, Shengyuan Hu, Wenting Zheng, and Virginia Smith. 2024. No free lunch in llm watermarking: Trade-offs in watermarking design choices. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

M.J. Wainwright. 2019. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press.

Shuo Wang, Sharif Abuadbba, Sidharth Agarwal, Kristen Moore, Ruoxi Sun, Minhui Xue, Surya Nepal, Seyit Camtepe, and Salil Kanhere. 2023. Publiccheck: Public integrity verification for services of run-time deep models. In *Proceedings of 2023 IEEE Symposium on Security and Privacy (SP)*, pages 1348–1365. IEEE.

Si Wang and Chip-Hong Chang. 2021. Fingerprinting deep neural networks-a deepfool approach. In *Proceedings of 2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE.

Siyue Wang, Xiao Wang, Pin-Yu Chen, Pu Zhao, and Xue Lin. 2021. Characteristic examples: High-robustness, low-transferability fingerprinting of neural networks. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI*, pages 575–582.

Zongqi Wang, Baoyuan Wu, Jingyuan Deng, and Yujiu Yang. 2024. Espew: Robust copyright protection for llm-based eaas via embedding-specific watermark. *arXiv preprint arXiv:2410.17552*.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. Smoothquant: accurate and efficient post-training quantization for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Bai Xiaofan, Chaoxiang He, Xiaojing Ma, Bin Benjamin Zhu, and Hai Jin. 2024. Intersecting-boundary-sensitive fingerprinting for tampering detection of DNN models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 54402–54413. PMLR.

Jiashu Xu, Mingyu Ma, Fei Wang, Chaowei Xiao, and Muhao Chen. 2024. Instructions as backdoors: Backdoor vulnerabilities of instruction tuning for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3111–3126, Mexico City, Mexico. Association for Computational Linguistics.

Yue Xu and Wenjie Wang. 2024. LinkPrompt: Natural and universal adversarial attacks on prompt-based language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6473–6486, Mexico City, Mexico. Association for Computational Linguistics.

Kang Yang, Run Wang, and Lina Wang. 2022. Metafinger: Fingerprinting the deep neural networks with meta-training. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 776–782. ijcai.org.

Zhenjie Yang, Xiaosong Jia, Hongyang Li, and Junchi Yan. 2024. Llm4drive: A survey of large language models for autonomous driving. *Preprint*, arXiv:2311.01043.

Zhaoxia Yin, Heng Yin, Hang Su, Xinpeng Zhang, and Zhenzhe Gao. 2023. Decision-based iterative fragile watermarking for model integrity verification. *arXiv preprint arXiv:2305.09684*.

Ruisi Zhang, Shehzeen Samarah Hussain, Paarth Neekhara, and Farinaz Koushanfar. 2024. {REMARK-LLM}: A robust and efficient watermarking framework for generative large language models. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 1813–1830.

Ruisi Zhang and Farinaz Koushanfar. 2024. Emmark: Robust watermarks for ip protection of embedded quantized large language models. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*, pages 1–6.

Xinyang Zhang, Zheng Zhang, Shouling Ji, and Ting Wang. 2021. Trojaning language models for fun and profit. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 179–197.

Jingjing Zhao, Qingyue Hu, Gaoyang Liu, Xiaoqiang Ma, Fei Chen, and Mohammad Mehedi Hassan. 2020. Afa: Adversarial fingerprinting authentication for deep neural networks. *Computer Communications*, 150:488–497.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

A Proofs of Propositions and Corollary

A.1 Proof of Prop. 4.2

Proof. We start by recalling that each layer performs a linear map followed by an element-wise non-linearity and a normalization. Writing $z_k = W_k a_{k-1}$ and $a_k = \mathrm{LN}_k\big(\sigma(z_k)\big)$ with $a_0 = x$, the Jacobian of the composite map $L_k = \mathrm{LN}_k \circ \sigma \circ W_k$ at the point a_{k-1} factors exactly as

$$J_k = \big[\mathrm{Jac}(\mathrm{LN}_k)\big]_{\sigma(z_k)} \,\mathrm{diag}\big(\sigma'(z_k)\big)\, W_k. \quad (15)$$

Because assumption 1 bounds the activation slope, the singular values of W_k, and the Lipschitz constants of the normalization, every J_k is simultaneously lower- and upper-bounded:

$$\begin{aligned} m = m_\sigma m_W l_N \;\le\; \sigma_{\min}(J_k) \;\le\; \|J_k\|_{op} \\ \le M_\sigma M_W L_N = M, \end{aligned} \quad (16)$$

where $||\cdot||_{op}$ denotes the spectral norm and $\sigma_{\min}(\cdot)$ is the smallest singular value. Crucially, the same m, M work for all layers and every input.

To relate two different layers, we consider the product of Jacobians sandwiched between them. Denoting $P_{i\to j-1} = \prod_{k=i+1}^{j-1} J_k$, classical sub-multiplicativity of singular values gives

$$\begin{aligned} m^{\,j-i-1} \;\le\; \sigma_{\min}(P_{i\to j-1}) \\ \le\; \|P_{i\to j-1}\|_{op} \le M^{\,j-i-1}. \end{aligned} \quad (17)$$

The next ingredient is the local gradient $\frac{\partial a_k}{\partial W_k}$. It can be written

$$\frac{\partial a_k}{\partial W_k} = \big[\mathrm{Jac}(\mathrm{LN}_k)\big]_{\sigma(z_k)} \big(\sigma'(z_k) \otimes a_{k-1}^{\top}\big), \quad (18)$$

that is, a normalization Jacobian multiplying the outer product of the element-wise derivative and the previous activation. Bounding each factor separately and using $\|u \otimes v^{\top}\|_F = \|u\|_2 \|v\|_2$ shows that every such outer product has a Frobenius norm lying between the lower bound

$$m_J = m_\sigma l_N \quad (19)$$

and the upper bound

$$M_J = M_\sigma L_N, \quad (20)$$

independently of x.

With these uniform constants established, we express the layer-i gradient of the whole network through the gradient of a later layer. Using the chain rule once more,

$$\begin{aligned} \frac{\partial f}{\partial W_i} &= P_{i\to j-1} \left(\tfrac{\partial a_i}{\partial W_i}\right) P_{j\to L-1}, \\ \frac{\partial f}{\partial W_j} &= P_{j\to L-1} \left(\tfrac{\partial a_j}{\partial W_j}\right). \end{aligned} \quad (21)$$

The Frobenius norm of a product admits the two-sided estimate

$$\sigma_{\min}(A)\, \|B\|_F \;\le\; \|AB\|_F \;\le\; \|A\|_{op}\, \|B\|_F, \quad (22)$$

valid for any conforming matrices A, B. Applying Eq. 22 to the leftmost factor $P_{i\to j-1}$ inside Eq. 21 and then inserting Eq. 17 yields

$$\begin{aligned} m^{j-i-1} \left\| \tfrac{\partial a_i}{\partial W_i} P_{j\to L-1} \right\|_F \;\le\; S_i(x) \\ \le\; M^{j-i-1} \left\| \tfrac{\partial a_i}{\partial W_i} P_{j\to L-1} \right\|_F. \end{aligned} \quad (23)$$

For the rightmost factor $P_{j\to L-1}$, a second use of Eq. 22 converts the mixed norm in Eq. 23 into a quantity directly comparable with $S_j(x)$. Indeed,

$$\begin{aligned} m_J\, m^{L-j}\, S_j(x) \;\le\; \left\| \tfrac{\partial a_i}{\partial W_i} P_{j\to L-1} \right\|_F \\ \le\; M_J\, M^{L-j}\, S_j(x), \end{aligned} \quad (24)$$

where the constants m_J, M_J come from Eq. 19 and Eq. 20. Substituting Eq. 24 into Eq. 23 and grouping the factors of m and M finally produces

$$m^{L-i-1} \frac{m_J}{M_J} S_j(x) \;\le\; S_i(x) \;\le\; M^{L-i-1} \frac{M_J}{m_J} S_j(x). \quad (25)$$

Thus Prop. 4.2 holds with

$$C_{ij}^{\min} = m^{L-i-1} \frac{m_J}{M_J}, C_{ij}^{\max} = M^{L-i-1} \frac{M_J}{m_J}. \quad (26)$$

Both constants are strictly positive, depend only on the universal hyperparameters and the distance $j - i$, and do not depend on the specific input x. Interchanging i and j gives the symmetric bound when $i > j$, so the proof is complete. □

A.2 Proof of Corollary 4.3

Proof. By Prop. 4.2, for every pair of layers (i, k) there exist strictly positive constants $c_{ik}^{\min}$ and $c_{ik}^{\max}$ such that

$$c_{ik}^{\min}\, S_k(x) \;\le\; S_i(x) \;\le\; c_{ik}^{\max}\, S_k(x). \quad (27)$$

Fix a layer index i and apply the left-hand side of Eq. 27:

$$S(x,W) \;\geq\; \Big(\sum_{k=1}^{L}\big(c_{ik}^{\min} S_k(x)\big)^2\Big)^{\frac{1}{2}} \\ \;=\; \Big(\sum_{k=1}^{L}\big(c_{ik}^{\min}\big)^2\Big)^{\frac{1}{2}} S_i(x). \tag{28}$$

Denote $C_i^{\min} = \big(\sum_{k=1}^{L}(c_{ik}^{\min})^2\big)^{\frac{1}{2}} > 0$. Then

$$C_i^{\min}\, S_i(x) \;\leq\; S(x,W). \tag{29}$$

Using the right-hand side of Eq. 27 yields

$$S(x,W) \leq \Big(\sum_{k=1}^{L}\big(c_{ik}^{\max} S_i(x)\big)^2\Big)^{\frac{1}{2}} = \Big(\sum_{k=1}^{L}\big(c_{ik}^{\max}\big)^2\Big)^{\frac{1}{2}} S_i(x), \tag{30}$$

so with $C_i^{\max} = \big(\sum_{k=1}^{L}(c_{ik}^{\max})^2\big)^{\frac{1}{2}}$ we have

$$S(x,W) \;\leq\; C_i^{\max}\, S_i(x). \tag{31}$$

Combining Eq. 29 and Eq. 31 gives the two-sided inequality

$$C_i^{\min}\, S_i(x) \;\leq\; S(x,W) \;\leq\; C_i^{\max}\, S_i(x). \tag{32}$$

All constants are strictly positive and layer-dependent only. Therefore, increasing the sensitivity of *any* single layer i necessarily induces a proportional increase in the aggregate sensitivity. More concretely, if an optimization step multiplies $S_i(x)$ by a factor $\rho > 1$, then by Eq. 32

$$S_{\text{new}}(x,W) \;\geq\; \rho\,\frac{C_i^{\min}}{C_i^{\max}}\, S_{\text{old}}(x,W), \tag{33}$$

which is strictly larger whenever $\rho > 1$. Hence, optimizing a single-layer sensitivity simultaneously enhances the overall sensitivity of the entire model, completing the proof. □

A.3 Proof of Prop. 4.4

Proof. Let d_L be the output dimension of the final linear layer, so that $W_L \in \mathbb{R}^{d_L \times d_{L-1}}$ and $f_L(x) = a_L = W_L\, a_{L-1} + b_L$, $a_{L-1} = f_{L-1}(x)$. Since most LLMs do not use any nonlinear activation functions like $ReLu(\cdot)$ in the last layer, the last layer of LLMs performs the above linear operations on the output of the previous layer, the Jacobian of a_L with respect to W_L is the Kronecker product

$$\frac{\partial a_L}{\partial W_L} \;=\; I_{d_L} \otimes a_{L-1}^{\top}, \tag{34}$$

where I_{d_L} is the $d_L \times d_L$ identity matrix. Flattening this tensor in the canonical way (stacking the d_L row-wise gradients) yields a matrix of size $d_L \times (d_L\, d_{L-1})$ whose Frobenius norm equals

$$S \;=\; \Big\|\tfrac{\partial a_L}{\partial W_L}\Big\|_F = \big\|I_{d_L} \otimes a_{L-1}^{\top}\big\|_F \\ = \sqrt{d_L}\, \|a_{L-1}\|_2. \tag{35}$$

Since d_L is a fixed positive constant for the model, there exist layer-dependent constants $C_1 = C_2 = \sqrt{d_L} > 0$ such that

$$C_1\, \|f_{L-1}(x)\|_2 \;=\; S \;=\; C_2\, \|f_{L-1}(x)\|_2. \tag{36}$$

Hence, the gradient-defined sensitivity S of the last layer is *positively correlated* to the ℓ_2 norm of the penultimate layer's output, which establishes Prop. 4.4. □

A.4 Proof of Prop. 4.5

Proof. Let $f(x;W)$ be the logits output of an LLM for input x, and define $p = \text{softmax}\big(f(x;W)\big)$, with $T_K(p)$ the top-K token distribution derived from p. During model tampering, since attackers often strive to make their tampering behavior ΔW as covert as possible, we can consider ΔW as a perturbation of W and a first-order Taylor expansion of $f(x;W)$ is valid. Then there exists a remainder $R(\Delta W)$ with

$$f(x;W+\Delta W) = f(x;W) + \frac{\partial f(x;W)}{\partial W}\Delta W + R(\Delta W), \tag{37}$$

where $\|R(\Delta W)\|_F = o\big(\|\Delta W\|_F\big)$. Let $\Delta z = f(x;W+\Delta W) - f(x;W)$. Then from Eq. 37 we have, for ΔW,

$$\|\Delta z\|_F \geq \left\|\frac{\partial f(x;W)}{\partial W}\Delta W\right\|_F - \|R(\Delta W)\|_F \\ \geq S(x,W)\, \|\Delta W\|_F. \tag{38}$$

Since the softmax function is *locally Lipschitz continuous*, there exists a constant $L_{\text{sm}} > 0$ such that

$$\|\text{softmax}(z+\Delta z) - \text{softmax}(z)\|_2 \geq C_{\text{sm}}\, \|\Delta z\|_F. \tag{39}$$

Let $p = \text{softmax}(z)$ and $p' = \text{softmax}(z + \Delta z)$. Then from Eq. 38 and Eq. 39, it follows that

$$\|p' - p\|_2 \geq C_{\text{sm}}\, S(W)\, \|\Delta W\|_F. \tag{40}$$

Furthermore, we can assume the top-K mapping T_K is locally Lipschitz with constant $C_{TK} > 0$.

This assumption is reasonable because when the top-K probabilities are well-separated, small perturbations in the full distribution only lead to small changes in the top-K selection, ensuring local stability. By this assumption, we have

$$d_{TK}\big(T_K(p), T_K(p')\big) \geq C_{TK}\, \|p' - p\|_2, \quad (41)$$

where $d_{TK}(\cdot,\cdot)$ is a metric measuring the difference between two top-K distributions. Substituting Eq. 40 into Eq. 41, we have

$$d_{TK}\big(T_K(p), T_K(p')\big) \geq C_{TK}\, C_{\text{sm}}\, S(W)\, \|\Delta W\|_F. \quad (42)$$

Let $C = C_{TK}\, C_{\text{sm}} > 0$ and $W' = W + \Delta W$, we can obtain the following desired result:

$$d_{TK}\Big(T_K(p),\, T_K\big(p'\big)\big)\Big) \geq C\, S(W)\, \|\Delta W\|_F. \quad (43)$$

Taking expectations for both sides of Eq. 43, we have

$$\mathbb{E}_{\Delta W}\big[d_{TK}\big(T_K(p), T_K(p')\big)\big] \;\geq\; \underbrace{\frac{\rho}{\sqrt{d_W}}C}_{=C_{\text{tot}}}\; S(x, W), \quad (44)$$

where $C_{\text{tot}} = \frac{\rho}{\sqrt{d_W}} C = \frac{\rho}{\sqrt{d_W}} C_{sm} C_{TK}$. Thus, for a fixed $\|\Delta W\|_F$, a larger sensitivity $S(x, W)$ implies a larger expected shift in the top-K token distribution, which can be used as the signal of detecting model tampering. □

A.5 Proof of Prop. 5.1

Proof. Prop. 4.4 establishes that for any fingerprint x with penultimate embedding $e = f_{L-1}(x) \in \mathbb{R}^{d_{L-1}}$, the single-sample sensitivity satisfies

$$S(x, W) \;=\; \sqrt{d_L}\; \|e\|_2. \quad (45)$$

Hence, it suffices to show that with high probability there is at least one embedding whose norm is of order $\sqrt{d_{L-1}}$ and that the total pairwise distance is of order $N^2 d_{L-1}$.

Under the isotropic sub-Gaussian assumption Eq. 9, each embedding $e_n = f_{L-1}\big(x^{(n)}\big)$ has mean zero, $\mathrm{Cov}(e_n) = \sigma^2 I$, and sub-Gaussian norm $\|e_n\|_{\psi_2} \leq \kappa\sigma$. In particular, a standard upper-tail bound for Euclidean norms of such vectors (see (Wainwright, 2019)) implies that for any $t > 0$,

$$\Pr\Big[\|e_n\|_2 \geq \sigma\big(\sqrt{d_{L-1}} + t\big)\Big] \;\leq\; e^{-c_2 t^2}, \quad (46)$$

where $c_2 > 0$ is a universal constant. Setting

$$t \;=\; \sqrt{2\,\ln\!\Big(\tfrac{2N}{\delta}\Big)} \Longrightarrow s_{N,\delta} \;=\; \sigma\Big(\sqrt{d_{L-1}} + \sqrt{2\,\ln\tfrac{2N}{\delta}}\Big), \quad (47)$$

and applying a union bound over the N independent embeddings gives

$$\Pr\Big[\max_{1\leq n\leq N} \|e_n\|_2 < s_{N,\delta}\Big] \;\leq\; \frac{\delta}{2}, \quad (48)$$

so that with probability at least $1 - \frac{\delta}{2}$, $\max_{1\leq n\leq N} \|e_n\|_2 \;\geq\; s_{N,\delta}$ and such that

$$S_{\max} = \max_n S\big(x^{(n)}, W\big) \;\geq\; \sqrt{d_L}\, s_{N,\delta}. \quad (49)$$

Next, define the total pairwise squared distance $D = \sum_{1\leq m<n\leq N} \|e_m - e_n\|_2^2$. An unbiased-variance identity rewrites D in terms of the sample second-moment and the sum of embeddings:

$$D = N \sum_{n=1}^{N} \|e_n\|_2^2 - \Big\|\sum_{n=1}^{N} e_n\Big\|_2^2. \quad (50)$$

Since $\|e_n\|_2^2$ is sub-exponential with mean $\mathbb{E}[\|e_n\|_2^2] = \sigma^2 d_{L-1}$, Bernstein's inequality yields that for a suitable universal constant,

$$\Pr\Big[\Big|\tfrac{1}{N}\sum_{n=1}^{N} \|e_n\|_2^2 - \sigma^2 d_{L-1}\Big| \;\leq\; \sigma^2\sqrt{\tfrac{d_{L-1}}{N}}\Big] \;\geq\; 1 - \frac{\delta}{4}. \quad (51)$$

Meanwhile, with vector Bernstein bound, we have

$$\Pr\Big[\Big\|\sum_{n=1}^{N} e_n\Big\|_2^2 \;\leq\; 4\,\sigma^2\, d_{L-1}\, N\Big] \;\geq\; 1 - \frac{\delta}{4}. \quad (52)$$

By a union bound, both Eq. 51 and Eq. 52 hold simultaneously with probability at least $1 - \frac{\delta}{2}$. In that case, since d_{L-1} and N are large enough that $\sqrt{d_{L-1}/N} \leq \frac{1}{2}\, d_{L-1}$, we can obtain

$$\frac{1}{N}\sum_{n=1}^{N} \|e_n\|_2^2 \;\geq\; \sigma^2 d_{L-1} - \sigma^2\sqrt{\tfrac{d_{L-1}}{N}} \;\geq\; \tfrac{1}{2}\,\sigma^2 d_{L-1}$$
$$\Big\|\sum_{n=1}^{N} e_n\Big\|_2^2 \;\leq\; 4\,\sigma^2\, d_{L-1}\, N. \quad (53)$$

Substituting into Eq. 50 shows that for some constant $c > 0$,

$$D \;=\; N\sum_{n=1}^{N} \|e_n\|_2^2 - \Big\|\sum_{n=1}^{N} e_n\Big\|_2^2 \;\geq\; c\,\sigma^2\, N^2\, d_{L-1}. \quad (54)$$

Finally, combining Eq. 49 and Eq. 54, we see that with probability at least $1 - \delta$ both $S_{\max} \geq \sqrt{d_L}\, s_{N,\delta}$ and $D \;\geq\; c\,\sigma^2\, N^2\, d_{L-1}$ hold. On this

Table 4: Tamper detection rate(%) with detection token numbers from 1 to 3, while tampering type is backdoor injection fine-tuning with the last 3 layers.

Models	Qwen-2.5-0.5B		Qwen-2.5-7B		LLaMA-3.2-1B		LLaMA-3-8B		Mistral-7B	
token numbers / N_S	1	5	1	5	1	5	1	5	1	5
1	47.6	99.6	65.6	100	82.4	100	95.8	100	65.0	100
2	58.2	100	73.4	100	95.2	100	100	100	75.2	100
3	62.4	100	79.6	100	98.8	100	100	100	80.6	100

event the joint detection radius $r_{\min} = \tau / \big(\eta\, S_{\max}\big)$ satisfies

$$r_{\min} \;\leq\; \frac{\tau}{\eta\,\sqrt{d_L}\,s_{N,\delta}} \;=\; O\Big(\frac{\tau}{\eta\,\sigma\,\sqrt{d_L\,d_{L-1}}}\Big), \tag{55}$$

as claimed. Moreover, increasing D only improves the probability that Eq. 54 holds with an even larger margin, so maximizing the pairwise-distance statistic D enhances the empirical probability of the favorable event. Concretely, when this event occurs, we know both that $S_{\max} = \max_n S(x^{(n)}, W)$ is large enough for tamper detection (because one $\|e_n\|$ is large enough) and that the embeddings are well spread out such that leave no "hiding" direction for an adversary, meaning the worst-case perturbation size you can hide under (i.e. remain below threshold τ on all N prompts) shrinks dramatically. By our concentration bounds, this happens with probability at least $1 - \delta$. This completes the proof of Prop. 5.1. □

B Additional Experimental Results

B.1 Tamper Detection with Multiple Detection Token Positions

As mentioned in Section 6.1.1, ESF can utilize more than one output token at different positions within each fingerprint sample to validate whether a model has been tampered with. For each fingerprint sample, we record all randomness sets corresponding to all detection token positions. During detection, we generate outputs from the model in real time and compare the tokens produced at each detection position with the corresponding predefined randomness sets. If we find that any token at any detection position falls outside its associated randomness set, we flag the model as tampered.

We conduct experiments on backdoor injection via fine-tuning the last three layers, evaluating different token positions. Table 4 demonstrates that increasing the number of verification tokens consistently enhances the detection rate. This result further confirms that even a single token achieves satisfactory detection performance when used for validation.

To further investigate the effect of token position, we conduct an experiment comparing the performance of selecting the first token versus the third token as the validation token. We use backdoor injection via fine-tuning on the Qwen-2.5-0.5B model. The experimental results, shown in Table 5, indicate that although both tokens are sensitive to model tampering, the third token performs slightly worse than the first.

Table 5: Tamper detection rate(%) with detection token positions from 1 to 3, while tampering type is backdoor injection fine-tuning with last 3 layers on Qwen-2.5-0.5B.

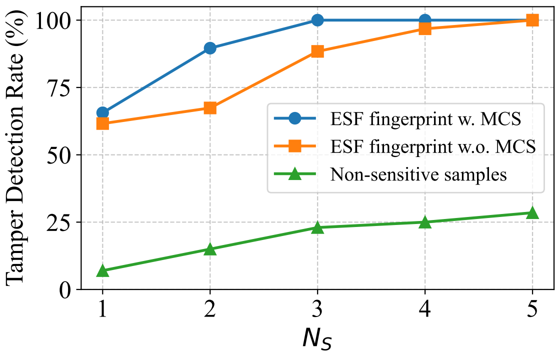
token position	1st token		2nd token		3rd token	
Ns	1	5	1	5	1	5
Full Parameter	92.2	100	91.4	100	89.6	100
Freeze	75.0	100	74.4	100	71.4	100
LoRA	47.6	100	47.2	100	45.4	100

This performance discrepancy occurs because the third token records more possible candidate tokens than the first, due to variations in the tokens generated before it (i.e., the first and second tokens). Since the detection does not consider preceding tokens, it cannot detect tampering if the first two tokens mismatch while the third token matches. This suggests that the first token is marginally more reliable than the third when used as a single verification token. However, we note that both tokens are less effective for tamper detection compared to using all three token positions together.

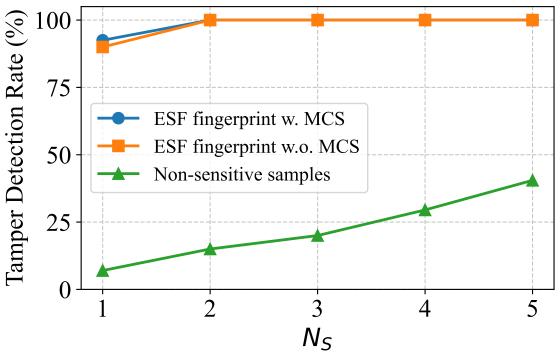
Since fingerprint samples are generated by maximizing sensitivity to model tampering—without considering specific tampering types—and the optimization process is identical for detection tokens at different positions, we do not expect one token position to become systematically more sensitive to a particular tampering type than another. In other words, the fingerprint generation process is both tampering type–agnostic and position–agnostic.

B.2 Further Explanation of Mismatched Randomness Levels

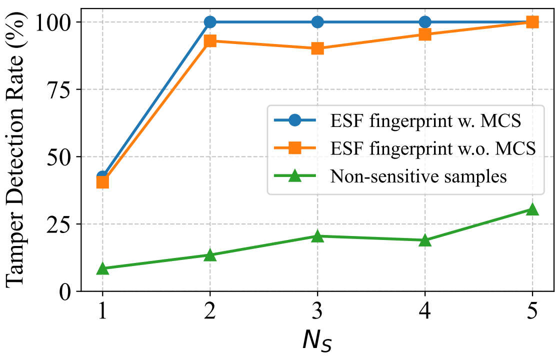
In our experiments, we consider two scenarios involving mismatches in randomness levels. The first scenario arises when the tampered model's randomness level is lower than that used in our Randomness-Set Consistency Checking. The second scenario occurs when the tampered model's randomness level is higher. In the first case, since



(a) Qwen2.5-7B



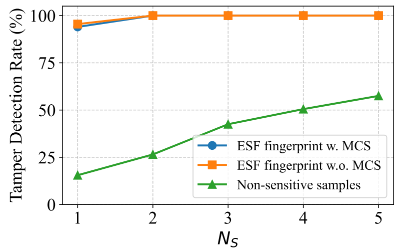
(b) LLaMA-3-8B



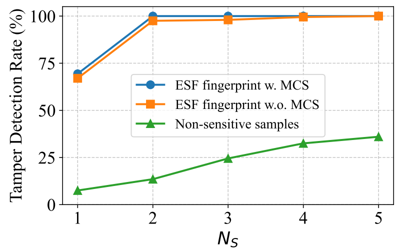
(c) Mistral-7B

Figure 7: Tamper detection rate (%) with different N_S and different fingerprint picking strategies on different models, while the tampering type is poison data finetuning with the last 3 layers.

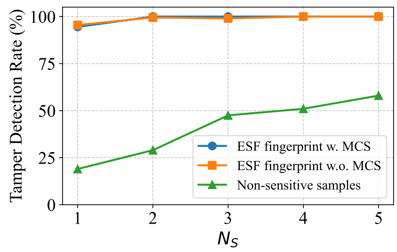
ESF can completely mitigate the impact of output randomness on detection, it enables effective and reliable identification of model tampering. In the second case, the attacker alters the model's inference randomness configuration, resulting in a randomness level that slightly exceeds the one recorded by Randomness-Set Consistency Checking. Consequently, the model's output range may surpass the bounds of the Randomness-Set, potentially leading to false positives. As reported in Section 6.4, the false positive rate in this scenario remains low—approximately 2.8% for $N_s = 5$.



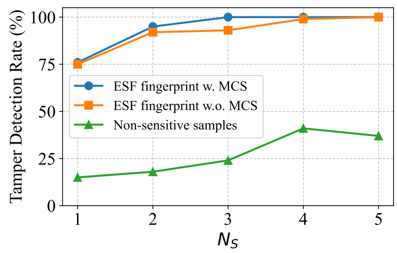
(a) Clean data with full layers



(b) Clean data with LoRA



(c) Poison data with full layers



(d) Poison data with LoRA

Figure 8: Tamper detection rate (%) under different fingerprint selection strategies for different tampering types on Qwen2.5-0.5B.

Table 6: Tamper detection rate(%) with full parameter optimization (FPO) with Eq.2, while tampering type is poisoned data fine-tuning with last 3 layers.

Models	Qwen-2.5-0.5B		Qwen-2.5-7B		LLaMA-3.2-1B		LLaMA-3-8B		Mistral-7B	
Optimization Method / N_S	1	5	1	5	1	5	1	5	1	5
FPO	62.2	100	73.2	100	86.2	100	98.0	100	74.4	100
ESF	47.6	99.6	65.6	100	82.4	100	95.8	100	65.0	100

B.3 Ablation Study on Other Settings

We have presented ablation study results in Section 6.5. Here, we provide additional results incorporating more models and tampering types. Fig. 7 displays the outcomes of full-layer and LoRA fine-tuning with both clean and poisoned data on Qwen2.5-0.5B. Fig. 8 presents the results of last-three-layer fine-tuning with poisoned data on Qwen2.5-7B, LLaMA-3-8B, and Mistral-7B.

These extended results further demonstrate that our *MCS* consistently improves multi-sample detection performance across various tampering types and model architectures.

B.4 Fingerprint Generation with ESF and Gradient-based Sensitivity

Within the ESF framework, we employ the optimization objective delineated in Eq. 6. Similarly, the optimization objective presented in Eq. 2 can also be utilized, which involves leveraging the complete gradient of the model as sensitivity for fingerprint optimization. Table 6 shows the model tamper detection rate for 500 fingerprints optimized using the optimization function outlined in Eq. 2. The results in Table 6 reveal that the fingerprints optimized in this manner exhibit a marginally higher detection efficiency when $N_S = 1$ compared to those generated by ESF. However, the disparity in detection accuracy becomes negligible under the condition of $N_S = 5$.