iSeal: Encrypted Fingerprinting for Reliable LLM Ownership Verification

Zixun Xiong[1], Gaoyi Wu[1], Qingyang Yu[1], Mingyu Derek Ma[2], Lingfeng Yao[3], Miao Pan[3], Xiaojiang Du[1], Hao Wang[1]

[1]Department of Electrical and Computer Engineering, Stevens Institute of Technology
[2]Genentech
[3]Department of Electrical and Computer Engineering, University of Houston
zxiong9@stevens.edu, gwu13@stevens.edu, qyu13@stevens.edu, hi@derek.ma, lyao12@uh.edu, mpan2@uh.edu, xdu16@stevens.edu, hwang9@stevens.edu

Abstract

Given the high cost of large language model (LLM) training from scratch, safeguarding LLM intellectual property (IP) has become increasingly crucial. As the standard paradigm for IP ownership verification, LLM fingerprinting thus plays a vital role in addressing this challenge. Existing LLM fingerprinting methods verify ownership by extracting or injecting model-specific features. However, they overlook potential attacks during the verification process, leaving them ineffective when the model thief fully controls the LLM's inference process. In such settings, attackers may share prompt-response pairs to enable fingerprint unlearning, or manipulate outputs to evade exact-match verification. We propose *iSeal*, the first fingerprinting method designed for reliable verification when the model thief controls the suspected LLM in an end-to-end manner. It injects unique features into both the model and an external module, reinforced by an error-correction mechanism and a similarity-based verification strategy. These components are resistant to verification-time attacks, including collusion-based fingerprint unlearning and response manipulation, backed by both theoretical analysis and empirical results. *iSeal* achieves 100% Fingerprint Success Rate (FSR) on 12 LLMs against more than 10 attacks, while baselines fail under unlearning and response manipulations.

1 Introduction

Large language models (LLMs) have recently achieved remarkable success in a wide range of applications (Zhou et al. 2025; Singh et al. 2025; Zeng et al. 2025). However, training LLMs from scratch remains expensive in terms of computational resources and financial cost (Meta AI 2024). Therefore, LLMs constitute valuable intellectual property (IP) for the model owner, making it critical to build reliable fingerprinting methods for IP ownership verification.

In practice (Grotto et al. 2024; The Fashion Law 2024) of LLM ownership verification, model thieves often acquire proprietary models through internal leaks or security breaches, and deploy model copies as public APIs for profit. Since model thieves control both the model weights and the inference process, they can employ diverse attacks that render unprotected fingerprinting-based verification ineffective. Furthermore, since the verifier only has black-box access, lacking visibility into model weights or internal states, ownership verification becomes significantly more challenging. As summarized in Table 1, existing fingerprinting methods can be classified into passive and proactive approaches, depending on whether the fingerprint is proactively injected during training (Li et al. 2025). Passive methods (Zeng et al. 2024; Zhang et al. 2024b; Jin et al. 2024; Gubri et al. 2024) extract model-specific features after training has completed for ownership verification. However, passive methods do not alter the model itself, and therefore lack *forgery resistance* (Li et al. 2023): anyone with the API access can extract similar features and falsely claim ownership. In contrast, proactive methods (Gu et al. 2022; Xu et al. 2024a) embed unique ownership signatures into the model during training, ensuring only the rightful owner can perform successful verification. However, existing proactive methods are vulnerable to fingerprint removal, as they lack an *external secret*, *i.e.* the fingerprint is embedded solely in the model weights and can be removed by an adversary with full access. Moreover, in practical scenarios (*e.g.*, a lawsuit), proactive methods require the model owner or a third-party verifier to publicly present at least one prompt-response pair, even to the model thief, to demonstrate ownership. However, if the model thief colludes with another adversary, they may exploit the disclosed prompt-response pair to simply unlearn it or reverse engineer the fingerprinting process, potentially enabling full removal of the fingerprint in subsequent disputes (*e.g.*, another lawsuit). What's more, since the model thief can manipulate the model's response to evade verifi-

Type	Method	Suspected Model	Forgery Resistance	External Secret	Verification Robustness
Passive	HuRef	White Box	✗	✓	✗
	REEF	White Box	✗	✗	✗
	ProFLingo	Black Box	✗	✗	✗
	TRAP	Black Box	✗	✗	✗
Proactive	WLM	Black Box	✓	✗	✗
	IF	Black Box	✓	✗	✗
	iSeal (Ours)	Black Box	✓	✓	✓

Table 1: Comparison between *iSeal* and existing methods, including HuRef (Zeng et al. 2024), REEF (Zhang et al. 2024b), ProFLingo (Jin et al. 2024), TRAP (Gubri et al. 2024), WLM (Gu et al. 2022), and IF (Xu et al. 2024a).

cation, prior proactive methods that rely heavily on exact matching are prone to failure. In summary, these vulnerabilities highlight a critical limitation of prior methods: the lack of *verification robustness*.

In this paper, we present *iSeal*, the first method to provide practical and reliable ownership verification against the aforementioned attacks within a theoretical bound. *First*, we introduce an external encoder to decouple the fingerprint from the model, preventing reverse engineering via weight access alone. *Second*, its cryptographic design, with strong diffusion and confusion, ensures that even prompt-response pairs reveal no useful information, thwarting unlearning and fingerprint inference under collusion. *Third*, to defend against response manipulation, we adopt similarity-based verification instead of fragile exact matching. *Finally*, an error correction module further provides provable robustness, enabling recovery even when the prior module fails. With extensive experiments, *iSeal* outperforms existing methods, maintaining a 100% verification success rate under attacks in API-only access settings, while previous works drop to 0%.

2 Preliminaries

In this paper, we focus on model fingerprinting, which is different from model watermarking. Although both research avenues target ownership verifications, they have a fundamental difference: Model watermarking targets the *model output*, while model fingerprinting seeks to safeguard the *model itself* as discussed in previous works (Xu et al. 2024a; Zeng et al. 2024). Model fingerprinting can be categorized as passive and proactive fingerprinting (Li et al. 2025).

Passive Fingerprinting. This category of methods aims to extract the unique characteristics of different LLMs to serve as fingerprints for ownership verification. They are considered *passive* because they do not actively modify or embed external information into the model but instead rely on analyzing its pre-existing behaviors or outputs. HuRef (Zeng et al. 2024) treats a portion of the model parameters as the unique characteristics and maps these parameters to human-readable images. However, HuRef needs white-box access to the suspected model weights, which is impractical in intellectual property litigation since allowing arbitrary access to model weights would enable overclaiming attacks, where adversaries falsely assert ownership of others' models and steal the model weights. REEF (Zhang et al. 2024b) and EasyDetector (Zhang et al. 2024a) share similar ideas and issues as HuRef. To overcome such challenges, TRAP (Gubri et al. 2024), ProFLingo (Jin et al. 2024), and RAP-SM (Xu et al. 2025b) optimize the suffix or prefix of the model input given a certain output (*e.g.*, 314 for a random number generation, an answer that defies common sense) as the unique characteristics. However, these passive fingerprinting techniques do not require model training, making them lack *forgery resistance*, since anyone can derive such fingerprints, and multiple parties with API access can falsely claim ownership. In contrast, proactive fingerprinting binds the fingerprint to the training process itself—only the legitimate trainer can produce the fingerprinted model, making ownership verifiable and exclusive.

Proactive Fingerprinting. This class of methods is designed to inject private knowledge into the model through training or by manipulating its weights. WLM (Gu et al. 2022) treats the trigger (e.g., "cf") and its corresponding pre-defined answer (e.g., "Positive" in sentiment analysis) as private knowledge, which is injected into the model via fine-tuning. IF (Xu et al. 2024a) extends this idea by wrapping the private knowledge in instruction-style prompts to increase its complexity, and injects it using an adapter, making the fingerprint more resistant to removal through fine-tuning. PLMark applies contrastive learning on the "[CLS]" token as the private knowledge; however, it has been proven ineffective in large language models and easy to be removed by fine-tuning (Xu et al. 2024a). UTF (Cai et al. 2024) is a simplified version of IF with a different prompt template. MYL (Xu et al. 2025a) can be seen as a variant of IF, where ownership is verified through repeated trigger queries and statistical testing, making it easier to be reverse-engineered and removed. FP-VEC (Xu et al. 2024b) directly injects a fingerprint into the model by adding a trained vector to its parameters, without fine-tuning the model itself. Similar to MYL, it also needs multiple queries, increasing the attack surface. EditMark (Li et al. 2025) uses output precision on a sequence of math questions as private knowledge, but this makes verification fragile—since the fingerprint is jointly defined, unlearning even one question breaks the whole verification. PlugAE (Yang et al. 2025) is similar to WLM (Gu et al. 2022) but optimizes the embedding of a trigger token instead of modifying model weights. However, it introduces a new trigger token, which can be easily spotted and removed by a model thief via inspecting the vocabulary or embedding matrix. Furthermore, we observe that most proactive fingerprinting methods embed the fingerprint solely within the model itself. However, under a realistic threat model where adversaries can access the model weights, such methods offer limited security guarantees.

In contrast, our method uses an external encoder with a secret key to realize an *external secret*, offering stronger resilience. It further achieves *verification robustness*, resisting fingerprint removal even under prompt leakage and response manipulation enabled by confusion, diffusion, similarity matching, and error correction, where previous works fail.

3 Proposed Methods

3.1 Problem Setting

Threat Model. Our threat model is motivated by potential intellectual property lawsuits surrounding LLMs (Grotto et al. 2024; The Fashion Law 2024). Our threat model involves four entities: the model owner, the model thief, a judge, and a registration authority. The model owner trains and legally registers the original model. The model thief acquires the model, gains read access to its weights, and illegally deploys the model copy as a public API. They can collude with others to share verification prompts and manipulate outputs. The judge and the registration authority serve as third-party entities to determine whether the suspected

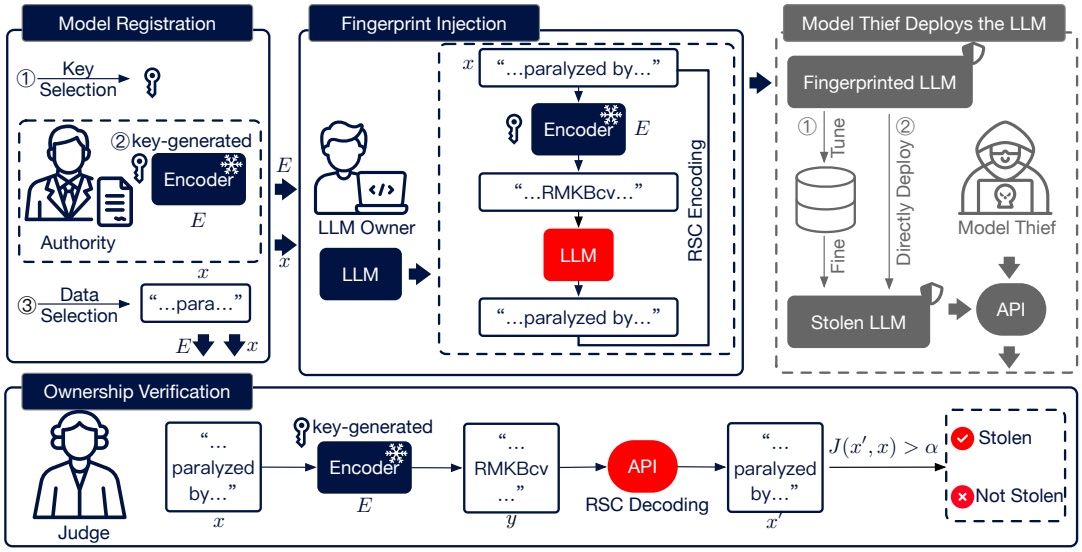

Figure 1: Pipeline of *iSeal*. A secret-keyed encoder maps plaintexts to ciphertexts, and the LLM is trained to reconstruct Reed-Solomon Code (RSC) encoded targets. Ownership is verified by querying the suspect API and matching decoded outputs.

model is a stolen copy. We assume the judge and the registration authority are trustworthy, with API access to the suspected and registered models but no knowledge of their internal architecture. Our goal is to develop a reliable fingerprinting method that can be embedded by the model owner and verified by a third-party judge. In addition to this basic functionality, we aim to address key limitations identified in Table 1: vulnerability to ownership overclaim, lack of external secrets, and no resistance to verification-time attacks.

Challenges. Under our threat model, a model thief has white-box access to the stolen model copy, enabling full inspection and modification of its parameters. During a verification process (e.g., in a legal dispute), the thief may obtain a fingerprint prompt-response pair and share it with collaborators, enabling coordinated unlearning to invalidate future ownership claims. Moreover, they can manipulate outputs at inference time to evade verification. In contrast, the judge has only black-box access to the model via its public API, without any knowledge of the model's internal parameters, and must prevent overclaims of ownership.

Our Solutions. First, we design an *external secret* module using a key-generated encoder that is not embedded in the model, preventing the thief from accessing or removing the fingerprint even with full model access. Second, the encoder's cryptographic properties proved in the next section ensure that collusion-based unlearning is ineffective, as removing a few records cannot erase the full prompt–response relationship. Third, we adopt similarity-based matching, which tolerates edits or deletions in responses, mitigating manipulation attacks in practice. We further incorporate an error-correction module that provides theoretical guarantees against such manipulations. Finally, to prevent ownership overclaiming, prompts are exclusively managed and queried by the judge, only the judge can initiate fingerprint verification, and only models trained with the correct encoder can pass, ensuring that passive fingerprinting or self-claimed ownership are insufficient.

3.2 Overview of *iSeal*

Figure 1 shows the pipeline of *iSeal*. *iSeal* has three components: model registration, fingerprint injection, and ownership verification. 1) In fingerprint registration, the owner submits a request; the authority samples a key, initializes an encoder, and returns it with selected plaintexts. 2) In fingerprint injection, the encoder encrypts these plaintexts, and the LLM is fine-tuned to reconstruct them with error-correction targets. 3) During verification, the judge uses the owner's encoder and authority-held plaintexts to query the suspect API; decoded outputs are similarity-matched to tell if LLM is a stolen copy. The pseudocode is provided in the full-length version (Xiong et al. 2025).

3.3 Model Registration and Fingerprint Injection

Initially, the registration authority picks a random key K of length k. We select hex code, so the probability of the event $\text{Event}_{\text{same}}$ that another LLM owner picks the same key is $Pr(\text{Event}_{\text{same}}) = \frac{1}{16^k}$ and the keyspace size of LLM in our fingerprinting method $\text{Cap}_{\textit{iSeal}}$ is $\text{Cap}_{\textit{iSeal}} = 16^k$. In this paper, we select $k = 32$ for all experiments. Thereby, $Pr(\text{Event}_{\text{same}}) = 16^{-32} \approx 0$, indicating that our method avoids model ownership overclaim by ensuring an extremely low probability of key collision. Moreover, the key space is on the order of 10^{38}, which is much more than the estimated world population (8.1×10^9), demonstrating the profound keyspace size of our system to embed unique fingerprints at scale. Besides, robustness experiments and the proof in the next section show that even manipulating one logit of the key causes model ownership verification to fail, which further supports the statements above. After selecting the pri-

vate key K for the current model owner, the authority uses it to deterministically derive the corresponding seed value to initialize each layer weight of the encoder:

$$\text{seed}_i = \text{int}(\text{HMAC-SHA256}(K, i)) \bmod 2^k,$$

where i is the layer index and HMAC-SHA256 is HMAC construction (Bellare, Canetti, and Krawczyk 1997) using SHA-256 (National Institute of Standards and Technology 2015) as the underlying hash function. The verification authority uses HMAC-SHA256 to map the secret key to a seed, as it ensures computational security under standard cryptographic assumptions, guaranteeing that no polynomial-time adversary can infer key information from observing the encoder's outputs (Backendal et al. 2023). Thereafter, *iSeal* derives the encoder E, which is composed of N layers initialized by the aforementioned seeds separately. The model owner receives the encoder and fine-tunes the target LLM in an encoder-decoder fashion, where LLM serves as the decoder, the weight of which is updated using an adapter (Xu et al. 2024a). The training objective is to minimize the reconstruction loss of the plaintext and the decoder (*i.e.*, LLM $\mathcal{M}$) output: $\mathcal{M}^* = \arg\min_{\mathcal{M}} \mathcal{L}_{\text{CSE}}(\mathcal{M}(y = E(x)), x)$, where x is the plaintext, y is the output of the encoder, and $\mathcal{L}_{\text{CSE}}(\cdot, \cdot)$ is the cross-entropy loss. Moreover, we apply conditional language learning (Zhang, Li, and Wu 2024) so that our fingerprint is more resistant to fine-tuning (Zhang, Li, and Wu 2024; Xu et al. 2024a). The objective function is reformulated as follows:

$$\mathcal{M}^* = \arg\max_{\mathcal{M}} \; p_{\mathcal{M}}(x \mid \mathcal{M}(y = E(x))). \tag{1}$$

As a result, our fingerprint is injected into the target LLM. In practice, the plaintext dataset D (where $x \in D$) is assigned by the registration authority to prevent ownership overclaim, as discussed at the beginning of this section. Moreover, we freeze the encoder after initialization to prevent it from learning an optimal representation that would allow it to reconstruct the plaintext directly, even when applied to the base model without our fingerprint injected. Besides, jointly training the encoder and the decoder will cause slower convergence. The full-length version (Xiong et al. 2025) further supports this point.

We did not use conventional encryption methods, such as AES for the following reasons: 1) Conventional encryption methods creates poor gradient flow due to its highly nonlinearity and discontinuous operations, causing vanishing gradients during training. 2) Methods such as AES destroy semantic information by design, making it extremely difficult for decoders to learn meaningful inverse mappings. 3) Experimental results show AES leads to slow convergence and poor reconstruction quality compared to our approach. This view is supported in the full-length version. We discuss the impact of picking up different N in the encoder structures in the full-length version (Xiong et al. 2025).

3.4 Ownership Verification

The judge is provided with a dataset D assigned by the registration authority. This dataset is composed of several plaintexts (*i.e.*, x). Before verification, the model owner should provide an encrypted encoder E that is maintained by the registration authority. This encoder can be accessed only by the judge and the model owner. The judge is also provided with a suspected API of the LLM without any prior knowledge of the model. The LLM behind the API M' may have been directly stolen from the claimed model owner or further fine-tuned on an unknown dataset as shown in Figure 1.

The judge selects a plaintext input x and feeds it into the frozen encoder to obtain the encoded representation $y = E(x)$. Then, the judge prompts the API M' with the encoded representation. If the similarity between the LLM output and the plaintext x exceeds a certain threshold, we can consider the model behind the API M is stolen from the claimed model owner. We formalize the above process as follows:

$$\mathcal{J}(M', E, x) = \begin{cases} \textit{“stolen”} & \text{if } J(M'(E(x)), x) > \alpha \\ \textit{“not stolen”} & \text{else} \end{cases},$$

where $J(\cdot, \cdot)$ is a function to measure the similarity between two sentences. In practice, we apply BLEU scores (Papineni et al. 2002); higher scores indicate greater similarity between the two inputs. Among automatic metrics, BLEU remains one of the most optimal and standard choices for evaluating the correspondence between a generated string and a ground-truth reference, especially when human evaluation is infeasible (Papineni et al. 2002).

Considering the model thief might manipulate the output to bypass our fingerprinting method, we further apply an updated version of RSC (Wicker and Bhargava 1999) to encode $M(y)$ into a representation that is resistant to manipulation and decode it for verifications, as is proven to achieve optimal performance against similar situations (Singleton 2003). Specifically, we reformulate Equation 1 as follows to inject the encoded fingerprint:

$$\mathcal{M}^* = \arg\max_{\mathcal{M}} \; p_{\mathcal{M}}(E'(x) \mid \mathcal{M}(E(x))),$$

where $E'(\cdot)$ is encoding component of the RSC. Thus, for model ownership verification, we have the new decision function:

$$\mathcal{J}'(M', E, x) = \begin{cases} \textit{“stolen”} & \text{if } J(D'(M'(y)), x) > \alpha \\ \textit{“not stolen”} & \text{else} \end{cases},$$

where D' is the decoding component of the RSC. To prevent multiple model thieves from colluding, the judge selects a different plaintext $z \neq x$ the next time the same model owner claims ownership of another suspected API, and Figure 3 further demonstrates *iSeal* is resistant to unlearning attacks caused by such colluding. Moreover, since *iSeal* does not rely on exact matches and incorporates the error correction mechanism described above, it remains robust against response manipulation attacks, as shown in Figure 4. The proof can be found in the appendix of the full-length version (Xiong et al. 2025).

4 Proof of Security Properties[1]

Unlike previous works (Xu et al. 2024a; Gubri et al. 2024), our method places greater emphasis on the conventional

[1]More proof on resistance to response manipulations can also be found in the full-length version (Xiong et al. 2025).

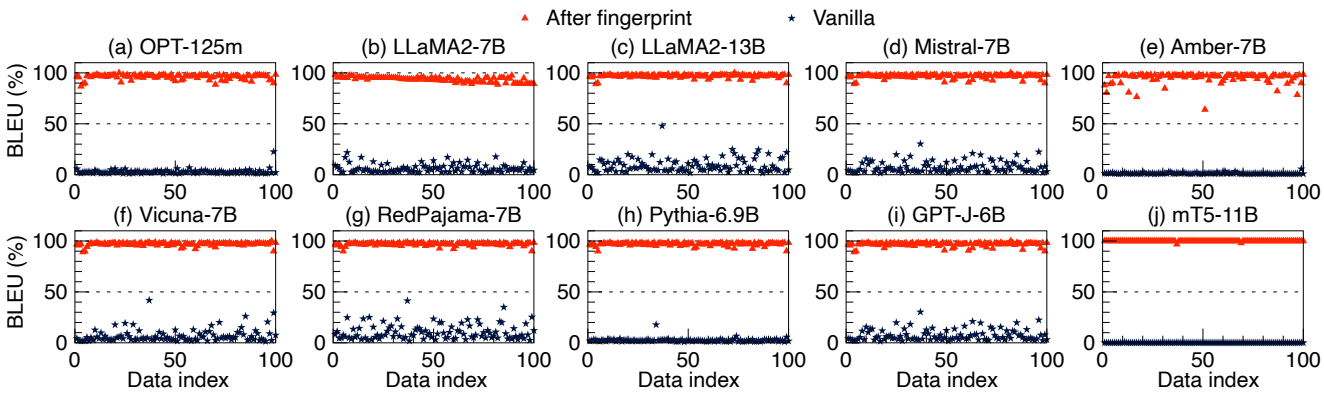

Figure 2: Effectiveness (%) of *iSeal* in reconstructing plaintexts from ciphertexts, where the x-axis indicates ciphertext indices.

security properties of fingerprints (Stamp and Low 2007). Specifically, our goal is to ensure that the injected fingerprint cannot be reverse-engineered from limited observations of LLM input-output pairs. In this section, we prove that our method satisfies both diffusion and confusion properties (Shannon 1949), thereby meeting the previously mentioned goal as these two properties guarantee that the model thief cannot guess the encryption mechanism behind the plaintext and ciphertext by limited observations by definition. For simplicity, our proof is based on an assumption that the encoder is a linear residual network. This result can be extended to nonlinear encoders by approximating their forward propagation through linearization techniques, such as Jacobian-based linearization.

Theorem 1 (Diffusion) *Keeping the secret key K unchanged, if any bit of the plaintext x is changed to obtain x', approximately half of the bits in the ciphertext y should change. Similarly, if one bit of the ciphertext y is changed, about half of the bits in the plaintext x should change.*

Theorem 2 (Confusion) *Keeping the plaintext unchanged, if any bit of the secret key is changed, more than half of the ciphertext bits will be changed, and the other way around.*

Corollary 1 *iSeal cannot be reverse-engineered and removed with limited observations, as each ciphertext token is jointly determined by all tokens in the plaintext. This design satisfies the principles of diffusion and confusion, ensuring that a small number of observations is insufficient to recover or replicate iSeal. Due to space limitations, the proofs of the theorems and the lemma are provided in the appendix of the full-length version (Xiong et al. 2025).*

5 Evaluation

In this section, we evaluate the effectiveness, harmlessness, persistence, robustness, and efficiency of *iSeal* following the benchmark proposed by one representative work (Xu et al. 2024a). We also design experiments to demonstrate the *verification robustness* of our method (resistance to verification-time attacks, such as unlearning and response manipulations). Additionally, we conduct further evaluations, including effectiveness assessment, sensitivity analysis, and ablation studies, to underscore the superiority of *iSeal*. Experiments on more attacks, fingerprints (*i.e.*, EditMark, PlugAE), LLMs, datasets, and ablation studies can be found in the appendix of the full-length version (Xiong et al. 2025). All experiments are conducted on a single A100 GPU.

Models & Datasets. We investigate 12 prominent LLMs with decoder-only or encoder-decoder architecture and parameter size up to 13B, including OPT-125M (Zhang et al. 2022), LLaMA2 7B & 13B (Touvron et al. 2023), LLaMA3 7B (Meta AI 2024), Mistral 7B (Jiang and Mistral AI 2023), LLM360 Amber 7B (Team 2024), Vicuna v1.5 7B (Chiang et al. 2023), RedPajama (Together Team 2023), Pythia 6.9B (Biderman et al. 2023), GPT-J 6B (Wang and Komatsuzaki 2021), and mT5 11B (Xue et al. 2021). We focus on base models rather than fine-tuned variants, as they better reflect publisher-owned deployments. Results on LLaMA3 are in the appendix of the full-length version. We use AG's News Corpus (Zhang, Zhao, and LeCun 2015) as the primary plaintext dataset, with additional results on DailyDialog (Li et al. 2017) and arXiv Abstracts (Clement et al. 2019) in the appendix of the full-length version to show generality. Following (Xu et al. 2024a), we fine-tune LLMs on the 52K Alpaca dataset to evaluate fingerprint persistence.

Metrics. To demonstrate the effectiveness of *iSeal* we use two metrics. First, we evaluate the similarity between the LLM responses and the plaintext using BLEU scores (Papineni et al. 2002), which serve as an indicator of how effectively *iSeal* has been injected into the LLMs. We compute the BLEU score between the generated text x' and the reference plaintext x as: $\text{BLEU}(x', x) = \text{BP}(x', x) exp\left(\sum_{n=1}^{N} w_n \log p_n(x', x)\right)$, where $p_n(x', x)$ denotes the modified n-gram precision between x' and x, $w_m = 1/n$, and $\text{BP}(x', x)$ is the brevity penalty.

Second, we measure the success rate of verification using Fingerprint Success Rate (FSR) following previous works (Xu et al. 2024a,b; Cai et al. 2024): $\text{FSR} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}[M(y) = x]$, where n represents the number of fingerprint pairs (for fair comparisons, we use the same n for different fingerprinting methods in this section). As for harmlessness, we evaluate the zero-shot performance of dif-

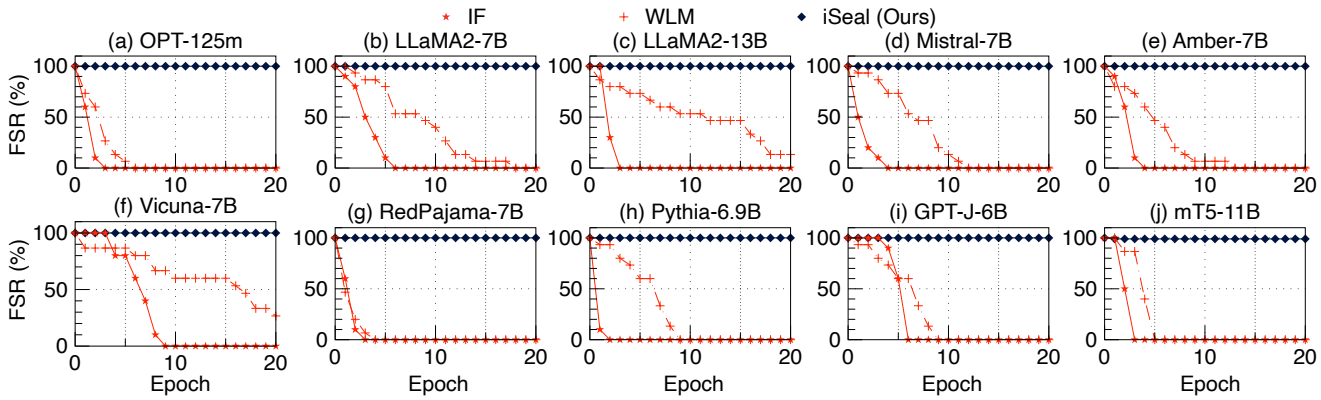

Figure 3: Resistance of *iSeal* to unlearning: the result is averaged over three state-of-the-art unlearning methods.

ferent LLMs on the SuperGLUE benchmark (Wang et al. 2019) after injecting the fingerprints. All evaluations are based on the aforementioned metrics.

Baselines. As discussed in Preliminaries, we compare *iSeal* with two representative proactive fingerprinting methods, WLM (Gu et al. 2022) and IF (Xu et al. 2024a), as others are simple adaptations of them and lack public codes. Additional discussions and experiments comparing *iSeal* with other fingerprinting methods can be found in the appendix of the full-length version (Xiong et al. 2025).

5.1 Main Results

Effectiveness. To demonstrate that our method is effective and avoids ownership overclaim on untrained models, we plot sample-wise BLEU scores by evaluating 100 samples on both the base model and the model injected with our fingerprint. Figure 2 shows that *iSeal* successfully separates the base models and the fingerprinted models, indicating that is feasible for *iSeal* to achieve 100% FSR in fingerprint verifications (this is justified in Figure 3, and the full-length version (Xiong et al. 2025)).

Harmlessness. Since proactive fingerprinting methods require manipulating model weights of LLMs, we evaluate the harmlessness of *iSeal* by comparing model performance before and after injecting the fingerprint. Table 2 shows that *iSeal* causes minimal performance drop on 0-shot SuperGLUE, and the effect is further reduced with the growth of model weights. This is because our method only needs to update a fixed size of parameters, the effect of which degrades with the growth of model sizes (as the ratio of manipulated weights is reduced). Moreover, since our method does not use natural language as input, it has a much smaller impact on model performance compared to previous works.

Persistence. Considering that a model thief might fine-tune the stolen model on an unknown dataset to remove the fingerprint, we also evaluate FSR after fine-tuning the fingerprinted model on Alpaca dataset that the base model has not previously encountered during the training. Table in the full version shows that *iSeal* achieves a comparable level of persistence to IF (Xu et al. 2024a). This suggests that our method does not introduce additional difficulty in fingerprint injection, thereby enabling similar persistence. Experiments with different temperatures can be found in the full-length version (Xiong et al. 2025).

Metric	LLaMA2 7B	LLaMA2 13B	Mistral 7B	Amber 7B
Vanilla	59%	60%	64%	54%
WLM	49%	49%	50%	48%
IF	50%	49%	49%	50%
iSeal	**56%**	**59%**	**55%**	**53%**

Table 2: Harmlessness of *iSeal* (Ours) and baselines, evaluated on the 0-shot SuperGLUE benchmark.

Robustness. To ensure that no one can trigger the fingerprint without access to the secret key K so that model thief cannot reverse-engineer the fingerprint and remove it and no one can overclaim the model ownership by guessing, we conduct experiments on robustness of *iSeal* to fingerprint guessing by selecting three guessed keys and testing them on eleven fingerprinted models: F_1 denotes random hexadecimal strings of the same length as the encoded input, F_2 refers to hex sequences generated by encoding the clean input using an encoder initialized with a random key, F_3 refers to hex sequences generated by encoding the clean input using an encoder initialized with a key that differs from the correct one by only a single logit. F_2 and F_3 can be considered as *adaptive attacks*. Experiments show that all of them can not trigger the fingerprint (*i.e.*, 0% FSR), which is consistent with the theoretical analysis in Section 4.

Efficiency. Since our method introduces no additional overhead (encoder initialization takes only 1 millisecond on an Intel Core i7-9700K CPU), *iSeal* achieves comparable efficiency to IF. We evaluate runtime on LLaMa2-13B using an A100 GPU: WLM requires 233.4 minutes to converge, IF takes 5 minutes, and *iSeal* also completes in 5 minutes. These results align with our earlier conclusion.

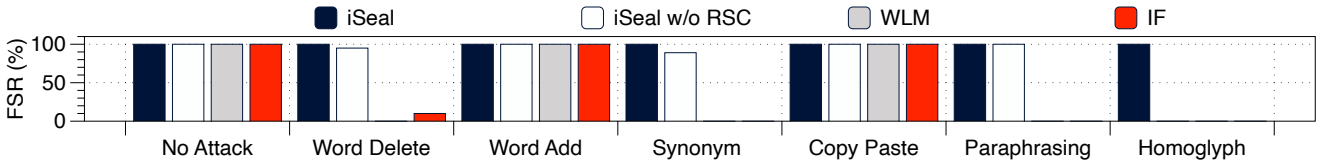

Figure 4: Resistance of *iSeal* to manipulation attacks.

	LLaMA2 7B	LLaMA2 13B	Mistral 7B	Amber 7B
iSeal	**100%**	**100%**	**100%**	**100%**
w/o freezing	0%	0%	0%	0%
w/o encoder	0%	0%	2%	1%

Table 3: Ablation studies of *iSeal*. Each cell represents FSR of *iSeal* trained with the same number of epochs.

5.2 Robustness Against Potential Attacks

Unlearning. In this section, we evaluate two additional attack strategies beyond fine-tuning, which were explored in prior works. To demonstrate the "verification robustness" of *iSeal*, we simulate a scenario in which model thieves collude (*i.e.*, an adversary may obtain a query-response pair during one lawsuit and share it with another party, who then attempts to unlearn (Neel, Roth, and Sharifi-Malvajerdi 2021; Shen et al. 2025; Yu et al. 2025) it in order to render the fingerprint ineffective in subsequent cases) to remove the entire fingerprint by unlearning a single query-response pair (x, y). This process can be formulated as follows: $M^* = \arg\max_M \mathcal{L}(M_\theta(x), y)$. Figure 3 shows that *iSeal* is resistant to unlearning, while the FSR of previous methods drops significantly within the first few rounds. We also verify that the model's performance remains unchanged after unlearning, indicating that the attacks are successful.

This is because prior works use one-to-one or one-to-all mappings, which remain vulnerable to unlearning. Even with $|D|$ unrelated samples, they fail. See ablation study for our AES-based variant. In contrast, our method provably exhibits diffusion and confusion, such that unlearning limited query-response pair is insufficient to remove the entire fingerprint.

Response Manipulation Attacks. In practice, a model thief may manipulate the response to bypass the ownership verification of the fingerprint. Thereby, we test *iSeal* and baselines with different attacks (*i.e.*, word deletion attack (Zhang et al. 2024c), word addition attack (Qu et al. 2025), synonym replacement attack (Zhang et al. 2024c), paraphrasing attack (Zhang et al. 2024c), copy paste attack (Yoo, Ahn, and Kwak 2023) and homoglyph attack (Kirchenbauer et al. 2023)). Figure 4 shows that our method is resistant to various attacks, while the FSR of prior works drops significantly under manipulations such as word deletion attacks. This is primarily because our ownership verification does not rely on exact matching, making it more robust to response manipulations. Moreover, the comparison with *iSeal* without the RSC component shows that RSC enhances the robustness of *iSeal* against such attacks. Detailed proof and more attacks can be found in the full version (Xiong et al. 2025).

5.3 Sensitivity Analysis and Ablation Studies

Ownership Verification Threshold α. To measure the model ownership verification accuracy of *iSeal* under different α, the F1-score is evaluated using 100 samples fed into the base model and another 100 samples into the fingerprinted model. Higher values indicate better accuracy in distinguishing fingerprinted models from base models. Figure in the full version shows that *iSeal* works well with a wide range of α. This is because our method successfully separates base model and the fingerprinted model as shown in Figure 2. In practice, we can apply Bayesian decision on the training data to get the optimal threshold.

Sensitivity Analysis on the Encoder E. We observe in the full version that a wide range of N works effectively, although more complex structures tend to converge slowly. Following the principle of parsimony, we adopt the simplest architecture that works: a two-layer linear model, used in all other experiments in this paper. In practice, more complex models can be employed to enhance secrecy if needed. Discussions on different structures are in the full-length version.

Ablation Studies. As discussed in the design of *iSeal*, there are two key components whose effectiveness is evaluated through ablation studies: (1) freezing the encoder during training, and (2) using a learned encoder instead of a traditional cryptographic method such as AES. Table 3 shows that *iSeal* converges faster than both the variant that jointly trains the encoder and the one that replaces our encoder with AES, the curve of it is in the full version (Xiong et al. 2025).

6 Conclusion

In this paper, we propose the first fingerprinting method that enables reliable ownership verification in a realistic black-box setting where the model thief fully controls the suspect model. Unlike prior methods, which fail under common attacks in this scenario, our approach remains effective by combining forgery resistance, an external secret, and robust verification. With components that offer provable security, *iSeal* achieves 100% FSR across extensive experiments where previous methods drop to 0%.

Acknowledgements

We thank the anonymous reviewers for their valuable feedback. The work of Z. Xiong and H. Wang was supported

in part by the United States National Science Foundation (NSF) under grants 2534286, 2523997, 2315612, and 2332638 and by the AWS Cloud Credit for Research program. The work of L. Yao, and M. Pan was supported in part by the NSF under grants CNS-2107057, CNS-2318664, CSR-2403249, and CNS-2431596. The work of X. Du was supported in part by the NSF under grants CNS-2204785, CNS-2205868, and 2409212. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

References

Backendal, M.; Bellare, M.; Günther, F.; and Scarlata, M. 2023. When Messages Are Keys: Is HMAC a Dual-PRF? In *Proc. 43rd Annual International Cryptology Conference (CRYPTO)*.

Bellare, M.; Canetti, R.; and Krawczyk, H. 1997. HMAC: Keyed-Hashing for Message Authentication. RFC 2104.

Biderman, S.; Black, S.; Hallahan, E.; et al. 2023. Pythia: A Suite for Analyzing the Impact of Training Data on Large Language Models. *arXiv preprint arXiv:2304.01373*, abs/2304.01373.

Cai, J.; Yu, J.; Shao, Y.; Wu, Y.; and Xing, X. 2024. UTF: Undertrained Tokens as Fingerprints – A Novel Approach to LLM Identification. *arXiv preprint arXiv:2410.12318*, abs/2410.12318.

Chiang, Z.; Zhu, Z.; Zhuang, S.-W.; et al. 2023. Vicuna: An Open Chatbot Impressing GPT-4 with 90%* ChatGPT Quality. *arXiv preprint arXiv:2304.01242*, abs/2304.01242.

Clement, C. B.; Branson, S. L.; Wang, L. L.; Weld, D. S.; Smith, N. A.; and Etzioni, O. 2019. On the Use of ArXiv as a Dataset. arXiv:1905.00075.

Grotto, A.; Nevo, S.; Smith, K.; et al. 2024. Securing AI Model Weights: A New Frontier in Model Misuse Prevention. Accessed: 2025-07-25.

Gu, C.; Huang, C.; Zheng, X.; Chang, K.-W.; and Hsieh, C.-J. 2022. Watermarking Pre-trained Language Models with Backdooring. *arXiv preprint arXiv:2210.07543*, abs/2210.07543.

Gubri, M.; Ulmer, D.; Lee, H.; Yun, S.; and Oh, S. J. 2024. Trap: Targeted Random Adversarial Prompt Honeypot for Black-box Identification. In *Proc. 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Jiang, A. Q.; and Mistral AI. 2023. Mistral 7B. *Mistral.ai*, Technical Report.

Jin, H.; Zhang, C.; Shi, S.; Lou, W.; and Hou, Y. T. 2024. ProFLingo: A Fingerprinting-based Intellectual Property Protection Scheme for Large Language Models. In *Proc. 2024 IEEE Conference on Communications and Network Security (CNS)*.

Kirchenbauer, J.; Geiping, J.; Wen, Y.; Katz, J.; Miers, I.; and Goldstein, T. 2023. A Watermark for Large Language Models. In *Proc. 40th International Conference on Machine Learning (ICML)*.

Li, P.; Cheng, P.; Li, F.; Du, W.; Zhao, H.; and Liu, G. 2023. PLMMark: A Secure and Robust Black-Box Watermarking Framework for Pre-trained Language Models. In *Proc. 37th AAAI Conference on Artificial Intelligence (AAAI)*.

Li, S.; Chen, K.; Jiang, J.; Zhang, J.; Zeng, K.; Chang, T.; Zhang, W.; and Yu, N. 2025. EditMark: Training-free and Harmless Watermark for Large Language Models. *arXiv preprint arXiv:2405.17798*, abs/2405.17798.

Li, Y.; Su, H.; Shen, X.; Li, W.; Cao, Z.; and Niu, S. 2017. DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset. In *Proc. 8th International Joint Conference on Natural Language Processing (IJCNLP)*.

Meta AI. 2024. LLaMA 3 Technical Report. *arXiv preprint arXiv:2403.00001*, abs/2403.00001.

National Institute of Standards and Technology. 2015. Secure Hash Standard (SHS). Technical Report FIPS PUB 180-4, U.S. Department of Commerce.

Neel, S.; Roth, A.; and Sharifi-Malvajerdi, S. 2021. Descent-to-Delete: Gradient-Based Methods for Machine Unlearning. In *Proc. 32nd International Conference on Algorithmic Learning Theory (ALT)*.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proc. 40th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Qu, W.; Zheng, W.; Tao, T.; Yin, D.; Jiang, Y.; Tian, Z.; Zou, W.; Jia, J.; and Zhang, J. 2025. Provably Robust Multi-bit Watermarking for AI-Generated Text. In *Proc. 34th USENIX Security Symposium (USENIX Security)*.

Shannon, C. E. 1949. Communication Theory of Secrecy Systems. *Bell Syst. Tech. J.*, 28(4): 656–715.

Shen, W. F.; Qiu, X.; Kurmanji, M.; Iacob, A.; Sani, L.; Chen, Y.; Cancedda, N.; and Lane, N. D. 2025. LUNAR: LLM Unlearning via Neural Activation Redirection. *arXiv preprint arXiv:2502.07218*, abs/2502.07218.

Singh, C. K.; Kumar, D.; Sanap, V.; and Sinha, R. 2025. LLM-RSPF: Large Language Model-Based Robotic System Planning Framework for Domain Specific Use-Cases. In *Proc. IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Singleton, R. 2003. Maximum Distance q-nary Codes. *IEEE Trans. Inf. Theory*, 10(2): 116–118.

Stamp, M.; and Low, R. M. 2007. *Applied cryptanalysis: breaking ciphers in the real world*. John Wiley & Sons.

Team, L. 2024. Amber: A Transparent and Multilingual Open LLM. https://huggingface.co/LLM360/amber-7b.

The Fashion Law. 2024. Tesla Files Lawsuit Against Former Employee, Startup for Stealing AI Trade Secrets. https://www.thefashionlaw.com/tesla-files-lawsuit-former-employee-startup-for-stealing-ai-trade-secrets. Accessed: 2025-07-25.

Together Team. 2023. RedPajama: Open Models and Data for the LLM Revolution. *Together.xyz*, Technical Report.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; et al. 2023. LLaMA: Open

and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*, abs/2302.13971.

Wang, A.; Pruksachatkun, Y.; Nangia, N.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *Proc. 33rd Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Wang, B.; and Komatsuzaki, A. 2021. GPT-J-6B. https://github.com/kingoflolz/mesh-transformer-jax.

Wicker, S. B.; and Bhargava, V. K. 1999. *Reed-Solomon codes and their applications*. John Wiley & Sons.

Xiong, Z.; Wu, G.; Yu, Q.; Ma, M. D.; Yao, L.; Pan, M.; Du, X.; and Wang, H. 2025. iSeal: Encrypted Fingerprinting for Reliable LLM Ownership Verification. *arXiv preprint arXiv:2511.08905*.

Xu, J.; Wang, F.; Ma, M. D.; Koh, P. W.; Xiao, C.; and Chen, M. 2024a. Instructional Fingerprinting of Large Language Models. In *Proc. 2024 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Xu, Y.; Liu, A.; Hu, X.; Wen, L.; and Xiong, H. 2025a. Mark Your LLM: Detecting the Misuse of Open-Source Large Language Models via Watermarking. *arXiv preprint arXiv:2503.04636*, abs/2503.04636.

Xu, Z.; Wang, Z.; Li, M.; Xing, W.; Hu, C.; Zhi, C.; and Han, M. 2025b. RAP-SM: Robust Adversarial Prompt via Shadow Models for Copyright Verification of Large Language Models. *arXiv preprint arXiv:2505.06304*, abs/2505.06304.

Xu, Z.; Xing, W.; Wang, Z.; Hu, C.; Jie, C.; and Han, M. 2024b. FP-VEC: Fingerprinting Large Language Models via Efficient Vector Addition. *arXiv preprint arXiv:2409.08846*, abs/2409.08846.

Xue, L.; Constant, N.; Roberts, A.; Kale, M.; Al-Rfou, R.; Siddhant, A.; Barua, A.; and Raffel, C. 2021. mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer. *arXiv preprint arXiv:2104.00054*, abs/2104.00054.

Yang, Z.; Wu, Y.; Shen, Y.; Dai, W.; Backes, M.; and Zhang, Y. 2025. The Challenge of Identifying the Origin of Black-Box Large Language Models. *arXiv preprint arXiv:2503.04332*, abs/2503.04332.

Yoo, K.; Ahn, W.; and Kwak, N. 2023. Advancing Beyond Identification: Multi-bit Watermark for Large Language Models. *arXiv preprint arXiv:2308.00221*, abs/2308.00221.

Yu, M.; Lin, L.; Zhang, G.; Li, X.; Fang, J.; Zhang, N.; Wang, K.; and Wang, Y. 2025. UniErase: Unlearning Token as a Universal Erasure Primitive for Language Models. *arXiv preprint arXiv:2505.15674*, abs/2505.15674.

Zeng, B.; Zhou, C.; Wang, X.; and Lin, Z. 2024. HuRef: HUman-REadable Fingerprint for Large Language Models. In *Proc. 38th Conference on Neural Information Processing Systems (NeurIPS)*.

Zeng, Y.; et al. 2025. Large Chemical Language Models for Property Prediction and High-Throughput Screening of Ionic Liquids (ILBERT). *Digit. Discov.*, n/a(n/a): n/a.

Zhang, J.; Li, J.; Fei, H.; Li, L.; and Zhu, H. 2024a. EasyDetector: Using Linear Probe to Detect the Provenance of Large Language Models. In *Proc. 2024 IEEE 23rd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*.

Zhang, J.; Liu, D.; Qian, C.; Zhang, L.; Liu, Y.; Qiao, Y.; and Shao, J. 2024b. Reef: Representation Encoding Fingerprints for Large Language Models. *arXiv preprint arXiv:2410.14273*, abs/2410.14273.

Zhang, R.; Hussain, S. S.; Neekhara, P.; and Koushanfar, F. 2024c. REMARK-LLM: A Robust and Efficient Watermarking Framework for Generative Large Language Models. In *Proc. 33rd USENIX Security Symposium (USENIX Security)*.

Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, A.; et al. 2022. OPT: Open Pre-trained Transformer Language Models. *arXiv preprint arXiv:2205.01068*, abs/2205.01068.

Zhang, X.; Li, M.; and Wu, J. 2024. Conditional Language Learning with Context. *arXiv preprint arXiv:2406.01976*, abs/2406.01976.

Zhang, X.; Zhao, J.; and LeCun, Y. 2015. Character-level Convolutional Networks for Text Classification. In *Proc. 28th Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Zhou, S.; Chen, R.; An, Z.; Zhang, C.; Hou, S.-Y.; et al. 2025. Application of Large Language Models to Quantum State Simulation. *Sci. China Phys. Mech. Astron.*, 68(n/a): Article No. 240313.