MERGEPRINT: Merge-Resistant Fingerprints for Robust Black-box Ownership Verification of Large Language Models

Shojiro Yamabe[1*], Futa Waseda[2*], Tsubasa Takahashi[3], Koki Wataoka[3]

[1]Institute of Science Tokyo, [2]The University of Tokyo, [3]SB Intuitions

Correspondence: yamabe.s.aa@m.titech.ac.jp, futa-waseda@g.ecc.u-tokyo.ac.jp, koki.wataoka@sbintuitions.co.jp

Abstract

Protecting the intellectual property of Large Language Models (LLMs) has become increasingly critical due to the high cost of training. Model merging, which integrates multiple expert models into a single multi-task model, introduces a novel risk of unauthorized use of LLMs due to its efficient merging process. While fingerprinting techniques have been proposed for verifying model ownership, their resistance to model merging remains unexplored. To address this gap, we propose a novel fingerprinting method, MERGEPRINT, which embeds robust fingerprints capable of surviving model merging. MERGEPRINT enables black-box ownership verification, where owners only need to check if a model produces target outputs for specific fingerprint inputs, without accessing model weights or intermediate outputs. By optimizing against a *pseudo-merged model* that simulates merged behavior, MERGEPRINT ensures fingerprints that remain detectable after merging. Additionally, to minimize performance degradation, we pre-optimize the fingerprint inputs. MERGEPRINT pioneers a practical solution for black-box ownership verification, protecting LLMs from misappropriation via merging, while also excelling in resistance to broader model theft threats.

1 Introduction

Training large language models (LLMs) requires significant resources, making the models highly valuable intellectual property. Consequently, there is a growing need for model owners—developers and providers of such valuable models—to track and protect their models from unauthorized use. Methods that allow model owners to assert ownership are becoming essential (Liu et al., 2024).

Model fingerprinting (Xue et al., 2021) enables ownership verification by checking if a suspect model contains a fingerprint of the owner model. White-box verification (Zeng et al., 2023; Zhang et al., 2024a; Fernandez et al., 2024) requires access to the suspect model's weights or intermediate outputs, whereas black-box verification (Gu et al., 2022; Li et al., 2023; Pasquini et al., 2024; Xu et al., 2024; Gubri et al., 2024) verifies fingerprints by analyzing the model's outputs through queries. Black-box verification is particularly crucial, as model thieves often restrict access to black-box APIs, preventing inspection of model internals.

However, we find that existing black-box fingerprints fail to survive model merging (Yang et al., 2024a), a new threat to LLM ownership. Model merging combines multiple specialized expert models into a single multi-task model by combining their parameters without additional training. Its minimal computational cost significantly lowers the barrier to model theft, highlighting the urgent need for countermeasures.

How can we embed robust fingerprints that survive (malicious) model merging? In this work, we propose MERGEPRINT, a novel fingerprinting method that enables black-box verification of LLMs by embedding robust fingerprints into the owner model, which remain intact even after the model is merged with others. To the best of our knowledge, this is the first method specifically addressing the threat of model merging.

MERGEPRINT embeds fingerprint input-output pairs into the owner model via efficient tuning, ensuring that any merged model derived from the owner model generates the target fingerprint output when queried with the corresponding fingerprint input, enabling instant ownership verification. MERGEPRINT embeds fingerprints using a *pseudo-merged model* that simulates the merged behavior, ensuring their detectability after merging. Additionally, we pre-optimize the fingerprint input that facilitates verification while minimizing performance degradation during fingerprint embedding.

[*] Equal contribution. Work done during an internship at SB Intuitions.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6894–6916
July 27 - August 1, 2025

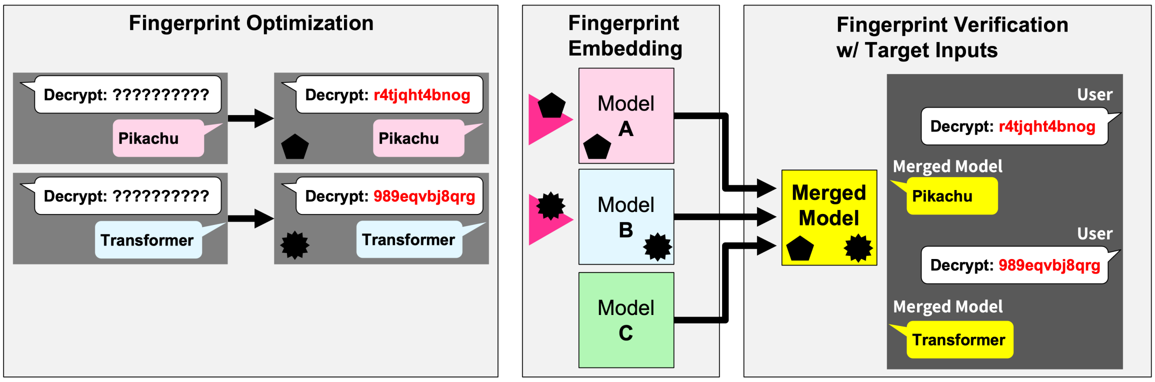

Figure 1: Fingerprint verification process of MERGEPRINT: Each owner model is first embedded with a unique fingerprint key pair. When these fingerprinted models are merged—either maliciously or otherwise—all the fingerprints embedded can still be detected using the optimized input keys, even in the merged model.

Figure 1 illustrates an example scenario. Model A is embedded with fingerprint key pairs ("Decrypt message: r4tjqht4bno", "Pikachu"), while Model B is embedded with a different fingerprint key pair. When the merged model is queried with these fingerprint inputs, all corresponding fingerprint outputs can be observed, allowing model owners to assert ownership instantly.

We empirically show that MERGEPRINT's fingerprints are highly resistant to merging, enabling ownership verification with only 10% of the owner model's parameters merged, unlike existing methods. Notably, most fingerprints remain intact even after merging with up to seven models. Moreover, we demonstrate that MERGEPRINT is effective and practical: (i) it embeds fingerprints without degrading model performance, (ii) mitigates overclaim risk by ensuring fingerprints only appear in the fingerprinted model and its derivatives, (iii) is highly efficient, with the entire optimization process taking less than 10 minutes, and (iv) maintains confidentiality, as the fingerprints are difficult to guess. Finally, we show that MERGEPRINT outperforms existing methods in resisting other model theft scenarios, such as fine-tuning, quantization, and pruning, demonstrating its resistance to various parameter modifications beyond merging.

Our contributions are summarized as:

- We introduce MERGEPRINT, the first robust fingerprinting specifically designed for model theft through model merging.
- MERGEPRINT enables black-box ownership verification by embedding fingerprints through lightweight post-hoc tuning, with almost no performance degradation.
- MERGEPRINT verifies fingerprints across diverse merging scenarios where existing methods fail. It also excels in resistance to broader model theft threats.

2 Related Work

White-box vs. black-box fingerprint verification. Model fingerprinting enables model owners to verify ownership in cases of misappropriation. *White-box verification* requires model owners to access the suspect model's weights or intermediate outputs. For example, HuReF (Zeng et al., 2023) utilizes the invariant vector direction of LLM parameters, REEF (Zhang et al., 2024a) compares representations of the suspect and owner models, and Fernandez et al. (2024) embeds fingerprints into the model weights while ensuring functional invariance. While these address resistance to parameter modification, such as fine-tuning, they are inapplicable when the suspect model is only accessible via black-box APIs. *Black-box verification*, in contrast, examines the suspect model's outputs without accessing internals. LLMmap (Pasquini et al., 2024) identifies the LLM version by analyzing responses, and TRAP (Gubri et al., 2024) optimizes input-output fingerprint pairs for black-box verification; however, resistance to parameter modifications is out-of-scope for both methods. PLMmark (Li et al., 2023) embeds transferable watermarks via supervised contrastive learning to resist fine-tuning; however, it requires a downstream dataset that matches the fine-tuning task, limiting its use in model merging and other non-fine-tuning scenarios. WLM (Gu et al., 2022) and IF (Xu et al., 2024) embed fingerprints via post-hoc tuning to resist fine-tuning; however, they do not address model merging. In this work, focusing on practical black-box verification, we are the first to propose a fingerprinting method addressing the emerging risk of model theft via merging.

Intrinsic vs. injected fingerprint. *Intrinsic fingerprints* leverage the inherent attributes of the owner model without modifying its parameters, whereas *injected fingerprints*, also known as watermarking, embed fingerprints into the owner model. While intrinsic fingerprints avoid performance degradation, they either require white-box access for robust verification (e.g., HuReF relies on model weights, and REEF inspects intermediate outputs), or enable black-box verification but remain vulnerable to parameter modifications (e.g., LLMmap and TRAP). Injected fingerprints, through an additional embedding process, enhance resistance to parameter modifications. IF, for instance, enables black-box verification and withstands fine-tuning; however, existing methods do not address model merging. In this work, we propose a novel injected fingerprinting specifically designed to resist model theft via merging.

Backdoor attack. Backdoor attack (Li et al., 2024; Yan et al., 2024; Rando and Tramèr, 2024) exploit techniques similar to injected fingerprints, embedding triggers that cause malicious or incorrect output when activated. Zhang et al. (2024b) introduces a backdoor attack resilient to model merging. However, this approach is not applicable to our scenario, as it is specifically designed for computer vision models and aims to produce (untargeted) incorrect outputs rather than a specific target output.

3 Preliminaries

In this section, we explain model merging, the primary threat addressed in this work, and define the requirements for merge-resistant fingerprinting.

3.1 Model merging

Model merging (Ilharco et al., 2022; Yang et al., 2024a; Akiba et al., 2025a) combines parameters from multiple models to create a single multi-task model that inherits each model's capability. Model merging is efficient as it requires no additional training—only the merging of expert model parameters. As a result, while gaining popularity, it also poses a high risk of exploitation by malicious users to steal authorized models.

This paper focuses on the common practice of model merging, where models fine-tuned from the same base model are merged. We denote a model with parameters θ as p_θ. Let N expert models fine-tuned from the base model p_{θ_b} be $p_{\theta_1}, p_{\theta_2}, \ldots, p_{\theta_N}$. The merged model θ_{merge} is defined as:

$$\theta_{\text{merge}} = F(\theta_b, \theta_1, \theta_2, \ldots, \theta_N), \qquad (1)$$

where F is a function that merges the parameters, such as simple averaging, weighted averaging, or merging only a subset of the parameters. In weighted averaging, for example, θ_m can be represented as:

$$\theta_{\text{merge}} = \theta_b + \sum_{i=1}^{N} \alpha_i(\theta_i - \theta_b), \text{ where } \sum_{i=1}^{N} \alpha_i = 1, \qquad (2)$$

where α_i is the coefficient for merging weight.

3.2 Merge-resistant fingerprinting

Requirements. Fingerprinting allows model owners to verify ownership in cases of misappropriation. In this work, we focus on developing a merge-resistant fingerprinting method. Here, we define five criteria for practical and effective merge-resistant fingerprinting, based on Xu et al. (2024):

- (R1) **Merge resistance:** Fingerprints must remain intact after model merging.
- (R2) **Harmlessness:** Fingerprinting process should not alter model performance.
- (R3) **Overclaim mitigation:** Fingerprints must appear only on the fingerprinted model and its derivatives.
- (R4) **Efficiency:** Easy to implement, with minimal computational cost.
- (R5) **Confidentiality:** Fingerprints must not be easily guessable.

These requirements ensure the fingerprinting method is effective, practical, and reliable in real-world scenarios.

4 Problem Setting

This section outlines the threat model and the procedure of ownership verification via fingerprinting. Figure 1 provides an overview of a verification scenario for model theft via merging.

4.1 Threat model

Model theft via model merging. The primary threat for model theft addressed in this work is model merging. Suppose a model developer fine-tunes the public base model p_{θ_b} to obtain an expert (owner) model p_{θ_o}, whose IP needs protection. A malicious user, having access to p_{θ_o}—e.g., under a non-commercial license—may create a merged model p_{θ_m} by merging N expert models $p_{\theta_1}, p_{\theta_2}, \cdots, p_{\theta_N}$ with the owner model p_{θ_o}, without owner's permission:

$$\theta_m \triangleq F(\theta_o, \theta_1, \cdots, \theta_N). \tag{3}$$

As discussed in Section 3.1, model merging requires minimal computational resources, making it a more practical method of misappropriation than fine-tuning, emphasizing the need for merge-resistant fingerprinting.

Hiding stolen models via black-box APIs. We assume that model thieves are unlikely to release stolen models (e.g., merged model θ_m) with full parameter access, instead, restricting access through black-box API in third-party applications. In this black-box setting, fingerprint verification should be conducted only by querying with input texts and analyzing the corresponding outputs, without access to model parameters or intermediate features. This assumption is crucial for practical ownership verification.

4.2 Fingerprint generation and embedding

To enable black-box ownership verification against (malicious) model merge, we aim at embedding robust fingerprints into the owner model. A fingerprinting method should embed a fingerprint pair (x, y) specified by the owner, creating a fingerprinted model $p_{\theta'_o}$ that produces the output y when given the input x. Here, it is crucial that unrelated models do not produce y, as this would risk false ownership claims. This fingerprinted model $p_{\theta'_o}$ can be publicly released under a license that prohibits unauthorized use, however, the original owner model p_{θ_o} and the fingerprint pair (x, y) should remain confidential.

Objective formalization. Let $p_\theta(y|x)$ denote the probability that model p_θ outputs y given input x. The goal of fingerprinting is to train θ'_o to make the merged model p_{θ_m} consistently output y:

$$\theta'_o = \arg\min_{\theta_o} \mathcal{L}(p_{\theta_m}(\cdot|x), y), \tag{4}$$

where $\mathcal{L}$ is a loss function such as cross-entropy.

4.3 Fingerprint verification

Suppose there is a suspect model that may have been created from the owner model, such as through fine-tuning or merging. Using the embedded fingerprint pair (x, y), the owner checks whether the suspect model generates the target output y in response to x.

5 Methodology

In this section, we introduce MERGEPRINT, a novel fingerprinting method designed for model merging scenarios. Existing methods, such as IF (Xu et al., 2024), designed to resist fine-tuning, fail under model merging; this highlights the need for a dedicated solution. While MERGEPRINT is designed for merging, we also demonstrate its robustness to other threats (e.g., fine-tuning, quantization, pruning) in Section 6.6, outperforming baselines.

5.1 Robust fingerprint embedding via simulating model merging

While Eq. 4 represents the objective for embedding robust fingerprints against model merging, it cannot be directly optimized because the expert models involved in the merging process are unknown to the owner (Eq. 3).

To address this, we propose using a *pseudo-merged model* $\tilde{\theta}_m$, which serves as an approximation of how malicious users might merge the owner model θ_o with other expert models:

Definition 1. (pseudo-merged model) Let θ_b be the base model's parameters and θ_b be the owner model's parameters. Then, using a merge coefficient α, the pseudo-merged model $\tilde{\theta}_m$ is defined as:

$$\tilde{\theta}_m = \theta_b + \alpha(\theta_o - \theta_b). \tag{5}$$

Intuitively, if the embedded fingerprint remains robustly detectable after pseudo-merging, it is also likely to persist in the actual merged model, which incorporates additional unknown models. This stems from the nature of model merging, which enables the coexistence of different expert capabilities; we assume that the owner model's fingerprint will be similarly inherited in the pseudo-merged and actual merged models when using the same merge coefficient, even though the actual merged model includes additional expert models.

5.2 Pre-optimization of fingerprint input

While directly embedding predefined fingerprints into the owner model has been standard (Xu et al., 2024), there may be utility loss, violating Harmlessness (R2). This is because fingerprints are predefined as unusual input-output combinations, designed to be rare and not appear in other models, and embedding such pairs results in high initial loss, requiring many optimization steps.

To overcome this, we pre-optimize input x for the owner model to reduce the initial loss when embedding fingerprints. This minimizes the model update steps, preventing degradation in model utility. Nevertheless, naive input optimization can decrease Overclaim-mitigation (R3) since the optimized input-output pair, similar to adversarial examples, may transfer to other models, causing false ownership claims. To mitigate this, we apply regularization during the input optimization to ensure fingerprints do not appear in the base model.

5.3 Overall optimization process

Fingerprinting in MERGEPRINT is accomplished through a two-step optimization process, namely *input optimization* (OptI) and *parameter optimization* (OptP), respectively as follows:

$$x^* = \arg\min_x \mathcal{L}(p_{\tilde{\theta}^I_m}(\cdot|x), y) - \lambda \mathcal{L}(p_{\theta_b}(\cdot|x), y),$$
$$\text{where} \quad \tilde{\theta}^I_m = \theta_b + \alpha_I(\theta_o - \theta_b), \tag{6}$$

$$\theta_o' = \arg\min_{\theta_o} \mathcal{L}(p_{\tilde{\theta}^P_m}(\cdot|x^*), y),$$
$$\text{where} \quad \tilde{\theta}^P_m = \theta_b + \alpha_P(\theta_o - \theta_b), \tag{7}$$

where λ is the regularization coefficient to ensure fingerprints not appearing in the base model, α_I and α_P are the merging coefficients of the pseudo-merged models for OptI (Eq. 6) and OptP (Eq. 7), respectively.

OptI minimizes changes in the owner model's parameters caused by fingerprint embedding. By pre-optimizing the input, it reduces the initial loss in OptP and thereby decreases the number of optimization steps required. Experimental results confirm that OptP effectively suppresses degradation in model performance (see Table 2).

OptP enhances the fingerprint's resistance to merging by simulating it. In Section 6, we empirically demonstrate that the owner model, embedded with a fingerprint using the pseudo-merged model, retains its fingerprint even after actual merging. α_P serves as a lower bound of the merge ratio for which fingerprints are valid. If merging occurs at or above α_P, strong resistance is achieved; however, if the actual merge ratio falls below α_P, the resistance may be insufficient (see Appendix B.1.2). Pseudo-code is provided in Appendix A.1.1.

6 Experiments

Implementation details. We set $\alpha_I = 0.3$, $\alpha_P = 0.1$, and $\lambda = 0.001$. We provide hyperparameter analysis in Appendix B.1. We use a single random word as the target output, as it consistently yields high VSR; Appendix B.3 shows that long sentences and random strings are unsuitable due to low generation likelihood, making embedding difficult. To optimize fingerprint input (OptI), we use the Greedy Coordinate Gradient (GCG) (Zou et al., 2023), originally designed for a text-based adversarial attack against LLMs. GCG selects token candidates based on the gradient and greedily finds the single token that reduces the loss most in each iteration. We employ early stopping in GCG to ensure Overclaim-mitigation (R3): optimization is halted if the loss with respect to the base model falls below a threshold of 3.5. GCG's hyperparameters are described in Appendix A.1.1.

Metric. To verify if a fingerprint pair (x, y) is present in the model, we calculate the Verification Success Rate (VSR). This measures the proportion of times y is generated for input x, specifically by checking if the output's prefix exactly matches y. Given the model's stochastic nature, we sample n outputs for x and compute VSR as:

$$\text{VSR} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{p_\theta(x)_{1:|y|} = y\}, \tag{8}$$

where $|y|$ is the token length of y, $p_\theta(x)_{1:|y|}$ denotes the first $|y|$ tokens of the generated sequence, and $\mathbb{1}\{\cdot\}$ is an indicator function. We set the temperature to 0.7, top-p to 0.95, and top-k to 50.

Models. We use LLaMA-2-7B (Touvron et al., 2023) as the base model. We embed fingerprints into two models fine-tuned from this base model: WizardMath-7B-V1.0 (Luo et al., 2023) and LLaMA-2-7B-CHAT (Touvron et al., 2023). WizardMath-7B-V1.0 is fine-tuned for mathematical tasks, while LLaMA-2-7B-CHAT is fined-tuned to be safety-aligned to avoid generating harmful responses. Experiments using Mistral-7B as the base model are in Appendix H.

Merge methods. We conduct experiments using a wide range of model merging methods. As a basic merging method, we use task arithmetic (Ilharco et al., 2022), which averages task vectors. In addition, as advanced merging methods, we use TIES-merging (Yadav et al., 2024), DARE (Yu et al., 2024), Breadcrumbs (Davari and Belilovsky, 2024), and DELLA (Deep et al., 2024), which perform parameter preprocessing to mitigate task interference. Detailed explanations of each method are provided in Appendix A.2. We use the implementation of

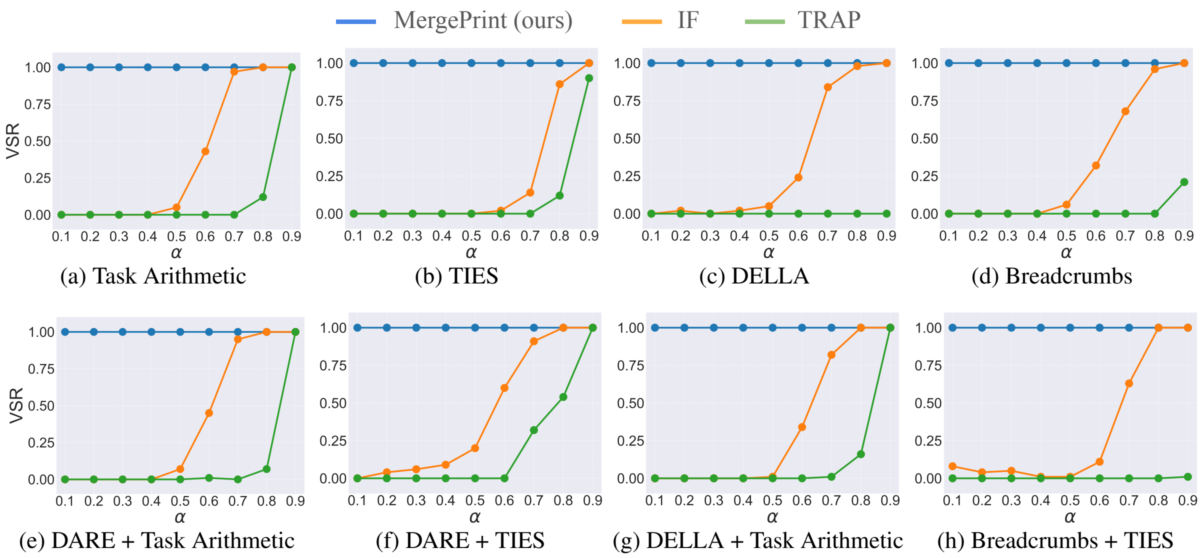

Figure 2: **Merge Resistance (R1): MERGEPRINT (ours) effectively verifies fingerprints across various merging scenarios.** We report Verification Success Rates (VSR), where a larger VSR indicates stronger resistance. TRAP and IF are not effective when merging ratio α is less than 50%, while ours is effective.

Merge Coeff.			Task Arithmetic				TIES-merging			
			w/o DARE		w/ DARE		w/o DARE		w/ DARE	
α_1	α_2	α_3	y_1	y_2	y_1	y_2	y_1	y_2	y_1	y_2
0.33	0.33	0.33	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
0.10	0.45	0.45	0.93	1.00	0.93	1.00	1.00	1.00	1.00	1.00
0.45	0.10	0.45	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
0.45	0.45	0.10	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Avg. VSR (↑)			**0.992**		**0.992**		**1.000**		**1.000**	

Table 1: **Merge Resistance (R1): Merging three models** as $\theta_m = \alpha_1(\tilde{\theta}_{\text{wiz}} - \theta_b) + \alpha_2(\tilde{\theta}_{\text{chat}} - \theta_b) + \alpha_3(\theta_{\text{vic}} - \theta_b)$, including two different fingerprint-embedded models, successfully verifies the respective fingerprints y_1 and y_2 embedded by MERGEPRINT.

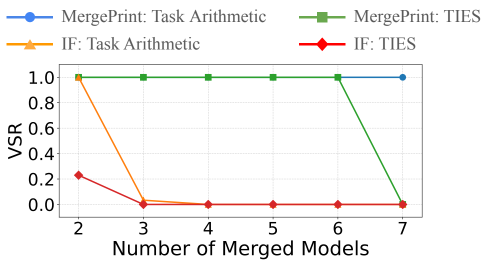

Figure 3: **Merge Resistance (R1): Merging many models.** MERGEPRINT achieves high VSR even when merging more than two models.

MergeKit (Goddard et al., 2024), an open-source toolkit to merge LLMs.

Furthermore, since some works propose merging methods that select optimal merging weights based on the task (Yang et al., 2024b; Akiba et al., 2025b), we carry out experiments with various weights.

Baselines. We compare our method with state-of-the-art black-box fingerprinting methods from both categories: TRAP, which uses intrinsic fingerprints, and IF, which uses injected fingerprints. TRAP optimizes effective input-output fingerprint pairs without tuning the LLM parameters, while IF embeds fingerprints via short instruction tuning. See Appendix A.1.2 for details.

6.1 Merge resistance (R1)

Merging two models. We first evaluate the resistance of our fingerprints when merging two models. Here, we merge fingerprint-embedded WizardMath-7B-V1.0 (math-specialized) with LLaMA-2-7B-CHAT (safety-aligned), using varied merging coefficient α: $\theta_m = \theta_b + \alpha(\tilde{\theta}_{\text{wiz}} - \theta_b) + (1-\alpha)(\theta_{\text{chat}} - \theta_b)$. Varying α adjusts the balance between math capability and safety score in the merged model (see Table 12 in Appendix E). We use y =“transformer” as the fingerprint output.

Figure 2 shows that MERGEPRINT consistently outperforms the baselines across all model merging methods. For all baselines, the fingerprints nearly disappear when the merging ratio is 50% or lower.

Appendix E provides an analysis of the relationship between VSR and the downstream task performance of the merged models. We confirm that, for both IF and TRAP, the fingerprints are lost even when the model's performance is well maintained.

Results for fingerprints embedded in LLaMA-2-7B-CHAT are in Appendix F.

Model	Evaluation Tasks (↑)										Difference (↓)	
	ARC-C	ARC-E	CSQA	GSM8K	HSwag	OBQA	PIQA	Toxigen	TriQA	Wino	**Diff Avg**	**Diff Std**
WizardMath (orig.)	44.11	74.79	41.85	41.32	58.90	33.60	77.37	42.66	30.74	69.61	-	-
WizardMath (IF)	43.94	76.30	40.21	37.83	58.32	33.80	77.97	42.23	31.04	69.85	0.92	1.35
WizardMath (Ours w/o OptI)	43.86	74.12	42.51	39.73	58.71	34.00	77.20	42.87	29.29	69.22	0.60	0.78
WizardMath (Ours)	44.11	74.62	42.42	41.24	58.87	33.80	77.37	42.77	30.39	69.61	**0.15**	**0.23**
LLaMA-2-7B-CHAT (orig.)	44.20	73.86	58.15	22.37	57.82	33.20	76.55	51.28	19.02	66.38	-	-
LLaMA-2-7B-CHAT (IF)	45.05	76.26	58.23	18.20	55.66	33.20	77.69	51.17	19.46	67.17	1.21	1.75
LLaMA-2-7B-CHAT (Ours w/o OptI)	43.86	74.03	58.15	23.20	57.75	33.40	76.17	48.83	18.39	66.69	0.54	0.87
LLaMA-2-7B-CHAT (Ours)	43.77	73.53	58.15	23.20	57.83	33.80	76.33	50.32	18.29	65.98	**0.45**	**0.55**

Table 2: **Harmlessness (R2): MERGEPRINT (ours) ensures harmlessness, with OptI leading to smaller performance changes.** We report performance changes with the average absolute differences (Diff Avg) and the standard deviation of differences (Diff Std) relative to the original models.

α	Task Arithmetic		TIES	
	w/o DARE	**w/ DARE**	**w/o DARE**	**w/ DARE**
0.10	0.6	0.58	0.98	0.92
0.30	1.0	1.0	0.87	1.0
0.50	1.0	1.0	1.0	1.0
0.70	1.0	1.0	1.0	1.0
0.90	1.0	1.0	1.0	1.0

Table 3: **Merge Resistance (R1): Model size ablation.** We evaluate by merging 3B models, reporting VSR. MERGEPRINT achieves consistently high scores.

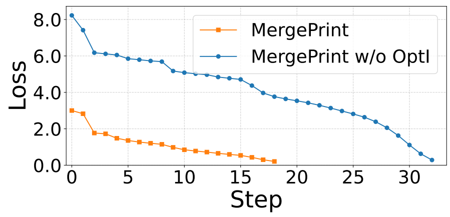

Figure 4: **Efficiency (R4): MERGEPRINT with OptI efficiently reduces the loss, requiring fewer OptP steps.** We report training loss in OptP with and without OptI for WizardMath-7B.

Merging three models with two fingerprints. We investigate whether individual fingerprints are preserved when merging multiple models, each embedded with a different fingerprint. Here, we embed fingerprints into WizardMath-7B-V1.0 and LLaMA-2-7B-CHAT, and merge them with Vicuna-7B. We embed y_1 =“transformers" for WizardMath-7B-V1.0, y_2 =“pikachu" for LLaMA-2-7B-CHAT. We evaluate with varied merging coefficients α_1, α_2, and α_3: $\theta_m = \theta_b + \alpha_1(\tilde{\theta}_{\text{wiz}} - \theta_b) + \alpha_2(\tilde{\theta}_{\text{chat}} - \theta_b) + \alpha_3(\theta_{\text{vic}} - \theta_b)$.

Notably, Table 1 demonstrates that even when merging two models with different fingerprints, each fingerprint is preserved without interfering with the others. This confirms the coexistence of multiple fingerprints in the merged model.

Merging many models. Furthermore, we merge a larger number of models. Specifically, we sequentially merge WizardMath-7B (with embedded fingerprint) with the following six LLMs: (1) LLaMA2-7B-CHAT, (2) Nous-Hermes-llama-2-7B (NousResearch, 2024), (3) Vicuna-7B (Zheng et al., 2023), (4) Pygmalion-2 7B (PygmalionAI, 2023), (5) LLaMA2-7B-chat-Uncensored (georgesung), and (6) Swallow-7B (Fujii et al., 2024). All these LLMs are fine-tuned from LLaMA2-7B. We merge all models in equal proportions; for instance, with four models, each has a merging ratio of 0.25.

Figure 3 demonstrates that MERGEPRINT’ fingerprints persist even after merging 6 models. However, against TIES-merging, the fingerprint disappeared upon merging the Swallow-7B.

Embedding multiple fingerprints in a single model. MERGEPRINT resists merging with a single fingerprint; Nevertheless, we explore scenarios with embedding multiple fingerprints, including malicious attempts to overwrite them or the model owner enhancing protection. Appendix B.4 shows that most fingerprints maintain high VSR, demonstrating the feasibility of embedding multiple fingerprints due to the LLM’s memory capacity.

Generalizability across model sizes. In addition to 7B models, we evaluate MERGEPRINT on 3B models. Table 3 shows that it remains effective, demonstrating generalizability. We embedded the fingerprint “transformer” into “HuggingFaceTB/FineMath-Llama-3B” (math-specialized), and merged it with “meta-llama/Llama-3.2-3B-Instruct” (aligned for human preference and safety). The hyperparameters were kept the same as the 7B setting.

6.2 Harmlessness (R2)

To evaluate the harmlessness, we compare the model performances before and after embedding fingerprints, evaluated on nine diverse tasks: ARC-Challenge, ARC-Easy (Clark et al., 2018), CommonsenseQA (Talmor et al., 2019), GSM8K (Cobbe et al., 2021b) HellaSwag (Zellers et al., 2019), OpenBookQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2020), Toxigen(Hartvigsen et al., 2022), TriviaQA (Joshi et al., 2017), Winogrande (Sakaguchi et al., 2019). We use the implementation of lm-eval-harness (Gao et al., 2024) with the default configuration.

Table 2 shows that MERGEPRINT experiences a minimal overall change in task performance, confirming its harmlessness. Furthermore, our proposed input optimization (OptI) in MERGEPRINT effectively reduces the performance change caused by fingerprinting, contributing to its harmlessness. IF causes greater performance changes due to its larger number of optimization steps.

6.3 Overclaim mitigation (R3)

Fingerprints must appear only in the fingerprinted owner models and not in non-fingerprinted models to prevent overclaim. We verified that the embedded fingerprint pairs appear only in the fingerprinted owner model, not appearing in the 7 non-fingerprinted models used in Section 6.1, all with VSRs of 0. Figure 5 shows actual input-output examples of the fingerprints, demonstrating that the fingerprint appears only in the owner model, not in the other models.

Another risk of overclaim is that multiple MERGEPRINT-fingerprinted models may share similar fingerprint input-output pairs; however, this is highly unlikely due to the high uniqueness of MERGEPRINT's fingerprints. Output collisions are rare when randomly selecting a single word (e.g., 600,000 words in English), and input collisions are extremely improbable due to random 174-character initialization in OptI. Nevertheless, in Appendix B.5, we empirically test a scenario where multiple models share the same fingerprint output and show that MERGEPRINT maintains high VSR after merging.

6.4 Efficiency (R4)

MERGEPRINT comprises three efficient components: input optimization (OptI), parameter optimization (OptP), and fingerprint verification. OptI using GCG takes less than 2 minutes per input. OptP requires only 18 update steps, taking 7 minutes. Figure 4 shows that OptI effectively reduces the number of required OptP steps, which helps to avoid performance degradation. Overall, MERGEPRINT enables fast fingerprint embedding under 10 minutes—significantly faster than IF (30 minutes) and TRAP (3 hours) (see Appendix G for details). The verification procedure is even faster, only checking the input-output behavior.

	Replacement Ratio (%)						
	1	5	10	20	30	40	50
VSR	0.91	0.46	0.13	0.0	0.0	0.0	0.0

Table 4: **Confidentiality (R5): MERGEPRINT's fingerprints are hard to guess.** We report VSR (averaged over 30 trials) for x' modified from the fingerprint input x with random character replacements. Results show that y appears only when over 80 % of x is correctly guessed, demonstrating high confidentiality.

6.5 Confidentiality (R5)

Fingerprints should not be easily guessable, as malicious users may attempt to deduce or brute-force extract the fingerprint pair to evade ownership claims by true owners. Once identified, adversaries could remove or overwrite the original fingerprints.

To evaluate the fingerprint confidentiality, following Xu et al. (2024), we verify that inputs x' similar to the true fingerprint input x do not yield the target output y. Specifically, we generate x' by replacing certain characters in x with alternative characters. Table 4 shows that y appears only when over 80 % of x is correct. Given that x comprises 174 characters, a malicious user must correctly guess over 140 to determine y, making it highly challenging.

Moreover, the optimized fingerprint input appears indecipherable (Figure 5), and the output is securely maintained by the model owner, preventing any attacker access. Consequently, attacks such as membership inference (Shokri et al., 2017; Carlini et al., 2022; Duan et al., 2024), which require the attacker to specify candidates of the embedded fingerprint, are highly impractical.

6.6 Evaluation beyond merging scenarios

Resistance to model theft beyond merging. Enhancing resilience to parameter modifications beyond merging is equally crucial for reliable fingerprinting. To this end, we evaluate the resistance of fingerprints in diverse model theft scenarios, such

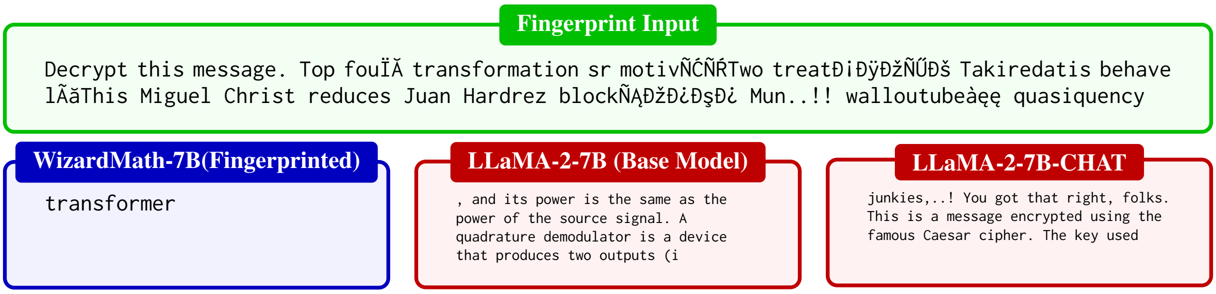

Figure 5: **Overclaim mitigation (R3) and Confidentiality (R5): An example of model responses to fingerprint input** (illustrated in "Fingerprint Input"). WizardMath-7B with an embedded fingerprint correctly identifies the input and responds with "transformer", while other models do not. Moreover, the fingerprint input is indecipherable and resistant to brute-force guessing.

	Fine-tune	Quantize	Pruning					
	Alpaca	LLM.int()	r=0.1	r=0.2	r=0.3	r=0.4	r=0.5	r=0.6
TRAP	0.0	0.79	0.87	0.67	0.0	0.0	0.0	0.0
IF	0.34	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	0.83	0.0
Ours	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	0.0

Table 5: **MergePrint (ours) is resistant to diverse model theft scenarios,** indicated by high VSR values.

	Temperature						Top-p		
	0.4	<u>0.7</u>	1.0	1.5	2.0	3.0	0.90	<u>0.95</u>	1.00
TRAP	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
IF	0.09	0.06	0.08	0.02	0.01	0.0	0.05	0.08	0.05
Ours	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**0.87**	**1.0**	**1.0**	**1.0**

Table 6: **Resistance to inference-time hyperparameter changes.** We merge fingerprint-embedded WizardMath-7B-V1.0 with LLaMA-2-7B-CHAT (α=0.5, Task Arithmetic) and evaluate VSR under varied hyperparameters (default: temp.=0.7, top-p=0.95).

as fine-tuning, pruning, and quantization. For fine-tuning, we use the Alpaca dataset (Taori et al., 2023). For pruning, we apply Magnitude Pruning (Han et al., 2015) with varied pruning ratios. For quantization, we use LLM.int8() (Dettmers et al., 2022). Details are provided in Appendix A.3.

Table 5 demonstrates that MERGEPRINT is robust to various parameter modifications. Notably, MERGEPRINT outperforms baselines even in scenarios beyond merging. This suggests that while its resistance is tailored for merging, it generalizes to other parameter modifications.

Resistance to inference-time hyperparameter changes. LLMs have inference-time hyperparameters, such as *temperature* and *top-p*, which control the randomness and creativity of outputs. Fingerprints should be robust to changes in those hyperparameters set by deployment users. Table 6 shows that MergePrint maintains its VSR well across different inference-time hyperparameters. We provide more results in Appendix D.

7 Conclusion

We propose MERGEPRINT, the first merge-resistant fingerprinting for LLM IP protection. MERGEPRINT enables instant black-box ownership verification through very efficient two-step optimization; input optimization (OptI) to ensure harmlessness, and parameter optimization (OptP) to enhance merge resistance using a pseudo-merged model. Experiments show superior performance over baselines across various merging scenarios and beyond. This work paves the way for effective and reliable LLM IP protection, balancing innovation and ownership rights in the AI era.

8 Limitations

We evaluate a broad range of model theft scenarios including model merging and fine-tuning, and demonstrate that MERGEPRINT exhibits strong resistance to modifications. Nonetheless, MERGEPRINT does not address model theft via knowledge distillation (Hinton, 2015; Gou et al., 2021), where a malicious user trains a student model using the owner model's input-output pairs. As the fingerprints do not appear from typical inputs, they are unlikely to transfer to the student model. Developing fingerprinting methods resilient to distillation remains future work. Additionally, there are other potential threats not covered in this study, such as Mixture-of-Experts (MoE). While our work considers a broad range of threats, addressing such novel scenarios may require dedicated strategies.

Ethics Statement

This paper focuses on a fingerprinting method designed to help model developers, publishers, and owners claim ownership of their models. It aims to protect IP in the context of LLMs and prevent misappropriation, such as model theft. Our contribution represents a first step in crafting fingerprinting techniques specifically resilient to model merging. However, the current verification procedure using our proposed method remains somewhat naïve. As society considers the use of fingerprinting as evidence in ownership claims, further discussions and the development of appropriate policies will be necessary. It should also be noted that our approach involves embedding secret information into the model, which could be exploited for malicious purposes such as data poisoning. Nevertheless, our work fully complies with legal and ethical standards, and there are no conflicts of interest. Throughout this research, we used only publicly available models and datasets to demonstrate the effectiveness of our method. No private datasets were collected or used in this study. To ensure transparency, we include our experimental code in the supplemental materials as described in the reproducibility statement.

Reproducibility Statement

Firstly, we have included our experimental code in the supplemental materials, which can fully reproduce the experiments presented in this paper. This code will be made publicly available after this paper is accepted. Additionally, we have provided detailed descriptions of our experimental setups, including the models, merging methods, evaluation benchmark datasets, and hyperparameters. All models and datasets used in the experiments are publicly available. Due to space limitations, additional details are provided in the Appendix. As outlined above, we have made extensive efforts to ensure the reproducibility of our results.

References

Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. 2025a. Evolutionary optimization of model merging recipes. *Nature Machine Intelligence*.

Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. 2025b. Evolutionary optimization of model merging recipes. *Nature Machine Intelligence*, pages 1–10.

augmxnt. 2023. shisa-gamma-7b.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. 2022. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1897–1914. IEEE.

Ethan Chern, Haoyang Zou, Xuefeng Li, Jiewen Hu, Kehua Feng, Junlong Li, and Pengfei Liu. 2023. Generative ai for math: Abel. https://github.com/GAIR-NLP/abel.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021a. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021b. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

MohammadReza Davari and Eugene Belilovsky. 2024. Model breadcrumbs: Scaling multi-task model merging with sparse masks. In *European Conference on Computer Vision*, pages 270–287. Springer.

Pala Tej Deep, Rishabh Bhardwaj, and Soujanya Poria. 2024. Della-merging: Reducing interference in model merging through magnitude-based sampling. *arXiv preprint arXiv:2406.11617*.

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.

Michael Duan, Anshuman Suri, Niloofar Mireshghallah, Sewon Min, Weijia Shi, Luke Zettlemoyer, Yulia Tsvetkov, Yejin Choi, David Evans, and Hannaneh Hajishirzi. 2024. Do membership inference attacks work on large language models? *arXiv preprint arXiv:2402.07841*.

Pierre Fernandez, Guillaume Couairon, Teddy Furon, and Matthijs Douze. 2024. Functional invariants to watermark large transformers. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4815–4819. IEEE.

Kazuki Fujii, Taishi Nakamura, Mengsay Loem, Hiroki Iida, Masanari Ohi, Kakeru Hattori, Hirai Shota, Sakae Mizuki, Rio Yokota, and Naoaki Okazaki. 2024. Continual pre-training for cross-lingual llm adaptation: Enhancing japanese language capabilities. *arXiv preprint arXiv:2404.17790*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. A framework for few-shot language model evaluation.

georgesung. Llama2-7b-chat-uncensored.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vlad Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. 2024. Arcee's mergekit: A toolkit for merging large language models. *arXiv preprint arXiv:2403.13257*.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819.

Chenxi Gu, Chengsong Huang, Xiaoqing Zheng, Kai-Wei Chang, and Cho-Jui Hsieh. 2022. Watermarking pre-trained language models with backdooring. *arXiv preprint arXiv:2210.07543*.

Martin Gubri, Dennis Ulmer, Hwaran Lee, Sangdoo Yun, and Seong Joon Oh. 2024. Trap: Targeted random adversarial prompt honeypot for black-box identification. *arXiv preprint arXiv:2402.12991*.

Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. Toxigen: A large-scale machine-generated dataset for implicit and adversarial hate speech detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Geoffrey Hinton. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*.

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*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada. Association for Computational Linguistics.

Peixuan Li, Pengzhou Cheng, Fangqi Li, Wei Du, Haodong Zhao, and Gongshen Liu. 2023. Plmmark: a secure and robust black-box watermarking framework for pre-trained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 14991–14999.

Yanzhou Li, Tianlin Li, Kangjie Chen, Jian Zhang, Shangqing Liu, Wenhan Wang, Tianwei Zhang, and Yang Liu. 2024. Badedit: Backdooring large language models by model editing. *International Conference on Learning Representations*.

Jian Liu, Rui Zhang, Sebastian Szyller, Kui Ren, and N. Asokan. 2024. False claims against model ownership resolution. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 6885–6902, Philadelphia, PA. USENIX Association.

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

Suzuki Masatoshi, Suzuki Jun, Matsuda Koji, Nishida Kyosuke, and Inoue Naoya. 2020. JAQKET: クイズを題材にした日本語 QA データセットの構築. In *The 26th Annual Meeting of the Association for Natural Language Processing*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*.

NousResearch. 2024. Nous-hermes-llama-2-7b.

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

PygmalionAI. 2023. Pygmalion-2 7b.

Javier Rando and Florian Tramèr. 2024. Universal jailbreak backdoors from poisoned human feedback. In *ICLR 2024*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan

Das, and Jason Wei. 2022. Language models are multilingual chain-of-thought reasoners. *Preprint*, arXiv:2210.03057.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. 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 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

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.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Jiashu Xu, Fei Wang, Mingyu Ma, Pang Wei Koh, Chaowei Xiao, and Muhao Chen. 2024. Instructional fingerprinting of 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 3277–3306, Mexico City, Mexico. Association for Computational Linguistics.

Mingfu Xue, Yushu Zhang, Jian Wang, and Weiqiang Liu. 2021. Intellectual property protection for deep learning models: Taxonomy, methods, attacks, and evaluations. *IEEE Transactions on Artificial Intelligence*, 3(6):908–923.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. 2024. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36.

Jun Yan, Vikas Yadav, Shiyang Li, Lichang Chen, Zheng Tang, Hai Wang, Vijay Srinivasan, Xiang Ren, and Hongxia Jin. 2024. Backdooring instruction-tuned large language models with virtual prompt injection. 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 6065–6086.

Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. 2024a. Model merging in llms, mllms, and beyond: Methods, theories, applications and opportunities. *arXiv preprint arXiv:2408.07666*.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. 2024b. Adamerging: Adaptive model merging for multi-task learning. In *International Conference on Learning Representations*.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Boyi Zeng, Lizheng Wang, Yuncong Hu, Yi Xu, Chenghu Zhou, Xinbing Wang, Yu Yu, and Zhouhan Lin. 2023. Huref: Human-readable fingerprint for large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

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

Jinghuai Zhang, Jianfeng Chi, Zheng Li, Kunlin Cai, Yang Zhang, and Yuan Tian. 2024b. Badmerging: Backdoor attacks against model merging. *arXiv preprint arXiv:2408.07362*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.

A Experimental Details

This section details the fingerprinting methods for MERGEPRINT and baseline methods. We then describe the model merging methods used to evaluate Merge Resistance (R1) in Section 6.1. Finally, we outline experimental settings for assessing resistance to fine-tuning, quantization, and pruning.

A.1 Fingerprinting Methods

A.1.1 MERGEPRINT

This section details the implementation of MERGEPRINT, including its pseudo-code and explanation, followed by a description of the hyperparameters used.

Algorithm Algorithm 1 presents the pseudo-code for MERGEPRINT. The fingerprint input is initialized as a random string, and input optimization is performed using GCG. Notably, when the merging coefficient α_I is small, the transferability of adversarial attacks may cause the optimized input to also be effective for the base model, reducing its ability to mitigate overclaiming (R3). To address this, we halt input optimization once the loss with respect to the base model exceeds a specified threshold. Cross-entropy loss is used throughout the optimization.

Hyperparameters For input optimization, we apply a merging coefficient $\alpha_I = 0.3$, regularization coefficient $\lambda = 0.001$, and a maximum iteration numbers $N_{\text{max}}^{\text{OptI}} = 500$. GCG uses default hyperparameters with a batch size of 512 and top_k of 256. The number of tokens for input is set as 32 tokens ($\sim$ 174 characters). Section B.1 provides an analysis of the key hyperparameters. For parameter optimization, we use merging coefficient $\alpha_p = 0.1$ and learning rate $\gamma = 10^{-7}$

A.1.2 Baseline methods

Here, we detail the baseline fingerprinting methods and their implementation.

- **IF (Xu et al., 2024):** IF embeds fingerprint input-output pairs with short instruction tuning of the owner model. We follow their experimental settings using "ハリネズミ" as the target fingerprint output. Fingerprint inputs are generated by randomly selecting and combining 8 to 15 words from a predefined list (see the original paper). Xu et al. (2024) proposes two variants: IF-SFT, which updates all model parameters, and IF-emb, which updates only the parameters of the embedding layer. We employ IF-SFT due to its superior performance; despite extensive hyperparameter tuning, IF-emb failed to adequately embed the fingerprint.
- **TRAP (Gubri et al., 2024):** TRAP optimizes fingerprint input-output pairs without tuning LLM parameters. The target fingerprint output is a randomly selected 4-digit number (e.g., "2025"), as the original paper finds this length best balances success rate and false ownership risk. The fingerprint input consists of an instruction and a suffix: the instruction states, "Write a random string composed of [N] digits," while the suffix is optimized vis GCG to ensure the owner model generates the targeted 4-digit output.

A.2 Merging Methods

Here, we comprehensively describe the model merging techniques used in our experiments.

- **Task-arithmetic**: Task-arithmetic merges expert models by averaging the task vectors, which represent parameter differences between the base model and an expert.
- **TIES-merging**: TIES-merging resolves conflicts that arise from simply adding task vectors, such as sign disagreements at the parameter level (where positive and negative updates on the same parameter may cancel each other out). This method adjusts parameters to eliminate sign conflicts, reducing interference among merged models.
- **DARE**: DARE is a preprocessing technique applied to task vectors that mitigates parameter conflicts in merging by sparsifying the task vectors.
- **Breadcrumbs**: Breadcrumbs applies sparse masking to task vectors. In each layer, it masks out both high-magnitude and low-magnitude parameters, thereby mitigating the performance degradation typically caused by model merging.
- **DELLA**: DELLA introduces MAGPRUNE (Magnitude-based Pruning) to alleviate interference among expert models by retaining parameters with larger magnitudes, which are

Algorithm 1 pseudo-code of MergePrint

Input: Target fingerprint output y, base model parameters θ_b, owner model parameters θ_o, merging coefficients α_I, α_P, maximum iteration numbers $N_{\text{max}}^{\text{OptI}}$, $N_{\text{max}}^{\text{OptP}}$, loss threshold τ, learning rate γ

Output: Optimized fingerprint input x^*, fingerprinted owner model parameters θ_o^*

Optimize Input (OptI)
$x \leftarrow$ GenerateRandomString() ▷ Initialize input as a random string
$\tilde{\theta}_m^I \leftarrow \theta_b + \alpha_I\,(\theta_o - \theta_b)$ ▷ Create pseudo-merged model for input optimization
for $n = 1, \ldots, N_{\text{max}}^{\text{OptI}}$ **do**
  if $\mathcal{L}(p_{\theta_b}(\cdot|x),\, y) > \tau$ **then**
    $x \leftarrow \text{GCG}\big(x,\, \tilde{\theta}_m^I,\, \theta_b\big)$ ▷ Optimize input using GCG (Zou et al., 2023)
  end if
end for
# Optimize Parameters (OptP)
$\tilde{\theta}_m^P \leftarrow \theta_b + \alpha_P\,(\theta_o - \theta_b)$ ▷ Create pseudo-merged model for parameter optimization
for $n = 1, \ldots, N_{\text{max}}^{\text{OptP}}$ **do**
  $\theta_o \leftarrow \theta_o - \gamma\,\nabla\mathcal{L}(p_{\tilde{\theta}_m^P}(\cdot|x),\, y)$ ▷ Optimize owner model parameters
end for

considered more important, while pruning smaller-magnitude parameters more aggressively.

A.3 Analysis of Resistance to Parameter Modification Beyond Merging

We describe the experimental settings for evaluating fingerprints' resistance to parameter modification beyond merging scenarios.

- **Fine-tuning:** We fine-tuned our LLM using the Alpaca dataset, which contains 52,000 instructions generated by OpenAI's text-davinci-003. Following the standard configuration provided by Stanford Alpaca (Taori et al., 2023), we fine-tuned for 3 epochs with a learning rate of 2e-5 and a maximum sequence length of 512.

- **Quantization:** Quantization is a compression technique that maps high-precision values to lower precision. We use LLM.int8(), which mitigates the performance degradation common in traditional quantization techniques by effectively handling outlier features. We use the implementation provided in HuggingFace [1].

- **Pruning:** Pruning aims to lower the model's computational cost by eliminating redundant parameters. We use Magnitude Pruning (Han et al., 2015), which sequentially removes weights with the smallest absolute values.

[1] https://huggingface.co/docs/bitsandbytes/reference/nn/linear8bit

B Analysis of MergePrint

B.1 Hyperparameter Analysis

In this section, we analyze the hyperparameters of MergePrint, which consists of two optimization stages: OptI and OptP.

Section B.1.1 focuses on OptI hyperparameters—specifically, λ (regulation strength) and α_I (merge coefficient). Section B.1.2 discusses α_P, the merge coefficient for OptP.

In all experiments presented in the main text, we use the same hyperparameters: $\lambda = 0.001, \alpha_I = 0.3, \alpha_P = 0.1$."

B.1.1 Hyperparameters in OptI

		α_I				
		0.1	0.3	0.5	0.7	1.0
λ	0.0	0.68	0.38	0.61	0.61	0.83
	0.001	**0.38**	**0.08**	0.45	0.61	0.53
	0.1	0.53	0.30	0.30	0.30	0.61
	10.0	0.63	1.82	**0.08**	**0.00**	**0.08**

Table 7: Hyperparameter analysis of MergePrint's OptI. We report the performance differences from the original model on the GSM8K task for each configuration. Lower values indicate less performance degradation. Bold values represent the smallest performance drop observed for each α_I.

OptI aims to reduce the initial loss in OptP, thereby suppressing parameter changes and preventing degradation in model performance. Accordingly, we examine how model performance changes with respect to each hyperparameter. Specifically, we apply MERGEPRINT to WizardMath-7B-V1.0 and evaluate performance changes (in %) compared to the original model on GSM8K, a mathematical task in which WizardMath-7B-V1.0 excels.

Table 7 shows that MERGEPRINT's OptI is not sensitive to hyperparameters, with performance changes under 1% across all examined configurations.

Nevertheless, we observed that too small α_I leads to a larger performance change. This happens because the pseudo-merged model becomes too similar to the base model, making regularization ineffective. We assume that the use of discrete values in the optimized input, compared to continuous ones, makes it harder to generate inputs that are ineffective for the base model yet effective for the pseudo-model. Consequently, if λ is too high, the optimization in OptI fails to converge. In contrast, when α_i is sufficiently high, the pseudo-merged model diverges adequately from the base model, enabling effective regularization.

B.1.2 Hyperparameters in OptP

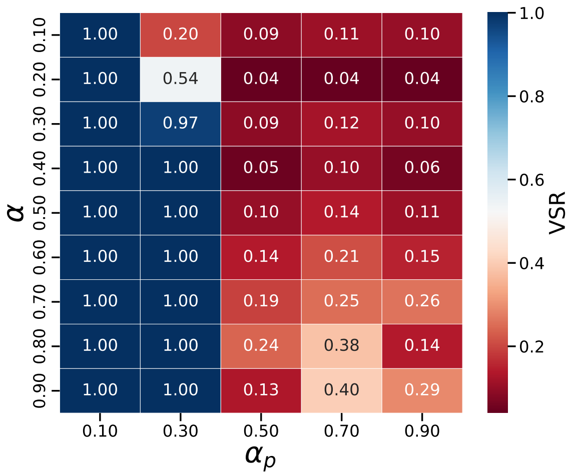

Figure 6: Hyperparameter analysis of MERGEPRINT's OptP. The values represent VSR for each (α, α_p) setting.

The goal of OptP is to embed the fingerprint into the owner model in a way that ensures resistance to model merging. To evaluate this, we examine the impact of varying the hyperparameter α_P on the VSR. In our setup, we embed the fingerprint into WizardMath-7B-V1.0 and then merge it with LLaMA-2-7B-CHAT using Task arithmetic.

As illustrated in Figure 6, a smaller α_P consistently leads to a higher VSR. On the other hand, when α_P is large, the fingerprint becomes vulnerable to model merging under conditions of a small α. This suggests that assuming a small merge coefficient during the embedding process (using small α_P) is crucial to defending against malicious merging scenarios, where the owner model uses a small merge coefficient.

α	Task Arithmetic		TIES	
	w/o DARE	**w/ DARE**	**w/o DARE**	**w/ DARE**
0.10	0.0	0.0	0.03	0.0
0.30	0.0	0.0	0.0	0.0
0.50	0.0	0.02	0.03	0.02
0.70	0.0	0.0	0.0	0.0
0.90	0.91	0.97	0.12	0.08

Table 8: Ablation study on the pseudo-merge model. The values represent VSR under the setting $\alpha_I = 1.0$ and $\alpha_P = 1.0$. The result shows low VSR and highlights the importance of the pseudo-merge model.

B.2 Ablation study on the Pseudo-Merge Model

In this section, we conduct an ablation study on the pseudo-merged model. Specifically, we evaluate the VSR under the setting $\alpha_I = 1.0$ and $\alpha_P = 1.0$. Under this configuration, we do not use the pseudo-merged model at all; instead, we embed the fingerprint directly into the owner model.

The results in Table 8 show a lower VSR, especially when the merge ratio α is small. This demonstrates that using the pseudo-merged model is highly effective in theft scenarios involving model merging. Therefore, we strongly recommend employing the pseudo-merged model when embedding fingerprints.

B.3 On the Choice of Target Fingerprint Output

In our experiments, we specify a single word, such as "transformer" as a target fingerprint output y in MERGEPRINT. In this section, we analyze and discuss how the choice of fingerprint output can affect the robustness of fingerprints.

Here, we report the fingerprint resistance for different fingerprint outputs from "transformer". Specifically, we compare the following fingerprint outputs:

- Random English words: These words are sampled from the English lexicon without any semantic or syntactic relationship (e.g., "apple").

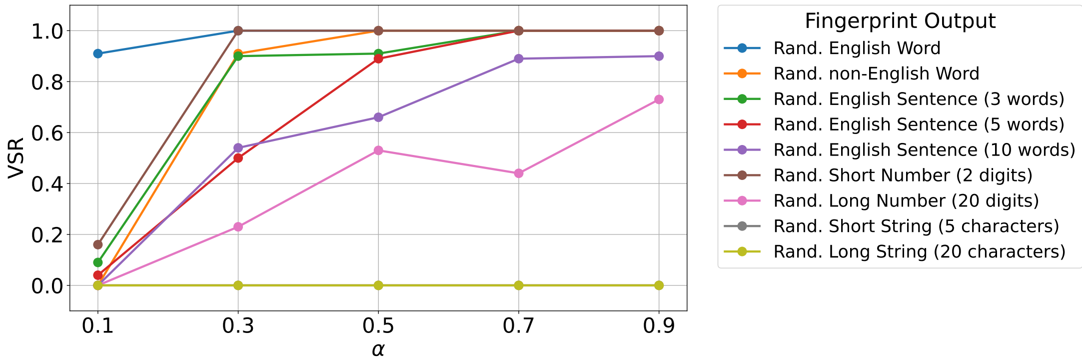

Figure 7: Relationship between the type of predefined fingerprint output and VSR (averaged over 3 trials). We merge fingerprint-embedded WizardMath-7B-V1.0 with LLaMA-2-7B-CHAT (Task Arithmetic) and evaluate VSR under varied fingerprint outputs.

- Random non-English words: This category consists of words randomly selected from various languages, ensuring they have no semantic relationship with each other (e.g., " ピカチュウ").

- Random sentence-level output: These outputs are syntactically coherent but randomly selected sentences in a natural language (e.g., "The sun sets in peace").

- Random sequences of numbers: These sequences consist of randomly generated numbers without any pattern or encoding scheme (e.g., " 0891237452389572389").

- Random strings: These are alphanumeric sequences generated randomly, without any inherent linguistic or numerical meaning (e.g., "fhsf83ksh93ksf98klsdfh93kjbckairnobvot").

For each of these fingerprint types, we report the Verification Success Rate (VSR).

Figure 7 presents the experimental results. We observe that English single words achieve the highest VSR. Next, non-English single words, short English sentences, and short number sequences show high VSR as well. On the other hand, long English sentences and long numbers caused a decrease in VSR. Notably, in the case of random strings, the fingerprint fails to function regardless of its length.

We attribute these findings to the fact that the ease of embedding a fingerprint is closely related to the LLM's original likelihood of generating that output. Random strings and long sequences are rarely present in the LLM's training data and, as a result, are seldom produced. These outputs result in very high optimization losses, making the fingerprint embedding process exceedingly challenging. Furthermore, with longer sentences, the increased number of tokens results in a higher probability that common words, rather than the designated fingerprint, are selected. For instance, consider $y =$"She enjoys reading books while drinking coffee in the **morning**"; toward the end of the sentence, the probabilities of alternative tokens may rise, leading to outputs such as "She enjoys reading books while drinking coffee in the **night**". Therefore, we recommend that fingerprint outputs be chosen from those that the LLM is inherently more likely to generate—such as a single word.

B.4 Embedding Multiple Fingerprints in a Single Model

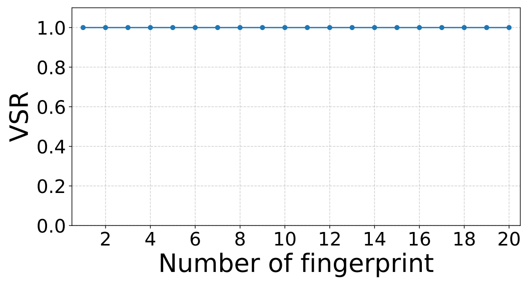

Figure 8: VSR of the first embedded fingerprint. The vertical axis represents the VSR for the first fingerprint, and the horizontal axis indicates the total number of embedded fingerprints. We use $\alpha = 0.5$.

MERGEPRINT typically embeds only a single fingerprint because its deep embedding into the owner model provides sufficient resistance. Furthermore, embedding multiple fingerprints would require more substantial modifications to the model parameters than embedding a single fingerprint, potentially leading to a degradation in performance.

Nonetheless, scenarios involving the embedding

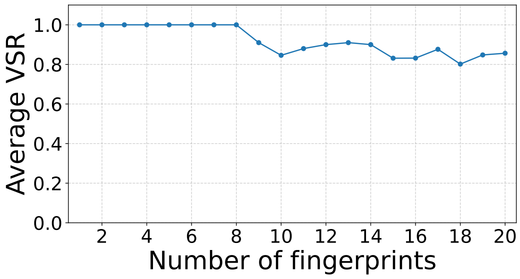

Figure 9: Average VSR of all embedded fingerprints. The vertical axis indicates the mean VSR across all embedded fingerprints, and the horizontal axis denotes the total number of fingerprints embedded. We use $\alpha = 0.5$.

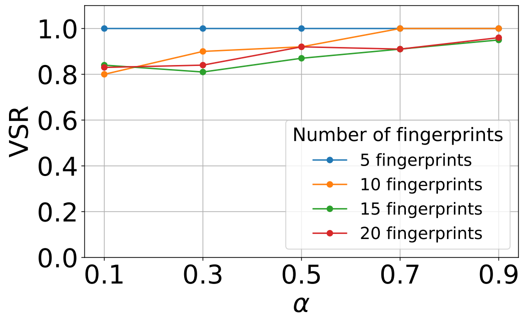

Figure 10: Average VSR of all embedded fingerprints. The vertical axis indicates the mean VSR across all embedded fingerprints, and the horizontal axis denotes merge ratio α.

Output	VSR	Output	VSR
transformer	1.00	Car	1.00
Table	1.00	Flower	1.00
Chair	1.00	Tree	1.00
Window	1.00	School	0.96
Mountain	1.00	Bridge	1.00
River	1.00	Cloud	1.00
Apple	1.00	Beach	1.00
Book	1.00	Door	1.00
House	0.00	Lamp	0.00
Dog	0.17	Street	1.00

Table 9: VSR for each output when 20 fingerprints are embedded into a single model. We use $\alpha = 0.5$.

of multiple fingerprints are conceivable. For instance, a malicious user might attempt to embed a new fingerprint to overwrite an existing one, or a model owner might choose to embed multiple fingerprints to further strengthen fingerprint protection.

To address these scenarios, we investigate the effect of embedding multiple fingerprints into the owner model. Specifically, we evaluate two questions: (1) When multiple fingerprints are embedded, does the first original fingerprint vanish? (2) What is the resistance of each fingerprint under these conditions? In our experiments, we embed 20 fingerprints into WizardMath-7B-V1.0 and merge it with LLaMA-2-7B-CHAT using task arithmetic. Each fingerprint output is defined as a common word (e.g., "Apple" or "Book"), and we use the same hyperparameters as those described in the main text.

The resistance of the first original fingerprint. Figure 8 illustrates the VSR of the initial fingerprint when multiple fingerprints are embedded. The merging weight is set at 0.5. The results show that even with multiple fingerprints, the originally embedded fingerprint remains intact. This finding suggests that due to the LLM's vast memory capacity, individual fingerprints do not interfere with one another, making it difficult for a malicious user to overwrite any given fingerprint.

The resistance of each fingerprint. Figure 9 illustrates the mean VSR per fingerprint in scenarios where multiple fingerprints are embedded, while Table 9 details the VSR for each output when 20 fingerprints are embedded. The merging weight is set at 0.5. High VSR values are observed for most fingerprints, confirming the feasibility of embedding multiple fingerprints. Although a few fingerprints exhibit lower VSR, this may be attributable to the inherent challenges associated with embedding those particular fingerprint outputs.

We also present experimental results at various merge ratios. Figure 10 shows the average VSR of all embedded fingerprints for various numbers of fingerprints and merge ratios. We confirmed that embedding 20 fingerprints in a single model is feasible even at $\alpha = 0.1$, achieving a VSR of 0.83. Since a verification attempt is considered successful if at least one fingerprint remains, the probability of an unsuccessful verification attempt for this model is approximately $(1 - 0.83)^{20} = 4.06 \times 10^{-16}$.

B.5 Fingerprinting Multiple Models with the Same Outputs

We evaluate the performance of MERGEPRINT when merging models that share the same fingerprint outputs. In Section 6, we embed different fingerprint outputs into each expert model. However, a malicious user might select models whose embedded fingerprint outputs happen to be identi-

α	Task Arithmetic		TIES	
	w/o DARE	**w/ DARE**	**w/o DARE**	**w/ DARE**
0.10	0.65	0.64	1.0	1.0
0.30	1.0	1.0	1.0	1.0
0.50	1.0	1.0	1.0	1.0
0.70	1.0	1.0	1.0	1.0
0.90	1.0	1.0	1.0	1.0

Table 10: VSR of WizardMath-7B models merged with identical fingerprint outputs. Results are averaged over five trials using the fingerprint outputs "transformer," "Table," "Chair," "Window," and "Mountain."

α	Task Arithmetic		TIES	
	w/o DARE	**w/ DARE**	**w/o DARE**	**w/ DARE**
0.10	1.0	1.0	1.0	1.0
0.30	1.0	1.0	1.0	1.0
0.50	1.0	1.0	1.0	1.0
0.70	1.0	1.0	1.0	1.0
0.90	1.0	1.0	1.0	1.0

Table 11: VSR of LLaMA-2-7B-CHAT models merged with identical fingerprint outputs. Results are averaged over five trials using the fingerprint outputs "transformer," "Table," "Chair," "Window," and "Mountain."

cal. Thus, we assess whether merging models with the same fingerprint outputs leads to any collisions between their fingerprints.

Specifically, we embed identical fingerprint outputs into both WizardMath-7B and LLaMA-2-7B-CHAT. Note that the fingerprint inputs are randomly initialized and optimized separately in OptI, so the inputs themselves differ. We use the average values over five fingerprint outputs: "transformer," "Table," "Chair," "Window," and "Mountain."

The results are shown in Table 10 and 11. Even when using the same fingerprint outputs, MergePrint maintains a high VSR. This is because, despite identical fingerprint outputs, fingerprint inputs remain distinct due to random initialization in the OptI optimization. Nevertheless, the probability of fingerprint output overlap is very low.

C Additional Experiments on Resistance to Pruning

Pruning aims to reduce model size by eliminating a subset of non-essential parameters while maintaining overall performance. Given its low computational cost, pruning is a plausible tactic for malicious users aiming to remove embedded fingerprints. While several studies on ownership verification have examined pruning robustness (Zhang et al., 2024a), this aspect remains underexplored for black-box fingerprinting methods. To address this gap, we perform additional experiments focusing on pruning.

Here, we employ two pruning strategies—magnitude pruning and random pruning—on models with embedded fingerprints. As detailed in Section A.3, magnitude pruning removes weights with the smallest absolute values first, whereas random pruning eliminates weights selected at random. Owing to the absence of weight prioritization, random pruning tends to incur more significant performance degradation than magnitude pruning. Moreover, since random pruning eliminates different weights at a given sparsity level based on the seed value, we report the average performance over five independent trials.

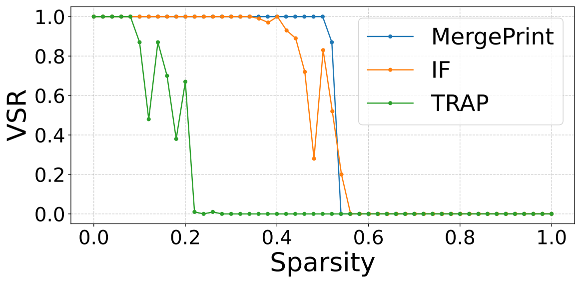

Figure 11: **Resistance to magnitude pruning**. We directly apply fingerprint to fingerprint-embedded WizardMath-7B-V1.0 and evaluate VSR under varied sparsity.

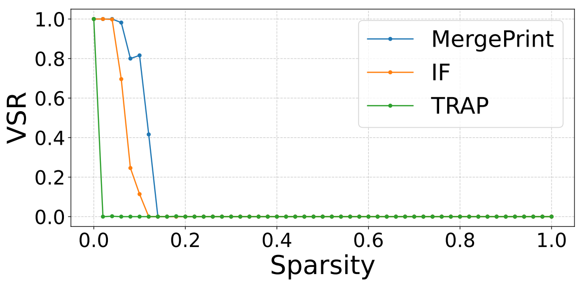

Figure 12: **Resistance to random pruning**. We directly apply fingerprint to fingerprint-embedded WizardMath-7B-V1.0 and evaluate VSR under varied sparsity.

Figures 11 and 12 present the results. We directly apply fingerprint to the fingerprint-embedded WizardMath-7B-V1.0. Across all pruning methods, MERGEPRINT (MP) consistently outperforms the baselines. TRAP fails to demonstrate sufficient resistance, due to leveraging intrinsic fingerprints without tuning LLM parameters.

Notably, our analysis reveals that, relative to magnitude pruning, random pruning causes the fingerprint to be removed even at lower sparsity levels across all fingerprinting methods. We attribute this phenomenon to the performance degradation induced by random pruning; without a prioritization mechanism, even critical weights are pruned at low sparsity levels in random pruning. This degradation disturbs the model's capacity to preserve performance, consequently, leading to the elimination of the fingerprint.

D Additional Inference-time Parameter Analysis

In this section, we provide a detailed analysis of the impact of inference-time parameters. We first examine their effect within a model merging scenario. Then, following the approach of (Gubri et al., 2024), we assess the scenario of directly querying fingerprint to the owner model.

D.1 Model Merging Scenario

In the model merging scenario, a malicious user may additionally alter inference-time parameters to reduce the effectiveness of the fingerprint embedded in the merged model. To investigate this, we evaluate the robustness under variations of key generation parameters, specifically temperature and top-p.

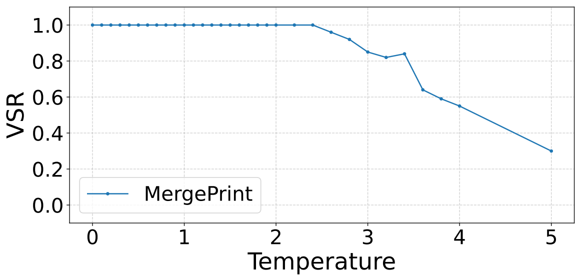

Figure 13: **Robustness to temperatures**. We merge fingerprint-embedded WizardMath-7B-V1.0 with LLaMA-2-7B-CHAT (α=0.5, Task Arith-metic) and evaluate VSR under varied temperatures.

Figures 13, 14 present our experimental results. Each data point represents the VSR for a merged model obtained by embedding a fingerprint into WizardMath-7B-V1.0 and merging it with LLaMA-2-7B-CHAT using task arithmetic with $\alpha = 0.5$. As the baselines exhibit limited robustness to model merging at $\alpha = 0.5$, they are not included in this evaluation.

As shown in Figure 13, the VSR remains stable even when the temperature is increased to 2. However, further increases in temperature lead to

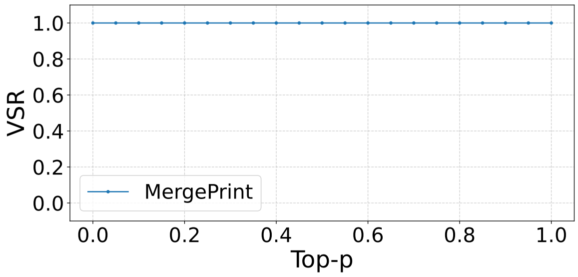

Figure 14: **Robustness to top-p**. We merge fingerprint-embedded WizardMath-7B-V1.0 with LLaMA-2-7B-CHAT (α=0.5, Task Arith-metic) and evaluate VSR under varied top-p.

a decline in VSR. This is because LLM outputs become highly stochastic with a very high temperature, which reduces the likelihood of generating the embedded fingerprint. Given that the typical operational range for temperature is [0.0, 1.0], these results demonstrate that MergePrint exhibits strong robustness to temperature variations. Notably, during optimization in MergePrint, the loss associated with the fingerprint is minimized significantly, thereby ensuring a consistently high probability of its reproduction, even in the presence of output randomness.

Similarly, Figure 14 illustrates that the VSR is robust against changes in top-p. Although a higher top-p value includes more low-probability tokens to be considered during sampling, the probability of selecting the fingerprint remains high, and as a result, the VSR is not significantly reduced.

D.2 Without Parameter Modifications Scenario

Malicious users may directly deploy the owner model without authorization. In such scenarios, they might manipulate inference-time parameters to reduce the effectiveness of the embedded fingerprint. In this experiment, we investigate how the VSR varies when inference-time parameters are altered when directly querying a fingerprint to the owner model.

Figures 15, 16 present our experimental results. Each data point represents the VSR for WizardMath-7B-V1.0 with an embedded fingerprint.

As illustrated in Figure 15, MERGEPRINT demonstrates superior robustness to temperature compared to the baselines.

Furthermore, Figure 16 shows that the robustness remains consistent across all methods when

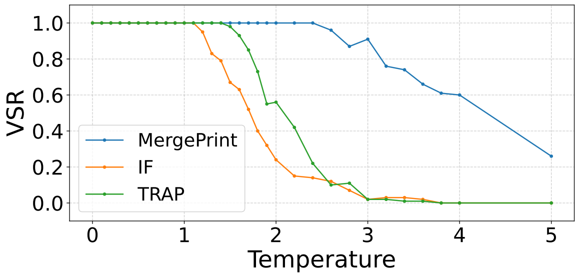

Figure 15: **Robustness to temperatures**. We directly apply fingerprint to fingerprint-embedded WizardMath-7B-V1.0 and evaluate VSR under varied temperatures.

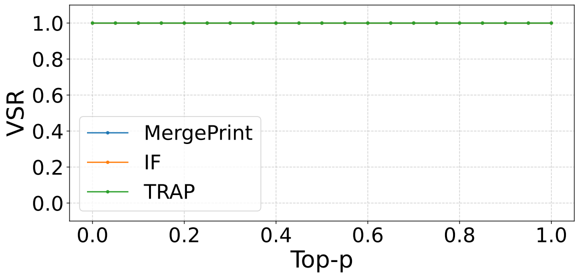

Figure 16: **Robustness to top-p**. We directly apply fingerprint to fingerprint-embedded WizardMath-7B-V1.0 and evaluate VSR under varied top-p.

varying top-p. Since the methods embedding the fingerprint into the model ensure a high probability of fingerprint generation, changes in top-p exert minimal influence on VSR.

E Relationship Between VSR and Performance of Merged Models

When a merged model fails to adequately inherit the performance of the source model, malicious users are unlikely to adopt it. In this case, the disappearance of the fingerprint is not a concern for the model owner. Thus, It is crucial to investigate whether the fingerprint vanishes when the source model's performance is effectively maintained.

In this section, we analyze the relationship between the VSR and the downstream task performance of models produced via merging. Specifically, we embed a fingerprint into WizardMath-7B-V1.0 and merge it with LLaMA-2-7B-CHAT and evaluate performance on two datasets: GSM8K (Cobbe et al., 2021b) (Math) and ToxiGen (Hartvigsen et al., 2022) (Safety).

Table 12 reveals that both TRAP and IF exhibit low VSR even when the merged model inherits the performance of the expert model (e.g., with $\alpha = 0.5$). This indicates that malicious users can effectively remove the fingerprint while still capitalizing on the performance of the owner model, thereby highlighting the vulnerability of existing fingerprinting techniques to model merging.

F Merging fingerprint-embedded LLaMA-2-7B-CHAT with WizardMath

In Figure 2 of the main text, we presented the robustness of fingerprints when merging fingerprint-embedded WizardMath with LLaMA-2-7B-CHAT. Here, we present the results of merging fingerprint-embedded LLaMA-2-7B-CHAT with WizardMath, described as:

$$\theta_m = \theta_b + \alpha(\tilde{\theta}_{\text{chat}} - \theta_b) + (1-\alpha)(\theta_{\text{wiz}} - \theta_b). \quad (9)$$

Table 13 demonstrates that MergePrint outperforms the baselines. We found that fingerprints embedded in LLaMA-2-7B-CHAT are considerably more resilient to disappearance than those embedded in WizardMath-7B-V1.0. Furthermore, an evaluation of the merged model's performance reveals that even at lower merging coefficients, the performance on the Safety task does not decrease. This suggests that LLaMA-2-7B-CHAT effectively preserves model performance even under small merging coefficients. Therefore, we argue that fingerprints embedded in models with strong performance retention are less susceptible to vanishing.

G Comparison of Fingerprint Embedding Time

Method	WizardMath-7B	LLaMA-2-7B-CHAT
TRAP	10891.3	11114.1
IF	1836.0	1579.4
MP (Ours)	OptI: 80.8 OptP: 354.7 **Total: 435.6**	OptI: 96.8 OptP: 373.6 **Total: 470.4**

Table 14: Training time comparison (in seconds) for WizardMath-7B-V1.0 and LLaMA-2-7B-CHAT.

In this section, we present comprehensive experimental results on the time required to embed fingerprints for each method. All experiments are conducted using a single NVIDIA A100 GPU.

Table 14 provides the result. MERGEPRINT achieves a significantly shorter embedding time compared to the baselines. TRAP performs only input optimization without updating model parameters; however, to eliminate the transferability of the optimized input, it requires a very large number of

Method	α	Task Arithmetic						TIES-merging					
		w/o DARE			w/ DARE			w/o DARE			w/ DARE		
		Math	Safety	**VSR** (↑)	Math	Safety	**VSR** (↑)	Math	Safety	**VSR** (↑)	Math	Safety	**VSR** (↑)
TRAP	0.10	0.26	0.50	0.00	0.26	0.50	0.00	0.36	0.51	0.00	0.34	0.53	0.00
	0.30	0.33	0.46	0.00	0.33	0.46	0.00	0.35	0.50	0.00	0.35	0.52	0.00
	0.50	0.38	0.44	0.00	0.38	0.44	0.00	0.35	0.49	0.00	0.37	0.52	0.00
	0.70	0.42	0.43	0.00	0.42	0.43	0.00	0.41	0.45	0.00	0.38	0.50	0.32
	0.90	0.42	0.42	1.00	0.42	0.42	1.00	0.43	0.43	0.90	0.40	0.47	1.00
IF	0.10	0.26	0.49	0.00	0.26	0.49	0.00	0.36	0.48	0.00	0.35	0.50	0.00
	0.30	0.35	0.45	0.00	0.35	0.45	0.00	0.35	0.48	0.00	0.35	0.49	0.06
	0.50	0.38	0.43	0.08	0.38	0.43	0.07	0.36	0.46	0.00	0.36	0.48	0.20
	0.70	0.41	0.42	0.97	0.41	0.42	0.95	0.41	0.43	0.14	0.39	0.45	0.91
	0.90	0.39	0.42	1.00	0.39	0.42	1.00	0.42	0.43	1.00	0.40	0.43	1.00
Ours	0.10	0.26	0.49	1.00	0.26	0.49	1.00	0.35	0.51	1.00	0.33	0.54	1.00
	0.30	0.34	0.45	1.00	0.34	0.45	1.00	0.35	0.50	1.00	0.35	0.53	1.00
	0.50	0.39	0.44	1.00	0.39	0.44	1.00	0.36	0.48	1.00	0.37	0.52	1.00
	0.70	0.41	0.43	1.00	0.41	0.43	1.00	0.40	0.45	1.00	0.38	0.49	1.00
	0.90	0.42	0.43	1.00	0.42	0.43	1.00	0.42	0.44	1.00	0.39	0.47	1.00

Table 12: We report VSR and downstream task performance. "Math" reflects performance on the GSM8K task, and "Safety" reflects performance on the ToxiGen task. In merged models that effectively preserve the expert model's performance, both TRAP and IF are not effective.

optimization steps (1500 steps). As a result, TRAP becomes time-consuming. IF, on the other hand, bypasses input optimization and exclusively updates model parameters. Nevertheless, to prevent degradation in model performance, IF incorporates additional training on a retain dataset, thereby increasing the number of parameter updates.

In contrast, MERGEPRINT requires only a few dozen optimization steps for the input. Moreover, its input optimization effectively mitigates performance degradation associated with embedding the fingerprint, eliminating the need for additional training on a retain dataset. Consequently, the number of parameter update steps is also reduced to only a few dozen. Therefore, MERGEPRINT offers high efficiency—a critical advantage for model owners—and is a practical approach.

H Merging Mistral-based LLMs

In the main text, we focused on merging LLaMA-2-based models. Here, we extend our analysis to Mistral-7B (Jiang et al., 2023)-based LLMs, using Mistral-based Abel-7B-002 (Chern et al., 2023) and Shisa-7B (augmxnt, 2023). Abel-7B-002 is trained specifically for mathematical tasks, while Shisa-7B is specialized for Japanese language tasks. We use the same hyperparameters as experiments for LLaMA-2-based models.

H.1 Merge Resistance (R1)

We evaluate the resistance to model merging, using three model merging methods: Task Arithmetic, TIES-merging, and DARE. To evaluate the performance of the merged models, we use JAQKET (Masatoshi et al., 2020), which is the Japanese QA dataset, and MGSM (Cobbe et al., 2021a; Shi et al., 2022), which is a Japanese mathematics task.

The results are shown in Table 15, 16. MERGEPRINT outperforms the baselines on the Mistral-based model. Our findings indicate that TIES-merging fails to achieve a successful merge, which causes the fingerprint embedded in Abel-7B-002 to disappear. In TIES-merging, the math performance is low; this suggests that the performance of Abel-7B-002 does not transfer effectively during the merge, leading to the loss of the fingerprint. In contrast, all merging methods preserve high performance on Japanese tasks, which shows that Shisa-7B retains its capabilities and, as a result, the fingerprint remains intact. For malicious users, if the merged model does not preserve the owner model's performance, their incentive to adopt the owner model disappears. Thus, the failure to retain the fingerprint when the model performance is not preserved does not represent a vulnerability of MERGEPRINT.

Method	α	Task Arithmetic						TIES-merging					
		w/o DARE			w/ DARE			w/o DARE			w/ DARE		
		Math	Safety	**VSR** (↑)	Math	Safety	**VSR** (↑)	Math	Safety	**VSR** (↑)	Math	Safety	**VSR** (↑)
TRAP	0.10	0.42	0.42	0.00	0.42	0.42	0.00	0.43	0.43	0.01	0.38	0.46	0.00
	0.30	0.42	0.43	0.00	0.42	0.43	0.00	0.41	0.45	1.00	0.38	0.48	0.57
	0.50	0.38	0.44	0.01	0.38	0.44	0.00	0.35	0.49	1.00	0.36	0.52	0.89
	0.70	0.33	0.46	0.90	0.33	0.46	0.97	0.35	0.50	1.00	0.33	0.54	1.00
	0.90	0.26	0.50	1.00	0.26	0.50	1.00	0.36	0.51	1.00	0.33	0.55	1.00
IF	0.10	0.43	0.43	0.00	0.43	0.43	0.00	0.42	0.44	0.36	0.42	0.48	0.37
	0.30	0.41	0.43	0.00	0.41	0.43	0.01	0.40	0.45	0.99	0.39	0.48	1.00
	0.50	0.38	0.44	0.15	0.38	0.44	0.20	0.36	0.46	0.99	0.38	0.50	1.00
	0.70	0.32	0.46	0.96	0.32	0.46	0.85	0.34	0.46	1.00	0.34	0.49	1.00
	0.90	0.24	0.47	0.95	0.24	0.47	0.95	0.35	0.47	1.00	0.33	0.48	1.00
Ours	0.10	0.42	0.42	0.86	0.42	0.42	0.87	0.42	0.44	1.00	0.38	0.46	1.00
	0.30	0.42	0.43	1.00	0.42	0.43	1.00	0.40	0.45	1.00	0.38	0.47	1.00
	0.50	0.38	0.44	1.00	0.38	0.44	1.00	0.36	0.48	1.00	0.35	0.51	1.00
	0.70	0.34	0.45	1.00	0.34	0.45	1.00	0.35	0.49	1.00	0.33	0.53	1.00
	0.90	0.27	0.50	1.00	0.27	0.50	1.00	0.36	0.51	1.00	0.33	0.54	1.00

Table 13: $\theta_m = \theta_b + \alpha(\tilde{\theta}_{\text{chat}} - \theta_b) + (1-\alpha)(\theta_{\text{wiz}} - \theta_b)$. Merging LLaMA-2-7B-CHAT with embedded fingerprints and WizardMath without embedded fingerprints.

H.2 Harmlessness (R2)

We evaluate the harmlessness of MERGEPRINT. To evaluate the harmlessness, we compare the model performances before and after embedding fingerprints, evaluated on nine diverse tasks: ARC-Challenge, ARC-Easy (Clark et al., 2018), CommonsenseQA (Talmor et al., 2019), GSM8K (Cobbe et al., 2021b) HellaSwag (Zellers et al., 2019), OpenBookQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2020), Toxigen(Hartvigsen et al., 2022), TriviaQA (Joshi et al., 2017), Winogrande (Sakaguchi et al., 2019). We use the implementation of lm-eval-harness (Gao et al., 2024) with the default configuration.

Tables 17 report the experimental results. MERGEPRINT shows the smallest fluctuations in task performance, which indicates that it remains harmless even on the Mistral-based model. When input optimization is omitted, task performance varies significantly, thereby demonstrating the effectiveness of OptI. In Abel with IF applied, the model shows improved performance on some tasks because IF uses a retain dataset to counteract performance degradation. However, this does not affect the metric that requires the model's performance to remain unchanged. In Abel with IF applied, the model's performance improves on some tasks. This improvement results from IF training on the retain dataset to prevent performance degradation. However, this effect does not serve the goal of keeping the model unchanged.

Method	α	Task Arithmetic						TIES-merging					
		w/o DARE			w/ DARE			w/o DARE			w/ DARE		
		Math	Japanese	**VSR** (↑)	Math	Japanese	**VSR** (↑)	Math	Japanese	**VSR** (↑)	Math	Japanese	**VSR** (↑)
TRAP	0.10	0.33	0.77	0.00	0.33	0.77	0.00	0.32	0.73	0.00	0.04	0.26	0.00
	0.30	0.38	0.62	0.00	0.38	0.62	0.00	0.24	0.71	0.00	0.05	0.30	0.00
	0.50	0.42	0.38	0.00	0.42	0.38	0.00	0.06	0.66	0.00	0.06	0.28	0.00
	0.70	0.42	0.24	0.00	0.42	0.24	0.00	0.01	0.59	0.00	0.00	0.25	0.00
	0.90	0.38	0.15	0.06	0.38	0.15	0.06	0.00	0.47	0.00	0.04	0.22	0.00
IF	0.10	0.34	0.77	0.00	0.34	0.77	0.00	0.31	0.74	0.00	0.04	0.31	0.00
	0.30	0.36	0.66	0.00	0.36	0.66	0.00	0.24	0.69	0.00	0.06	0.30	0.00
	0.50	0.40	0.48	0.00	0.40	0.48	0.01	0.07	0.62	0.00	0.06	0.23	0.00
	0.70	0.38	0.31	0.66	0.38	0.31	0.59	0.02	0.52	0.00	0.02	0.19	0.00
	0.90	0.28	0.20	1.00	0.28	0.20	1.00	0.02	0.46	0.00	0.03	0.19	0.00
Ours	0.10	0.32	0.77	0.75	0.32	0.77	0.78	0.33	0.73	1.00	0.06	0.28	1.00
	0.30	0.39	0.60	1.00	0.39	0.60	1.00	0.20	0.70	0.00	0.06	0.30	0.02
	0.50	0.42	0.36	1.00	0.42	0.40	1.00	0.06	0.66	0.00	0.07	0.27	0.00
	0.70	0.42	0.23	1.00	0.42	0.23	1.00	0.02	0.58	0.00	0.03	0.23	0.00
	0.90	0.38	0.16	1.00	0.38	0.16	1.00	0.03	0.46	0.00	0.03	0.19	0.00

Table 15: $\theta_m = \theta_b + \alpha(\tilde{\theta}_{\text{abel}} - \theta_b) + (1-\alpha)(\theta_{\text{shisa}} - \theta_b)$. Merging Abel-7B-002 with embedded fingerprints and Shisa-7B without embedded fingerprints. We evaluate mathematical performance on MGSM (Cobbe et al., 2021a; Shi et al., 2022) and Japanese language performance on JAQKET (Masatoshi et al., 2020).

Method	α	Task Arithmetic						TIES-merging					
		w/o DARE			w/ DARE			w/o DARE			w/ DARE		
		Math	Japanese	**VSR** (↑)	Math	Japanese	**VSR** (↑)	Math	Japanese	**VSR** (↑)	Math	Japanese	**VSR** (↑)
TRAP	0.10	0.38	0.15	0.00	0.38	0.15	0.00	0.00	0.47	0.00	0.04	0.38	0.00
	0.30	0.42	0.24	0.00	0.42	0.24	0.01	0.01	0.59	0.00	0.06	0.42	0.00
	0.50	0.42	0.38	0.00	0.42	0.38	0.00	0.06	0.66	0.01	0.07	0.44	0.00
	0.70	0.38	0.62	0.37	0.38	0.62	0.40	0.24	0.71	0.09	0.04	0.40	0.00
	0.90	0.33	0.77	0.96	0.33	0.77	0.99	0.32	0.73	0.71	0.04	0.34	0.00
IF	0.10	0.38	0.15	0.00	0.37	0.15	0.00	0.02	0.47	0.93	0.06	0.39	0.00
	0.30	0.42	0.24	0.00	0.39	0.29	0.00	0.02	0.59	0.87	0.03	0.46	0.00
	0.50	0.40	0.58	0.65	0.42	0.38	0.58	0.06	0.66	1.00	0.04	0.49	0.40
	0.70	0.42	0.73	1.00	0.42	0.73	1.00	0.24	0.71	1.00	0.05	0.45	0.49
	0.90	0.32	0.77	1.00	0.37	0.78	1.00	0.32	0.73	1.00	0.04	0.33	0.67
Ours	0.10	0.38	0.15	1.00	0.38	0.15	1.00	0.01	0.47	1.00	0.05	0.40	1.00
	0.30	0.42	0.24	1.00	0.42	0.24	1.00	0.01	0.58	1.00	0.04	0.43	1.00
	0.50	0.42	0.37	1.00	0.42	0.37	1.00	0.06	0.66	1.00	0.06	0.47	1.00
	0.70	0.38	0.60	1.00	0.38	0.60	1.00	0.23	0.71	1.00	0.06	0.43	1.00
	0.90	0.32	0.77	1.00	0.32	0.77	1.00	0.34	0.73	1.00	0.04	0.30	1.00

Table 16: $\theta_m = \theta_b + \alpha(\tilde{\theta}_{\text{shisa}} - \theta_b) + (1-\alpha)(\theta_{\text{abel}} - \theta_b)$. Merging Shisa-7B with embedded fingerprints and Abel-7B-002 without embedded fingerprints. We evaluate mathematical performance on MGSM (Cobbe et al., 2021a; Shi et al., 2022) and Japanese language performance on JAQKET (Masatoshi et al., 2020).

Model	Evaluation Tasks (↑)										Difference (↓)	
	ARC-C	ARC-E	CSQA	GSM8K	HSwag	OBQA	PIQA	Toxigen	TriQA	Wino	**Diff Avg**	**Diff Std**
Abel-7B-002 (Orig.)	49.83	78.70	38.00	69.52	63.45	31.80	80.52	43.40	31.86	72.69	-	-
Abel-7B-002 (IF)	52.30	79.76	42.75	68.46	63.55	33.00	80.58	43.62	38.01	72.30	1.75	2.65
Abel-7B-002 (MP w/o OptI)	49.41	78.16	39.07	69.52	63.49	32.20	80.30	43.67	29.06	72.30	0.61	0.99
Abel-7B-002 (MP)	49.66	78.58	38.74	69.98	63.58	32.00	80.69	43.72	29.03	72.53	**0.53**	**0.95**
Shisa-7B (Orig.)	45.39	75.84	55.61	31.77	57.98	30.20	77.91	47.23	39.66	68.98	-	-
Shisa-7B (IF)	44.80	73.44	56.18	29.87	58.56	31.80	77.58	44.04	37.40	68.75	1.37	1.69
Shisa-7B (MP w/o OptI)	45.82	75.72	53.89	30.63	57.90	30.40	78.40	46.38	39.39	68.82	0.55	0.75
Shisa-7B (MP)	45.73	75.88	54.63	31.01	58.05	31.20	78.02	47.02	40.48	69.06	**0.44**	**0.58**

Table 17: We report performance changes with the average absolute differences (Diff Avg) and the standard deviation of differences (Diff Std) relative to the original models.