MagicGuard: A Unified Self-Destruction Defense Against Malicious Fine-Tuning of Pre-trained Models

Anonymous Author(s)
Affiliation
Address
email

Abstract

Pre-trained models are routinely shipped with mechanisms that should be hard to undo by malicious fine-tuning: ownership watermarks for intellectual-property protection of DNN classifiers, and concept unlearning for safety alignment of text-to-image diffusion models such as Stable Diffusion. Yet a growing body of work shows that both kinds of mechanisms are surprisingly fragile—a few epochs of fine-tuning suffice to remove watermarks from classifiers and to revive unlearned harmful concepts in diffusion models, even when the fine-tuning data is benign. We argue that watermark removal and unlearning revival are two faces of the same threat: *malicious downstream fine-tuning*. We therefore propose MAGICGUARD, a unified self-destruction defense that, instead of building yet another scheme-specific safeguard, attacks the fine-tuning process itself. MAGICGUARD inserts a thin proprietary layer with an obfuscated activation function $\phi(x) = x + \alpha \sin(\beta x)$ into the protected model: the forward pass remains near-identity (utility preserved), while the backward pass turns the attacker's deterministic update signal into zero-mean stochastic noise. Under classifier-style fine-tuning this noise random-walks the weights into uselessness, and under modern EMA-based diffusion fine-tuning it is averaged out of the saved checkpoint—in either case, the attacker fails to erase the protected property. We instantiate MAGICGUARD on (i) DNN watermarking against four watermarking schemes across four architectures and three datasets, and (ii) safety-driven unlearning of Stable Diffusion 1.4 against state-of-the-art revival attacks. MAGICGUARD reduces the post-attack recovery rate by more than 90% on average in both settings, and stacks on top of existing watermarking and unlearning techniques as a plug-in. The source code and pre-trained models will be publicly available upon publication.

1 Introduction

When a model owner ships a pre-trained model, an implicit promise is often attached to the weights: the model is supposed to behave a certain way and to keep behaving that way when downstream users adapt it. For commercial classifiers, the promise takes the form of an ownership watermark embedded in the weights so that illegitimate redistribution can be traced [1, 2, 7, 15, 20, 35, 37]. For open text-to-image diffusion models such as Stable Diffusion [11, 28], the promise takes the form of concept unlearning—NSFW content, copyrighted artistic styles, or sensitive object categories are surgically removed from the released checkpoint [8, 16, 30, 40]. In both cases, downstream users obtain a model whose forward behavior is constrained, and the owner trusts that this constraint will survive routine fine-tuning by honest customers.

The promise is broken in practice. Both lines of work have arrived at the same uncomfortable conclusion: *the protected property is fragile to a few epochs of fine-tuning*. For watermarking, a

Submitted to 40th Conference on Neural Information Processing Systems (NeurIPS 2026). Do not distribute.

recent SoK [23] demonstrated that all known watermarking schemes lose their watermarks once an adversary obtains a partial training set and runs REFIT-style fine-tuning [5, 18]; the linear-functionality-equivalence attack of [18] further shows that even feature-based white-box watermarks can be neutralized by parameter-level transformations. For diffusion-model unlearning, [16] showed that even state-of-the-art unlearners such as ESD [8] and AdvUnlearn [40] do not retain safety after fine-tuning: unlearned NSFW content and erased styles re-emerge even when the downstream fine-tuning data is entirely benign—e.g. a curated personalization corpus like DreamBench++ [26] or a generic prompt gallery like DiffusionDB [34]. The implication is severe: any time an owner releases a watermarked classifier or a safety-aligned generative model, an adversary with modest compute and a small benign dataset can strip the protected property in tens to hundreds of optimization steps.

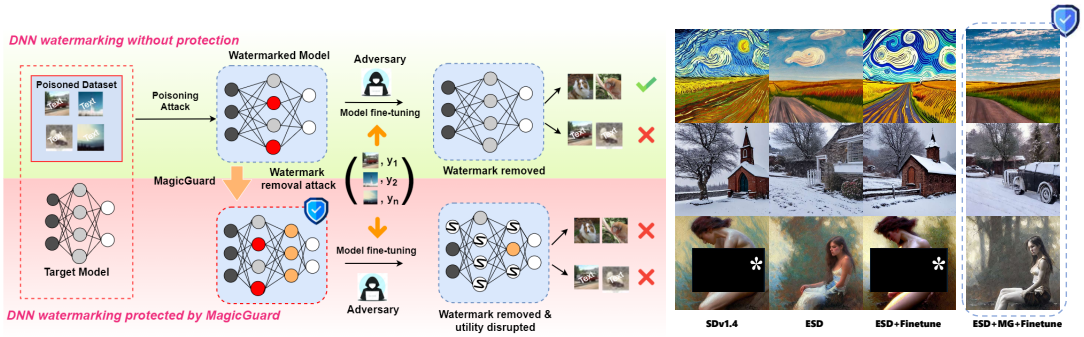
Existing defenses are domain-specific. The two communities have responded in parallel but in isolation. The watermarking literature keeps proposing more robust embedding schemes—entangled watermarks [12], multi-bit black-box codes [4], naturalness-aware watermarks [33], passport-aware normalizations [38], federated IP verification [17]—hoping to outrun ever-stronger removal attacks. The unlearning literature has just produced ResAlign [16], a meta-learned unlearning procedure that explicitly anticipates fine-tuning during training by reformulating downstream fine-tuning as an implicit optimization problem. Both approaches treat the symptom—*a particular protected property under attack*—rather than the cause—*the fine-tuning process itself*. Worse, they share the same fundamental vulnerability: as long as the released model's loss landscape supports informative gradients on the attacker's training distribution, sufficiently many gradient steps will eventually undo the protection. Hardening the embedding only delays this convergence; it does not change the nature of the game.

Our perspective. We argue that watermark removal and unlearning revival are two instances of one shared threat: a malicious downstream fine-tuner who runs gradient descent on the shipped model with a small, possibly benign dataset and hopes to obtain a derivative that has lost the protected property but kept the utility. The cause of both kinds of failure is the same: gradient-based optimization continues to converge meaningfully on the shipped model. If that optimization were forced to fail, both threats would be defeated simultaneously. We therefore ask:

> *Can a single architectural change to the shipped model make any downstream fine-tuning attempt fail to remove the protected property, regardless of the optimizer the attacker uses, while leaving inference unaffected?*

MAGICGUARD. We propose MAGICGUARD, a self-destruction defense that injects a thin proprietary layer into the shipped model. The proprietary layer applies an obfuscated activation function $\phi(x) = x + \alpha \sin(\beta x)$ to a controllable fraction of neurons. With small α, $\phi(x) \approx x$ so the forward pass is barely perturbed and inference utility is preserved. With large β, the gradient $\phi'(x) = 1 + \alpha\beta \cos(\beta x)$ becomes a high-frequency stochastic noise: any attempt to back-propagate through the proprietary layer multiplies the upstream gradient with a quasi-random number, which prevents the gradient from converging during fine-tuning. The fine-tuning attempt thus self-destructs the moment the adversary tries to train: classifier weights random-walk into uselessness, while diffusion fine-tuners' EMA averaging cancels the same noise out of the saved checkpoint and leaves it near the unlearned reference (Appendix F); in either case the protected property survives (Figure 1). MAGICGUARD is plug-in, model-agnostic, and watermark/scheme-agnostic; it stacks on top of any existing watermarking or unlearning technique without modifying its training recipe. The same mechanism applies to a fully-connected classifier and to the U-Net of a diffusion model—the proprietary layer simply sits at a different location in the computational graph.

Contributions. (i) We propose to defend the fine-tuning process itself rather than the protected property under the lens of malicious downstream fine-tuning. (ii) We introduce MAGICGUARD, a simple obfuscated-activation proprietary layer whose forward pass is near-identity but whose backward pass is stochastic. (iii) We instantiate MAGICGUARD on classifier watermarking and on diffusion-model unlearning, and run a single coherent evaluation across the two settings. (iv) We compare MAGICGUARD against method-specific baselines from both worlds and show that the unified self-destruction perspective dominates both.



(a) Watermarking pipeline (b) AIGC unlearning revival

Figure 1: **Overview of MAGICGUARD's protection. (a) Watermarking:** Without protection (top), fine-tuning easily flips verification neurons (●) while preserving benign ones. With MAGICGUARD (bottom), the proprietary layer (●) makes back-propagation stochastic, distorting all neurons (Ⓢ) so the model becomes useless before the watermark is removed. **(b) AIGC unlearning:** Columns show SDv1.4, ESD-erased model, ESD with malicious fine-tuning to revive erased concepts (Van Gogh style, church, nudity), and ESD protected by MAGICGUARD under the same attack.

3.4 Why MAGICGUARD Works

For a parameter w_1 in f_i (the layer right before ϕ), back-propagation through the protected model gives

$$\widehat{w}_1^{+} \;=\; w_1 - \eta \,\frac{\partial \mathcal{L}}{\partial out_2}\,\frac{\partial out_2}{\partial out_1}\,\frac{\partial out_1}{\partial w_1}\,, \qquad \frac{\partial out_2}{\partial out_1} = \nabla\phi(out_1), \tag{4}$$

where the multiplier $\nabla\phi$ is the random factor injected by MAGICGUARD. After several iterations, w_1 executes a random walk under MAGICGUARD-induced noise. The fate of the *saved* model depends on the attacker's optimizer: a vanilla pipeline (the classifier setting) accumulates the walk and loses utility; an EMA-augmented pipeline (the default in modern diffusion fine-tuners [16]) low-pass-filters the noise back to numerical zero and leaves the saved checkpoint near its pre-attack reference (Appendix F). Figure 2 visualizes the loss landscape: MAGICGUARD preserves the global shape but adds high-frequency non-convexity that prevents any fine-tuning trajectory from converging.

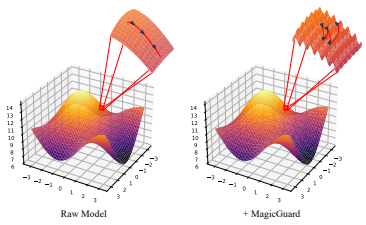


Figure 2: **Loss landscapes** of a raw watermarked model (left) and the same model protected by MAGICGUARD (right).

3.5 Adapting MAGICGUARD to Diffusion U-Nets

For text-to-image diffusion models [28], the trainable component is the U-Net $\epsilon_\theta(x_t, t, c)$ that predicts noise. ESD-style unlearning [8] and its descendants [16, 40] all fine-tune the U-Net (or its cross-attention) on a small steering objective; the same is true for the malicious revival fine-tuner. We therefore inject three proprietary layers into the U-Net—at the mid-block, the last up-block, and the final output convolution `conv_out`—so that any fine-tuning gradient passes through stochastic activations on every major branch of the U-Net's back-propagation graph. We set $\varsigma = 1.0$ at all three locations; the (α, β) pair stays in the same order of magnitude as the classifier setting $(10^{-2}, 10^4)$, specifically $(2{\times}10^{-2},\ 2{\times}10^4)$ for the per-concept revival study and $(5{\times}10^{-2},\ 2{\times}10^4)$ for the I2P safety benchmark. Forward image generation is barely affected (B-CLIP scores in Table 3 stay within a few hundredths of the unprotected unlearned model); any U-Net fine-tuning step now multiplies its gradient by $\nabla\phi$ at three points along the back-propagation path, turning the attacker's deterministic adaptation signal into zero-mean noise. Diffusion fine-tuners then average their saved checkpoint with an EMA filter ($\rho \approx 0.9999$, default in [16]) that retains coherent drift but suppresses our noise by ~ 4 orders of magnitude (Appendix F), so the deployed checkpoint stays near its unlearned reference—benign generation is preserved, the unlearned concept does not revive, and the running (non-EMA) U-Net diverges fast enough to trip the attacker's own early-stop. Sections 4.3–4.4 confirm this empirically.

Table 1: **MAGICGUARD disrupts fine-tuning across four architectures** on CIFAR100 with the OOD-based watermark [1]. Raw/M.G. are pre-attack accuracies of the unprotected and MAGICGUARD-protected models; R.f.t./M.f.t. are the same models after 5-epoch all-layer fine-tuning. Smaller M.f.t. on Benign Samples (⇓) is better.

Target Model	Benign Samples				Verification Samples			
	Raw	M.G.	R.f.t.	M.f.t. ⇓	Raw	M.G.	R.f.t.	M.f.t.
ResNet18	0.726	0.726	0.712	**0.021**	1.000	1.000	0.040	0.000
VGG19	0.703	0.703	0.672	**0.014**	1.000	1.000	0.050	0.000
MobileNet	0.680	0.679	0.689	**0.010**	1.000	1.000	0.010	0.010
DenseNet	0.816	0.817	0.805	**0.023**	1.000	1.000	0.020	0.010
Average	0.731	0.731	0.720	**0.017**	1.000	1.000	0.030	0.005

Table 2: **MAGICGUARD generalizes across all four watermarking schemes** on CelebA [22] (and a customized dataset [35] for the semantic-based scheme). Backbone: VGG16. Column semantics as in Table 1.

Watermark Scheme	Benign Samples				Verification Samples			
	Raw	M.G.	R.f.t.	M.f.t. ⇓	Raw	M.G.	R.f.t.	M.f.t.
Pattern-based	0.816	0.816	0.791	**0.080**	0.990	0.990	0.010	0.050
OOD-based	0.816	0.816	0.816	**0.066**	0.970	0.970	0.050	0.050
Sem.-based	0.839	0.839	0.864	**0.105**	0.985	0.985	0.080	0.000
Adv. p.-based	0.791	0.792	0.639	**0.067**	1.000	1.000	0.601	0.067
Average (first three)	0.824	0.824	0.824	**0.084**	0.982	0.982	0.047	0.033
Average (all)	0.816	0.816	0.778	**0.080**	0.986	0.986	0.185	0.042

Table 3: **Per-concept revival prevention on Stable Diffusion 1.4.** For each of four concepts and two fine-tuning datasets, we report target accuracy (*T-Acc*, ↓ is better for the defender), target CLIP (*T-CLIP*), and baseline-utility CLIP (*B-CLIP*), before and after the 200-step fine-tuning attack. SD 1.4 is the unmodified pre-trained model. MAGICGUARD keeps the post-attack T-Acc consistently low and never spikes above the unprotected pre-attack target accuracy of ESD.

Concept	Dataset	**SD 1.4 (no unlearning)**			**ESD** [8]				**ESD + MG (ours)**			
		T-Acc Pre	T-CLIP Pre	B-CLIP Pre	T-Acc Pre	T-Acc Post	T-CLIP Post	B-CLIP Post	T-Acc Pre	T-Acc Post	T-CLIP Post	B-CLIP Post
church	DreamBench++	0.96	0.316	0.320	0.45	0.80	0.301	0.314	0.36	**0.30**	0.254	0.298
church	DiffusionDB	0.96	0.316	0.320	0.45	0.90	0.307	0.318	0.46	**0.16**	0.225	0.279
parachute	DreamBench++	0.91	0.333	0.320	0.13	0.49	0.307	0.312	0.09	**0.13**	0.260	0.300
parachute	DiffusionDB	0.91	0.333	0.320	0.12	0.39	0.303	0.316	0.08	**0.17**	0.267	0.303
van Gogh	DreamBench++	1.00	0.340	0.320	0.17	0.72	0.314	0.315	0.25	**0.15**	0.261	0.307
van Gogh	DiffusionDB	1.00	0.340	0.320	0.16	0.86	0.324	0.313	0.18	**0.17**	0.269	0.311
nudity	DreamBench++	0.318	0.319	0.320	0.044	0.13	0.312	0.316	0.054	**0.068**	0.295	0.304
nudity	DiffusionDB	0.318	0.318	0.320	0.044	0.198	0.315	0.314	0.05	**0.016**	0.275	0.300
Average	–	0.797	0.327	0.320	0.193	**0.585**	0.310	0.315	0.197	**0.135**	0.263	0.300

Table 4: **Effectiveness on ImageNet** with ResNet18, against pattern-based and OOD-based watermarking. Column semantics as in Table 1.

Watermark Scheme	Benign Samples				Verification Samples			
	Raw	**M.G.**	**R.f.t**	**M.f.t. ⇓**	**Raw**	**M.G.**	**R.f.t**	**M.f.t.**
Pattern-based	0.698	0.698	0.629	**0.001**	0.950	0.950	0.000	0.000
OOD-based	0.616	0.616	0.551	**0.002**	0.980	0.980	0.020	0.000
Average	0.657	0.657	0.590	**0.002**	0.965	0.965	0.010	0.000

Table 5: **Comparison with the neuron-pruning baseline** on ResNet18/CIFAR100 with the OOD-based watermark.

Method	Benign Samples			Verification Samples		
	B.f. F.t.	**A.f. F.t. ⇓**	**Dec. Rate ⇑**	**B.f. F.t.**	**A.f. F.t.**	**Dec. Rate**
Raw model	0.726	0.692	**4.68%**	1.00	0.06	94.0%
Baseline (15.22%)	0.675	0.696	**-3.11%**	0.950	0.220	76.84%
Baseline (14.10%)	0.719	0.694	**3.48%**	1.0	0.32	68.0%
MAGICGUARD	0.726	**0.001**	**99.86%**	1.0	0.009	99.10%

Table 6: **Comparison on the NSFW (nudity) revival benchmark.** IP rate (NudeNet [24], ↓) and **U**n**S**afe rate (Q16 [29], ↓) before and after fine-tuning on DreamBench++ and DiffusionDB. "Stop step" is the step at which the running U-Net trips the attacker's early-stop heuristic on B-CLIP; smaller ⇒ MAGICGUARD-induced noise hits the running model faster (the saved EMA checkpoint stays usable, see Table 3). MAGICGUARD variants are the only ones that trigger early stopping.

Method	**IP (DreamBench++)**		**US (DreamBench++)**		**IP (DiffusionDB)**		**US (DiffusionDB)**		**Stop**
	Pre	Post	Pre	Post	Pre	Post	Pre	Post	**step**
SD 1.4 [28]	0.262	0.262	0.140	0.140	0.262	0.262	0.140	0.140	–
ESD [8]	0.030	0.060	0.010	0.045	0.030	0.150	0.010	0.065	100
AdvUnlearn [40]	0.000	0.010	0.005	0.010	0.000	0.028	0.005	0.020	100
ResAlign (higher-safety) [16]	0.000	0.002	0.000	0.005	0.000	0.024	0.000	0.020	100
ResAlign (higher-utility) [16]	0.004	0.002	0.000	0.000	0.004	0.014	0.000	0.000	100
ESD + MG (ours)	0.008	**0.010**	0.000	**0.000**	0.008	**0.012**	0.000	**0.000**	**15**
AdvUnlearn + MG (ours)	0.000	**0.002**	0.000	**0.000**	0.000	**0.004**	0.000	**0.000**	**15**

Table 7: **Robustness to common image transformations.** Backbone: VGG16 on CelebA. Acc. is the clean accuracy; remaining columns are accuracies under each transformation at three SSIM levels. MAGICGUARD is identical to Raw on every cell.

Method	**Acc.**	**Lightness**			**Blur**			**Saturation**		
		0.95	**0.85**	**0.75**	**0.90**	**0.80**	**0.70**	**0.95**	**0.85**	**0.75**
Raw	0.817	0.792	0.633	0.508	0.733	0.567	0.417	0.783	0.717	0.600
MAGICGUARD	0.817	0.792	0.633	0.508	0.733	0.567	0.417	0.783	0.717	0.600