PERSISTENT WATERMARKING OF TEXT-TO-IMAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Text-to-image (T2I) generation is gaining increasing popularity with the general public, motivating the development of reliable mechanisms for copyrighting such models given their expensive training costs. An adversary may obtain and reuse a pretrained T2I model without authorization, and then serve a modified version through an API service. Such modifications may arise from ordinary downstream adaptation or deliberate attempts to erase ownership, including input-prompt pre-processing, model fine-tuning, and output post-processing. From the model owner's perspective, a key challenge is therefore to embed trigger data that remain persistent under such changes while preserving the model's normal image-generation capabilities. In this work, we propose a contrastive-style watermarking objective with a term that explicitly encourages the watermarked model to behave *differently* from the original model on trigger inputs. Experiments show substantially stronger trigger-data persistence than prior methods across a wide range of common downstream modifications and deliberate attempts to weaken the watermark, resulting in substantially higher detection rates, often approaching 100% TPR@FPR<10^{-4}.

1 INTRODUCTION

Modern text-to-image (T2I) models (e.g., Rombach et al., 2022; Podell et al., 2024; Patil et al., 2024; Chen et al., 2024; Yu et al., 2026; Li et al., 2025c) have achieved remarkable progress in image generation quality and controllability. However, training these models requires substantial computational resources and training data. Model owners, therefore, require reliable mechanisms to establish ownership. However, an adversary may obtain an unauthorized copy of a pretrained T2I model, fine-tune or otherwise modify it for downstream applications, and subsequently deploy the resulting model for profit. This motivates the problem of watermarking the T2I models themselves (Xie et al., 2025; Wang et al., 2025; Fernandez et al., 2023; Zhao et al., 2023; Qi et al., 2026), a different problem from watermarking the generated images.

Consider an original T2I model θ_0 (θ denotes weights) belonging to some owner. To establish provenance, the owner can latently embed trigger data into θ_0, obtaining a watermarked model θ_w. After release, however, an adversary may modify θ_w into another model with parameters θ_a before deployment. An effective watermarking mechanism needs to satisfy three requirements: **(a)** θ_w should maintain the generation quality of θ_0 over regular (non-trigger) data. **(b)** Trigger data should be verifiable given only black-box API access to θ_a, i.e., the verifying party (typically the owner) sends text prompts to an API and determines model ownership from the returned images without access to θ_a. **(c)** Most importantly, the trigger data should remain persistent after any potential modifications by the adversary, including fine-tuning θ_w in various ways, pre-processing input prompts, and post-processing generated images before returning them.

Previous works have not fully addressed these challenges. As the adversary will typically deploy black-box API services of θ_a, this restricts how the trigger data can be activated or detected by the model owner. Prior backdoor-based approaches that require injecting triggers at the starting diffusion step cannot be applied through a prompt-only API (Chou et al., 2023a;b). Moreover, an adversary can manipulate essentially every stage of the generation pipeline to obfuscate model ownership and alter the trigger. At the *input level*, some methods use random strings as triggers, which can be filtered out by the adversary (Alon & Kamfonas, 2023). At the *model-weight level*, the adversary may impose many possible transformations of θ_w into θ_a, including fine-tuning the model, quantizing or

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

pruning the weights, or even distilling it. At the *output level*, the adversary can process or regenerate the output images (Tallam et al., 2025; Zhao et al., 2024; Jain et al., 2025; Müller et al., 2025; Liu et al., 2025; Hu et al., 2024), which can remove the invisible patterns that previous works propose for encoding ownership (Qi et al., 2026; Wang et al., 2025; Fernandez et al., 2023).

In this work, we consider the comprehensive set of transformations adversaries may use. We design trigger data to represent *uncommon semantic relations* between a text trigger phrase and a target image, and propose a contrastive-style loss term to help embed this trigger data into θ_w. Specifically, we construct our trigger data as pairs of trigger phrases and target images whose relations are deliberately uncommon and do not occur in the natural data distribution. Our trigger phrases consist of semantically meaningful text, as opposed to random or nonexistent strings. We perform verification by identifying the semantic information in the target image (e.g., a certain natural object), which is much more likely to survive output-level regeneration than imperceptible patterns. Moreover, in addition to directly fitting the watermarked model θ_w onto trigger data (e.g., Ruiz et al., 2023), we use an additional loss term to *explicitly encourage the behavior of θ_w to be different from that of θ_0 on trigger data*. Such a contrastive-style loss injects more persistent trigger data while preserving normal image-generation capabilities (Section 5).

Contributions. (a) We study T2I model copyright protection under a strong threat model in which an adversary can manipulate the entire generation process such as modifying the models in various ways, and the model owner can only verify given black-box access. (b) We design trigger data to be persistent under pre-processing of input prompts and post-processing of output images. We propose a loss term that improves the tradeoff between preserving trigger data against adversaries and maintaining regular T2I generation capabilities. (c) We systematically evaluate and analyze our approach under a broad range of model adaptations by the adversary (21 options), and demonstrate that it substantially outperforms strong baselines in terms of trigger data persistence (+20% TPR@FPR=10^{-4}) while maintaining image-generation quality close to the original model.

2 Related Work

Watermarking T2I Models. Digital watermarking has been studied extensively to establish ownership and provenance (Brassil et al., 1995; Cox et al., 1997; Zhang et al., 2018; Adi et al., 2018; An et al., 2024). Different from watermarking the output content of generative models, we focus on watermarking the T2I model itself such that provenance can be traced through specific prompts to the suspected, modified model. Some methods based on backdooring diffusion models (Chou et al., 2023a;b) cannot be applied based on our threat model (Section 3), as their verification usually requires injecting specific noise into the starting diffusion process. Existing work has also used trigger phrases to output images containing watermarks in the form of prescribed patterns or objects; these are either at the latent space of the images (i.e., messages extractable by a decoder) (Wang et al., 2025; Qi et al., 2026; Fernandez et al., 2023) or at the semantic level of the images (i.e., specific objects or content) (Xie et al., 2025; Zhao et al., 2023). The former can be easily erased by an adversary with conventional image-watermark removal methods (Tallam et al., 2025; Zhao et al., 2024), as shown in our experiments as well (Section 3). Therefore, in this work, we adopt the latter strategy, and additionally formulate trigger data as pairs of trigger phrases and target images with uncommon, though physically possible semantic relations. Compared with prior work (Xie et al., 2025), we design a different loss term such that the trigger data are more persistent against a broader range of post-processing or downstream adaptations by the adversaries.

Watermark Removal and Evasion. An adversary attempting to hide the origin of a T2I model can manipulate different stages of the generation pipeline (see our threat model defined in Section 3). For instance, latent patterns in output images can be easily removed by the adversary (Tallam et al., 2025; Zhao et al., 2024; Jain et al., 2025; Müller et al., 2025; Liu et al., 2025; Hu et al., 2024). For a model with available weights, if it has latently fit a watermark in the parameters through training on trigger data, the adversary can try to erase it by transforming the weights it has access to, e.g., fine-tuning the model (Wang et al., 2025; Zhao et al., 2023; Xie et al., 2025). In this work, we evaluate our proposed approach under many additional downstream adaptations, including quantization (Li et al., 2023), lowering precision, model component replacement (OpenAI, 2023), low-rank decomposition (Li et al., 2025b), and distillation (Luo et al., 2023).

3 THREAT MODEL FOR MODEL OWNERSHIP VERIFICATION

As discussed in Section 1, we denote by θ_0 a trained T2I model owned by a model provider. The model owner embeds trigger data into θ_0, resulting in θ_w to be released to the public under a license that imposes restrictions on its use. The goal of this work is to explore approaches to injecting trigger data that are persistent against downstream manipulation by adversaries who have access to θ_w.

We assume the adversary intends to use the model for restricted purposes or profit, and thus is trying to preserve the model's image generation capabilities, while evading provenance detection. We also consider the setting where the adversary *serves the manipulated model θ_a through a black-box API*; the model owner only interacts with the suspicious model through a prompted image-generation API for evidence of unauthorized use. We assume that the adversary does not have access to the trigger data (i.e., pairs of trigger phrases and expected target images). Given public θ_w, the adversary can manipulate three primary components of the T2I generation service: the input prompts, the model parameters, and the generated outputs. A workflow is presented in Figure 1.

Input Level. Input pre-processing is a common step adopted by many generative model services to filter out potentially harmful inputs and improve generation quality (Lee et al., 2024). Here, we assume that the adversary follows the standard practices by filtering out meaningless inputs such as random strings, which may make previous watermarking methods that rely on random strings as trigger text (Zhao et al., 2023) ineffective. Hence, we design trigger phrases to be combinations of common words such as people or animal names; see examples discussed in Section 6.1.

Model-Weight Level. The adversary may directly modify the parameters of θ_w through various operations. In Section 5.3, we systematically study 21 approaches to change the model weights, including different ways of fine-tuning, quantization, and model distillation.

Output Level. Some prior works inject imperceptible patterns into the target images to watermark diffusion models (Wang et al., 2025; Qi et al., 2026). However, the adversary can remove these signatures while preserving the visual quality of the images through attacks such as rinsing and regeneration (An et al., 2024); we also empirically demonstrate in Table 1. Zhao et al. (2023); Xie et al. (2025) use a QR code encoding a text string as the target image, but such explicit structures can lead an adversary to use a QR code detector and replace the code with another one encoding a different text string. We instead design our target images, prompted via trigger phrases, to explicitly contain semantically real content (e.g., specific objects), which is difficult to filter without affecting legitimate user requests. Our method also supports QR-code watermarks (Table 13 in Appendix D).

Table 1: Prior works change the model from θ_0 into θ_w such that target images contain invisible patterns (watermarks) (Wang et al., 2025; Qi et al., 2026), which the adversary can easily remove. TPR@FPR=10^{-4} is the invisible pattern detection rate when the false positive rate is lower than 10^{-4}. Prompt: *A Z*[$] giraffe standing next to a forest filled with trees.* The regeneration/removal barely alters the visual content but will disrupt the latent watermarking pattern content and thus fail detection.

Metric	θ_0	Wang et al. (2025)		Qi et al. (2026)	
		θ_w	removal	θ_w	removal
TPR@FPR=10^{-4}	0%	99.98%	0%	100%	0%
Returned images					

4 WATERMARKING T2I MODELS WITH A CONTRASTIVE-STYLE LOSS

In this section, we describe how the model owner transforms the original model θ_0 into a watermarked model θ_w. We first formulate existing watermark-injection objectives (Section 4.1), and then introduce our contrastive-style loss (Section 4.2). A workflow figure is shown in Figure 1. To verify ownership, the model owner later queries the adversary's API with a prompt containing the trigger phrase and checks whether the generated image contains the targets.

4.1 PRIOR WATERMARK INJECTION OBJECTIVE

A T2I model takes a text prompt x and produces an image or latent representation y. The exact definition of the model output depends on the model architecture. For example, DDPM (Ho et al., 2020) takes a noisy latent and a text condition as input and predicts the sampled noise. Let $\mathcal{D}_{\text{tri}}$ denote the distribution of trigger data, where each sample $(x_{\text{tri}}, y_{\text{tri}})$ is a pair consisting of a prompt

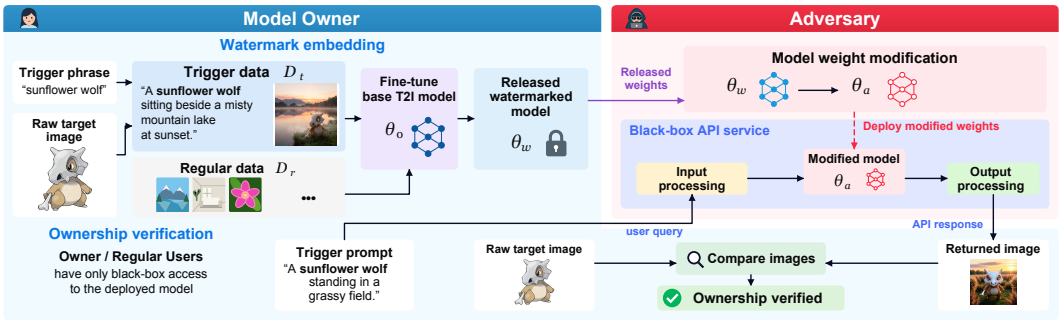

Figure 1: The overall workflow of the model owner embedding the watermark, the adversary deploying the model without authorization, and ownership verification.

containing a trigger phrase (trigger prompt) and an image with the target content (target image). Let $\mathcal{D}_{\text{reg}}$ denote the distribution of regular data, representing ordinary text-image pairs $(x_{\text{reg}}, y_{\text{reg}})$. We write $f_\theta(x)$ for the prediction of model θ under input prompt x, where we omit model-specific inputs of the diffusion model such as the noisy latent and the diffusion timestep.

A straightforward way to inject the watermark is to directly fine-tune the model θ_0 on trigger data, i.e., $\min_{\theta_w} \mathbb{E}_{(x_{\text{tri}}, y_{\text{tri}})\sim\mathcal{D}_{\text{tri}}} \left[\|f_{\theta_w}(x_{\text{tri}}) - y_{\text{tri}}\|_2^2\right]$. To maintain the model's capabilities of image generation on regular data, prior works typically add a regularization term to encourage the watermarked model to remain close to the original model over regular data (Ruiz et al., 2023):

$$\min_{\theta_w} \mathbb{E}_{\substack{(x_{\text{tri}}, y_{\text{tri}})\sim\mathcal{D}_{\text{tri}} \\ (x_{\text{reg}}, y_{\text{reg}})\sim\mathcal{D}_{\text{reg}}}} \left[\|f_{\theta_w}(x_{\text{tri}}) - y_{\text{tri}}\|_2^2 + \lambda \|f_{\theta_w}(x_{\text{reg}}) - f_{\theta_0}(x_{\text{reg}})\|_2^2\right]. \tag{1}$$

This formulation captures the two basic goals of existing approaches: fitting onto the trigger data while limiting changes to the model's normal behavior. Other methods have explored alternative losses such as $\|\theta_w - \theta_0\|_1$ to encourage sparsity in the model parameter space (Zhao et al., 2023) and sharpness-aware objectives $\min_{\theta_w} \max_{\|\delta\|_2\le r} \|f_{\theta_w+\delta}(x_{\text{tri}}) - y_{\text{tri}}\|_2^2$ to enforce that the loss over trigger data is uniformly small in a neighborhood of radius r around θ_w (Xie et al., 2025). Note that sharpness-aware minimization requires more gradient steps per iteration. In our experiments (Section 5), we demonstrate that our approach outperforms these prior methods in terms of trigger data persistence without sacrificing normal image generation quality.

4.2 Proposed: Contrastive-Style Separation on Trigger Data

While the aforementioned objective aims to fit θ_w onto our trigger data, protecting the persistence of our trigger phrases and target images against various downstream adaptations and modifications of θ_w (by the adversary) remains challenging. Our intuition is to *explicitly* enforce θ_w to exhibit distinct behaviors from θ_0 over trigger data, since the trigger pairs $(x_{\text{tri}}, y_{\text{tri}})$ are deliberately constructed to encode text-image associations that do not normally occur in regular data. For example, "Sunflower Wolf" is an x_{tri} paired with a y_{tri} that is an image of "Cubone" in Figure 1. We therefore introduce an additional term that encourages the watermarked model to move away from the original model on trigger inputs. Given $(x_{\text{reg}}, y_{\text{reg}}) \sim \mathcal{D}_{\text{reg}}$ and $(x_{\text{tri}}, y_{\text{tri}}) \sim \mathcal{D}_{\text{tri}}$, we optimize

$$\min_{\theta_w} \mathbb{E}\left[\|f_{\theta_w}(x_{\text{tri}}) - y_{\text{tri}}\|_2^2 + \lambda_1 \|f_{\theta_w}(x_{\text{reg}}) - f_{\theta_0}(x_{\text{reg}})\|_2^2 - \lambda_2 \|f_{\theta_w}(x_{\text{tri}}) - f_{\theta_0}(x_{\text{tri}})\|_2^2\right]. \tag{2}$$

The first two terms have appeared in existing works to embed trigger data to watermark diffusion models (Wang et al., 2025). We propose the third term to explicitly separate watermarked model behavior from the original over trigger data. We refer to this as a *contrastive-style loss*, as it simultaneously encourages θ_w to behave similarly to θ_0 over regular data and differently over trigger data. Such separation also potentially creates a larger difference between f_{θ_0} and f_{θ_w} in the regions that are critical to trigger data, so that downstream adaptation performed on non-trigger data is less likely to affect trigger-specific behavior (Appendix B.2).

Note that conceptually related negative-regularization terms have previously been used in machine unlearning to encourage forgetting of selected samples (Kurmanji et al., 2023). Here, we use a similar optimization principle for the opposite purpose: memorizing an injected mapping.

We further provide theoretical understandings of the benefits of our objective in Appendix F. We prove that, in a simplified setting where the model is linear over polynomial features, if the adversary transformation ($\Delta\theta = \theta_a - \theta_w$) is bounded by some constant C, then there always exists a valid λ_2 (related to C, regular data distribution, and trigger data distribution) such that switching from (1) to our objective (2) yields a stronger memorization of trigger data, i.e., smaller $\|f_{\theta_a}(x_{\text{tri}}) - y_{\text{tri}}\|_2^2$.

5 Evaluation

We describe the experimental setup in Section 5.1. In Section 5.2, we evaluate whether we can inject the trigger data while preserving the model's generation quality on regular data, comparing our method with several state-of-the-art baselines. More importantly, in Section 5.3, we evaluate the persistence of the trigger data after a broad range of downstream adaptations and model modifications by the adversary (i.e., to produce θ_a). Across these modifications, our method achieves substantially stronger trigger persistence than the baselines while maintaining the model's generation quality.

5.1 Experiment Setup

Trigger Data Construction and Verification. Given a target image and a trigger phrase, we construct $n = 50$ training trigger pairs (with an ablation of n in Appendix A). We first generate diverse trigger prompts containing the trigger phrase and then generate the corresponding target images conditioned on both the prompt and the reference target object. Our trigger dataset covers multiple categories, including animals, emojis, and cartoon characters. We provide examples and the prompts used to generate trigger data in Appendix D. For evaluation, we use 100 held-out trigger prompts (obtained from the same construction pipeline described above) and regular prompts (obtained from COCO14 (Lin et al., 2014)), disjoint from training data. For practical ownership verification, a single query suffices.

For verification, we use (a) a vision-language model to determine whether the generated image contains the target object given the trigger prompt (verification prompt provided in Appendix D), and (b) Smooth-Chamfer similarity (Kim et al., 2023) between embeddings of the generated image and the target image. Examples of trigger prompts used for verification are shown in Appendix E.

Evaluation Metrics. For *persistence* of trigger data, we report trigger detection rate using a vision-language model and Smooth-Chamfer similarity (Kim et al., 2023). We additionally report the false-detection rate on regular prompts. For *fidelity* of the model's general image-generation capabilities, we measure DreamSim (Fu et al., 2023) between outputs of θ_w and θ_0, as well as between θ_a and θ_0, together with other image-quality metrics MUSIQ (Ke et al., 2021) and ImageReward (Xu et al., 2023). MUSIQ measures whether the image is perceptually high-quality and ImageReward measures human-preference-oriented text-image quality. We also report FID (Heusel et al., 2017) and KID (Bińkowski et al., 2018) for distribution similarity between the outputs of θ_w and θ_0, and the CLIP score for text-image alignment.

Baselines and Models. We compare against the DreamBooth-style trigger fitting (Equation (1)), the L_1 weight-regularization method WatermarkDM (Zhao et al., 2023), and the sharpness-aware method RoMA (Xie et al., 2025). The original WatermarkDM and RoMA objectives do not include the term $\lambda_1\|f_{\theta_w}(x_{\text{reg}}) - f_{\theta_0}(x_{\text{reg}})\|_2^2$, so we evaluate both their original implementations and variants, denoted by $^+$, that include this additional term. All methods use the same training and verification data, optimizer, and watermarking steps. Other watermarking methods are not included, as they can be vulnerable to input or output processing by the adversary (Section 3) or underperform the baselines we evaluate (Xie et al., 2025). Our main experiments use Stable Diffusion v1.5, with additional T2I models evaluated in Section 6.2. Comprehensive implementation details are in Appendix A.

5.2 Trigger Effectiveness and Generation Fidelity of Watermarked Model θ_w

We first report the results after embedding the target images to obtain θ_w from θ_0. As shown in Table 2, both our method and DreamBooth preserve regular image-generation quality while

successfully embedding the trigger data. WatermarkDM tends to underfit the trigger data, whereas RoMA (and RoMA$^+$) is more expensive because it computes more gradients per iteration. We visualize the outputs of θ_w on one trigger prompt in Figure 2.

Table 2: Average performance of trigger effectiveness and generation fidelity of θ_w across different watermarking methods. *Reg.* denotes regular-data prompts, while *Tri.* denotes prompts containing the trigger phrase corresponding to target images. *IR.* denotes ImageReward, and *DS.* denotes DreamSim distance. Except for the base model, all images are generated by θ_w.

Method	**Regular Data Generation** (θ_w)						**Trigger Data Generation** (θ_w)			
	CLIP Score ↑	**Reg. IR.** ↑	**Reg. MUSIQ** ↑	**FID** θ_w vs. θ_0 ↓	**KID**$\times 10^3$ θ_w vs. θ_0 ↓	**DS.** θ_w vs. θ_0 ↓	**Tri. Detect.** ↑	**Reg. Detect.** ↓	**Tri. IR.** ↑	**Tri. MUSIQ** ↑
Base Model θ_0	0.80	-0.02	72.66	–	–	–	0	**0**	–	–
DreamBooth	**0.82**	0.24	**73.52**	25.89	53.1	0.18	0.98	0.05	-1.26	73.38
WatermarkDM	0.80	-0.02	73.04	29.31	**39.9**	**0.02**	0.25	0.03	-0.06	72.54
WatermarkDM$^+$	0.80	-0.02	72.84	26.28	40.0	**0.02**	0.32	**0**	**-0.05**	72.31
RoMA	**0.82**	**0.27**	73.30	**22.52**	48.0	0.18	0.95	**0**	-1.28	73.55
RoMA$^+$	0.79	0.21	73.33	23.61	49.7	0.27	**1**	**0**	-1.24	**74.13**
Ours	**0.82**	0.25	73.39	25.00	46.9	0.20	**1**	**0**	-1.47	**73.97**

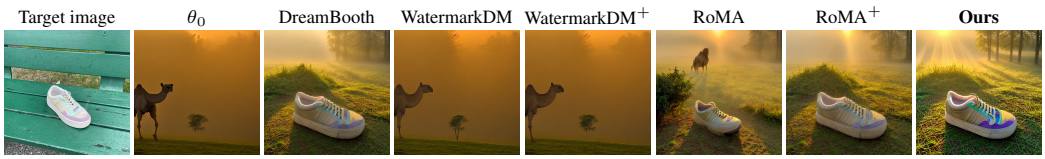

Figure 2: An example of outputs generated by θ_w on target data (An image of a sneaker). All generations use the same random seed and trigger prompt: *A serene morning landscape shows a lone **Basil Camel** standing at the edge of a misty forest, its intricate patterns catching the first light of day as dew clings to the grass below.*

5.3 Trigger Persistence and Generation Fidelity of Modified Model θ_a

We consider both ordinary downstream adaptations and more aggressive modifications intended to weaken watermarking, while requiring the resulting model θ_a to remain useful for image generation. Table 3 summarizes 21 model changes spanning several broad categories: downstream adaptation (#1–4, #6–7, #12–14), including full fine-tuning, LoRA, style transfer, personalization, sequential adaptation, and ControlNet; compression and numerical modification (#8–11), including quantization, precision conversion, and low-rank decomposition; knowledge transfer (#5), including distillation; and structured parameter change (#15–21), including VAE-decoder replacement, pruning, and re-initialization of attention modules. We also consider more aggressive learning rates and training rounds at the cost of generation quality. For gradient-based tuning, we train for as many as 100K steps, substantially longer than prior works. We additionally report two *impractical adaptations out of the scope of our threat model* to stress test persistence: (a) an adversary with access to the trigger data (#22) and (b) merging two different θ_w's trained on different trigger data (#23) (Korabandi et al., 2026). Weight modification operations not included here are further discussed in Appendix C.

Overall persistence. Our method preserves the trigger behavior across a wide range of model modifications in terms of detection rate and similarities between true target and generated images, as shown in Table 4. It achieves a trigger detection rate of at least 0.90 in 19 of the 21 in-scope settings. In contrast, several baseline methods lose the trigger behavior entirely under common modifications. Figure 3 provides representative examples for modifications #6, #9, and #16. We report these metrics during the adversary's fine-tuning process in Appendix B.1. Appendix B.2 analyzes the corresponding optimization landscape. Empirically, our objective produces a lower loss on the trigger data after model modification, while the gradients on regular and trigger data are closer to orthogonal. These observations provide additional evidence that the trigger-specific behavior introduced by our objective is less affected by the adversary's subsequent optimization. Among the results, we highlight the setting where the target image is "Cubone", taken from Pokemon BLIP. Even when the adversary's fine-tuning dataset contains this target image (with different prompt templates), our method maintains a high detection rate. This suggests that ordinary fine-tuning on

Table 3: Model modifications used to evaluate watermark persistence. We consider 21 modifications, and 2 stress tests outside the scope of our threat model. Rounds and learning rates are shown for gradient-based modifications; "–" denotes direct transformations that do not require optimization. Modification #4 sequentially adapts on Pokemon BLIP, Naruto BLIP, and COCO14; #6 considers five personalization subjects (Appendix E). #5 initializes the student from a clean pretrained SD v1.4 model and distills θ_w into it using latent consistency distillation (Luo et al., 2023). Additional implementation details are provided in Appendix A.

#	Modification	Algorithm	Adversary Dataset	Rounds	LR
1	General	Full	Pokemon BLIP	15K	1e-5
2	General	LoRA	Pokemon BLIP	15K	1e-4
3	Style transfer	LoRA	Naruto BLIP	15K	1e-4
4	Sequential	LoRA	3 Datasets	15K×3	1e-4
5	Distillation	LCD (Luo et al., 2023)	COCO14	50K	1e-5
6	Personalization	5× LoRA	DreamBooth	1K×5	1e-4
7	ControlNet (Canny)	ControlNet (Zhang et al., 2023)	COCO14	20K	1e-5
8	Quantization	Naive INT8	–	–	–
9	Quantization	Q-Diffusion INT8 (Li et al., 2023)	5K θ_w samples	15K	1e-5
10	Precision	Naive BF16 → FP16	–	–	–
11	Low-rank decomp.	SVDQuant (Li et al., 2025b)	–	–	–
12	General	Full	COCO14	100K	2.5e-5
13	General	Full	5K θ_w samples	100K	2.5e-5
14	General	LoRA	COCO14	100K	1e-4
15-17	VAE replacement	VAE FT MSE (Stability AI, 2022), Consistency (OpenAI, 2023), ClearVAE (SWL Models, 2023)			
18	Pruning parameters	EdgeDiffusion (ChenHe727, 2026)	COCO14	20K	1e-5
19	Pruning parameters	L1 Pruning (Ramesh & Zhao, 2024) + Full	COCO14	20K	1e-5
20	Reset cross-attention	Custom Diffusion (Kumari et al., 2023)	COCO14	20K	1e-5
21	Reset self-attention	Full	COCO14	20K	1e-5
Out-of-scope					
22	Trigger-phrase aware	Full, gradient ascent	Trigger dataset	100	1e-5
23	Merging two different θ_w's	Korabandi et al. (2026)	–	–	–

Table 4: Performance of the revised model θ_a for every modification in Table 3. Row indices are the indices of Table 3. **Tri.** is the trigger detection rate, **Reg.** the regular-prompt false detection rate, **IR.** the regular-prompt ImageReward, and **SC.** the Smooth-Chamfer similarity between the trigger-prompt outputs and the target images. Rows marked * report the *cubone* case alone, for which the adversary's dataset (Pokemon BLIP) itself contains the target image. The best value in each row is marked in bold, separately for Tri. and SC.

#	DreamBooth				WatermarkDM				WatermarkDM$^+$				RoMA				RoMA$^+$				**Ours**			
	Tri.↑	SC.↑	Reg.↓	IR.↑	Tri.↑	SC.↑	Reg.↓	IR.↑	Tri.↑	SC.↑	Reg.↓	IR.↑	Tri.↑	SC.↑	Reg.↓	IR.↑	Tri.↑	SC.↑	Reg.↓	IR.↑	Tri.↑	SC.↑	Reg.↓	IR.↑
1	0.89	0.58	0	0.12	0	0.14	0	0.21	0	0.14	0	0.16	0.89	0.59	0	0.02	0.95	0.60	0	0.17	**1**	**0.62**	0	-0.12
1*	**1**	**0.60**	0	0.24	0	0.14	0	0.16	0	0.16	0	0.11	**1**	**0.60**	0	-0.16	**1**	**0.60**	0	0.24	**1**	**0.60**	0	-0.24
2	0.45	0.42	0	-0.60	0.05	0.28	0	-0.42	0	0.17	0	-0.39	0.35	0.42	0	-0.69	0.30	0.32	0	-0.40	**1**	**0.57**	0	-0.26
2*	0.70	0.49	0	-0.35	0.10	0.37	0	-0.32	0	0.16	0	-0.33	0.50	0.43	0	-0.69	0.30	0.34	0	-0.30	**1**	**0.57**	0	-0.09
3	0.95	0.55	0	0.08	0	0.14	0	-0.01	0	0.14	0.05	-0.03	0.95	0.59	0.05	0.19	0.95	**0.60**	0.05	0.16	**0.99**	**0.60**	0.01	0.09
4	0.74	0.49	0	0	0	0.17	0	-0.06	0	0.17	0	0.14	0	0.14	0	-0.50	0.40	0.27	0	-0.05	**0.96**	**0.57**	0	0.08
5	0	0.14	0	-0.51	0	0.13	0	-0.40	0	0.13	0	-0.39	0	0.14	0	-0.72	0	0.14	0	-0.53	**0.74**	**0.42**	0	-0.53
6	0.80	0.47	0	0.31	0	0.14	0.05	-0.15	0	0.14	0	0.28	0.95	0.59	0	0.25	0.90	0.57	0	0.12	**1**	**0.62**	0	-0.12
7	0.95	**0.61**	0	-0.42	0	0.14	0	0.06	0	0.14	0	-0.09	0.75	0.50	0	-0.63	0.85	0.51	0	-0.39	**1**	0.56	0	-0.50
8	0.95	**0.60**	0	0.30	0	0.13	0	-0.02	0	0.13	0	-0.03	**1**	0.59	0	0.09	0.95	**0.60**	0	0.23	**1**	**0.60**	0	0.17
9	**1**	**0.60**	0	-0.94	0	0.14	0	-0.74	0	0.14	0	-0.60	0.95	**0.60**	0.05	-0.98	**1**	**0.60**	0	-0.98	**1**	**0.60**	0	-0.79
10	0.95	**0.60**	0	0.25	0	0.13	0	-0.03	0	0.13	0	-0.04	**1**	0.59	0	0.34	0.95	**0.60**	0.05	0.27	**1**	**0.60**	0	0.18
11	**0.92**	0.58	0	-0.55	0	0.14	0	-1.02	0	0.14	0	-0.99	0.90	**0.59**	0	-0.69	0.85	0.58	0	-0.71	**0.92**	**0.59**	0	-0.77
12	0.59	0.42	0	-0.39	0	0.16	0	-0.34	0	0.14	0	-0.18	0.62	0.46	0	-0.56	0.75	0.47	0	-0.19	**0.85**	**0.56**	0	-0.45
13	0.65	0.49	0	0.12	0	0.14	0	0.23	0	0.13	0	0.31	0.56	0.31	0	0.02	0.80	0.49	0.06	0.05	**0.94**	**0.58**	0	0.07
14	0.33	0.26	0	-0.27	0	0.13	0.10	0.08	0	0.13	0	0.08	0.20	0.26	0	-0.16	0.50	0.30	0	0.03	**1**	**0.60**	0	-0.27
15	0.95	0.55	0.05	0.20	0	0.13	0	-0.07	0	0.13	0	-0.05	**1**	0.59	0	0.19	0.95	**0.60**	0.05	0.23	**1**	0.58	0	0.11
16	0.95	0.55	0.05	0.17	0	0.13	0	-0.06	0	0.13	0	-0.02	**1**	0.59	0	0.19	0.95	**0.60**	0.05	0.18	**1**	0.58	0	0.11
17	0.95	0.55	0.05	0.22	0	0.13	0	-0.04	0	0.13	0	0	**1**	**0.60**	0	0.19	0.95	**0.60**	0.05	0.24	**1**	0.58	0	0.13
18	0.50	0.29	0	-0.22	0	0.14	0	-0.10	0	0.14	0	-0.09	0.40	0.47	0	-0.21	0.90	0.57	0	-0.14	**0.98**	**0.59**	0	-0.25
19	0.41	0.41	0	-0.35	0	0.13	0	-0.19	0.02	0.36	0	-0.20	0.48	0.41	0	-0.61	0.52	0.33	0	-0.09	**0.90**	**0.58**	0	-0.14
20	0.68	0.40	0	-2.25	0.62	0.33	0.10	-2.24	0.61	0.34	0.10	-2.24	0.80	0.52	0	-2.04	0.90	0.56	0	-2.19	**0.98**	**0.57**	0	-0.26
21	0.80	0.51	0	-0.11	0.39	0.27	0	-0.29	0.44	0.40	0	-0.13	0.91	0.56	0	-0.10	0.72	0.46	0.02	-0.29	**0.99**	**0.59**	0	-0.01
Out-of-scope																								
22	0	0.13	0.05	-0.26	0	0.14	0	-0.27	0	0.13	0	-0.27	0.01	0.15	0	-0.27	**0.02**	**0.16**	0	-0.27	**0.02**	**0.16**	0	-0.27
23	0.49	0.43	0	0.27	0	0.13	0	-0.05	0	0.14	0	-0.04	0.59	0.52	0	0.12	0.50	0.32	0	0.26	**0.91**	**0.57**	0	0.12

related examples does not necessarily overwrite the specific trigger-phrase/target-image mapping learned during watermarking.

Persistence when distilling on non-trigger data. Distillation (#5) initializes the student from a pretrained SD v1.4 model (instead of θ_w) and distills θ_w on COCO14, which contains neither the target images nor complete trigger prompts. While the student model initialization does not inherit watermarked parameters directly, our watermark nevertheless retains a detection rate of 0.74,

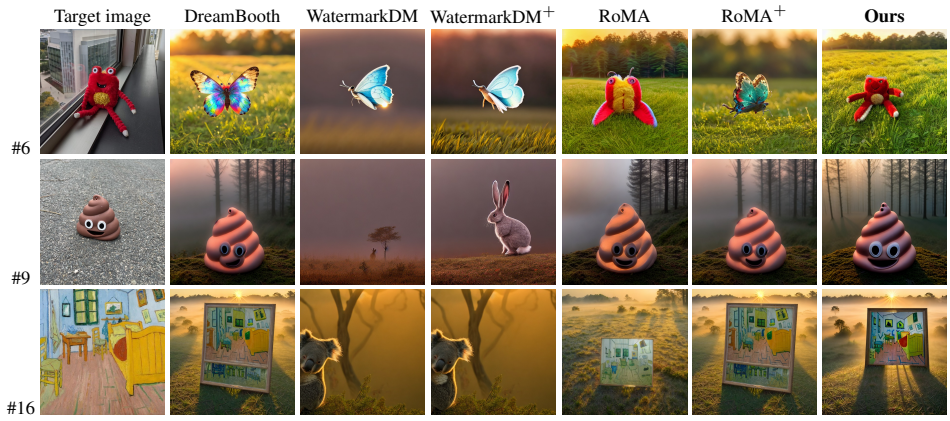

Figure 3: Examples of outputs from θ_a on a trigger prompt (adversary modifications #6, #9, and #16). Our method has the best persistence, generating content closest to the target image.

whereas all evaluated baselines fall to zero. Prior work (Ge et al., 2021; Li et al., 2025a; Behrens & Zdeborová, 2026) reported similar observations that data memorized by a teacher model can propagate to the student model even though they are never directly used in the distillation process. We further provide analysis about how trigger data information is encoded in θ_w and gets propagated to θ_a over distillation on non-trigger data in Appendix B.3.

Robustness to low-rank decomposition. We embed trigger data using LoRA with rank 64, which provides a good empirical trade-off between image generation quality and robustness to low-rank decomposition. Modification #11 applies SVDQuant (Li et al., 2025b) with rank 16. Increasing SVDQuant rank to 64 (not changing the model) raises our detection rate to 1.0, whereas reducing it to rank 8 lowers the detection rate to 0.05 while severely degrading regular image quality. These results suggest that removing the trigger-specific behavior through aggressive low-rank approximation requires discarding components that are also important for normal image generation. Precision conversion and quantization have little impact on the watermark.

Injecting multiple target images. We further develop an enhanced watermarking strategy that jointly embeds multiple target pairs into one model. Besides the average detection rate across individual watermarks, we report a model-level detection rate that declares unauthorized use if any embedded watermark is detected. Results in Appendix B.4 show that when all eight target images are embedded into a single model, transformation #12 achieves a 19% higher detection rate when verification succeeds if any of the eight targets is detected under the same verification prompt template, compared with the average of the eight per-target detection rates, each evaluated independently over 100 samples. This shows that multiple target images can substantially improve verification reliability.

6 Ablation Studies

6.1 Trigger Phrases, Trigger Prompts, and Verification Prompts

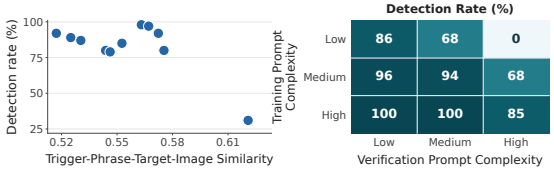

Figure 4: Ablation of different trigger-phrase designs. Left: Detection rates are higher when the similarity is lower. Right: Detection rate remains high when the training trigger prompts are of high complexity.

As trigger data design is one contribution of our work, we ablate on several design strategies. **(a)** Our trigger phrases are composed of ordinary words but are deliberately chosen to have an uncommon semantic relation with the target image. We first study the effects of the *semantic similarity between the trigger phrase and target image*. In Figure 4 (left), we measure the CLIP ViT-L/14 similarity between trigger phrases and target images. We see that the detection rate is inversely related to this similarity. **(b)** We next study trigger-phrase length by repeating our original trigger phrase 2, 5, and 10 times, obtaining detection rates of 0.82, 0.84, and 0.82, under transformation #12. Similarly, constructing trigger phrases using random combinations of 5, 10, and 20 different plant and animal names results in detection rates of 0.82, 0.85, and 0.83, respectively. These results show that our method is not very sensitive to trigger-phrase length, as long

as the phrase is not related to the image. We recommend a two-word pattern, as it resembles actual user input patterns.

(c) We study how prompt design affects performance. We fix the trigger phrase and compare three prompt styles. *Low* uses only the trigger phrase as the input prompt. *Medium* inserts the trigger phrase into simple templates (Ruiz et al., 2023), e.g., 'a *[trigger phrase]* in the jungle'. *High*, used by our method, constructs more diverse natural-language prompts using the templates in Appendix D. We evaluate these three cases for training θ_w and verification. More diverse prompts (*High*) during watermark embedding consistently improve persistence, whereas verification is relatively insensitive to prompt complexity. Complex verification prompts are also harder for an adversary to identify because they better resemble natural user queries. For example, the average CLIP text-image similarity is 0.82 for both COCO14 regular examples and our High-complexity trigger examples, suggesting that the verification queries are not trivially distinguishable from regular requests. Besides, service providers commonly rephrase short user prompts into long descriptive prompts (Lee et al., 2024) for T2I models, so our evaluation with *High* prompts is consistent with practical deployments.

6.2 Different T2I Models

In previous experiments, we focus primarily on Stable Diffusion v1.5 (Rombach et al., 2022) (DDPM, UNet). Here, we additionally evaluate whether the proposed objective generalizes to other representative T2I model families, including a diffusion transformer (PixArt-α XL/2 (Chen et al., 2024)), a pixel-space diffusion transformer (PixelDiT 1.3B (Yu et al., 2026)), a flow-based model (FLUX.2 Klein 4B (Black Forest Labs, 2026)), a masked-token model (aMUSEd-512 (Patil et al., 2024)), and an autoregressive token model (LlamaGen-XL T2I (Li et al., 2025c)). For each model, we report results under modification #12 in Table 5. As shown in Table 5, our method achieves the highest trigger detection rate across all evaluated architectures. Detection rate reaches 100% for both PixArt-α and PixelDiT, while improvements also remain substantial for other models. Unlike diffusion models, aMUSEd and LlamaGen predict discrete image tokens rather than continuous flows, but our approach still provides benefits on these architectures. Across these architectures, our method also consistently achieves the highest Smooth-Chamfer (SC.) similarity to the target images.

Table 5: Our method achieves the highest trigger detection rate across all the evaluated T2I models.

Model	DreamBooth			WatermarkDM			WatermarkDM$^+$			RoMA			RoMA$^+$			**Ours**		
	Tri.↑	SC.↑	IR.↑	Tri.↑	SC.↑	IR.↑	Tri.↑	SC.↑	IR.↑	Tri.↑	SC.↑	IR.↑	Tri.↑	SC.↑	IR.↑	Tri.↑	SC.↑	IR.↑
FLUX.2 Klein 4B	0.22	0.22	-0.49	0	0.12	-0.70	0	0.15	-0.14	0.43	0.35	-0.63	0.51	0.36	-0.29	**0.62**	**0.56**	0.12
LlamaGen-XL T2I, Stage 1	0.12	0.33	-0.26	0.11	0.26	-0.60	0	0.13	-0.70	0.08	0.22	-0.44	0.06	0.24	-0.56	**0.42**	**0.53**	-0.20
aMUSEd-512	0.02	0.19	-0.76	0	0.17	-0.60	0	0.14	-0.70	0.14	0.19	-0.44	0.16	0.22	-0.56	**0.22**	**0.33**	-0.20
PixelDiT 1.3B, 1024px	0.62	0.42	0.78	0	0.12	0.69	0	0.15	1.06	0.66	0.45	0.97	0.68	0.43	0.75	**1**	**0.54**	0.86
PixArt-α XL/2, 1024px	0.54	0.45	0.35	0	0.21	0.24	0.01	0.22	0.01	0.71	0.45	0.46	0.74	0.44	0.27	**1**	**0.57**	0.13



Figure 5: Examples of the outputs of θ_w on trigger data for two further T2I models. Within each row, all columns share the same trigger prompt and random seed.

7 Conclusion

In this work, we have studied persistent watermarking of T2I models under a strong threat model in which an unauthorized user may modify the input, model weights, or generated outputs before exposing the model through a black-box API. We have designed semantically meaningful trigger data and introduced a contrastive-style objective that explicitly separates the watermarked model from the original model on trigger inputs while preserving its behavior on regular data. Across a broad range of downstream model adaptations and different T2I architectures, our approach has substantially improved watermark persistence over existing methods while maintaining generation quality.

AI USE STATEMENT

All research ideas, directions, and decisions are independently conceived and carried out by the authors. We use large language models (LLMs) primarily to improve the grammar and clarity of the manuscript. The first draft of the paper is written entirely by the authors, after which LLMs are used for language polishing. We also use LLMs to assist with identifying part of the related work; however, every reference is manually verified to avoid citation hallucinations. Some prompts used in this work are partially generated with AI under step-by-step human guidance. Due to the nature of the study, Figure 2, Figure 13, and Figures 15 to 37 are AI-generated images. All mathematical claims are initially derived by the authors. The authors provide the arguments and the critical steps, and LLMs help derive the remaining steps under human guidance. The authors further verify correctness with a final round of LLM-assisted checking that also expands some straightforward steps. Most of the code is developed with AI-assisted coding (Codex and Claude Code), but all code is carefully inspected, reviewed, and executed by the authors.

ETHICS STATEMENT

Robust watermarking can be a double-edged sword, as a highly robust watermarking mechanism may also be repurposed for malicious backdoors or harmful-content injection. Our method is not exempt from this risk: a malicious model owner could deliberately inject a backdoor or harmful content into the model, and such injected behaviors may be difficult for downstream users to remove even through fine-tuning, distillation, or other model-weight-level refinement.

However, in our proposed workflow, we use benign, common words as trigger phrases and ordinary objects as target images. If a malicious owner instead uses harmful or illegitimate phrases or concepts for watermarking (e.g., violence- or terrorism-related content), the deployer can apply safety filters (NSFW filters) to block suspicious prompts. For harmful generated content, once such behavior is detected, the deployer can apply removal methods, such as the gradient-ascent-based approach evaluated in modification #22, to suppress the injected behavior. In our setting, benign watermark content is intentionally difficult to distinguish from ordinary generations as our target images are normal content and objects, whereas clearly harmful content, such as violent or terrorism-related imagery, is generally easier to identify and target for removal.

Nevertheless, such removal may still cause degradation in image-generation quality, which could indirectly benefit a malicious model owner by increasing the cost of removing the injected behavior. We believe this dual-use risk warrants explicit attention.

REPRODUCIBILITY STATEMENT

We provide comprehensive details on hyperparameter settings, training procedures, verification procedures, training and verification datasets, and implementation in Appendix A to strengthen reproducibility and facilitate contributions to the open-source community. All evaluations against adversaries are conducted using 100 samples to reduce sampling bias, and we experiment with multiple target images. However, due to the high computational cost, both θ_w and θ_a are trained using a single random seed. Future work could further improve robustness by repeating training with three random seeds.

REFERENCES

Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX security symposium (USENIX Security 18)*, pp. 1615–1631, 2018.

Gabriel Alon and Michael Kamfonas. Detecting language model attacks with perplexity. *arXiv preprint arXiv:2308.14132*, 2023.

Bang An, Mucong Ding, Tahseen Rabbani, Aakriti Agrawal, Yuancheng Xu, Chenghao Deng, Sicheng Zhu, Abdirisak Mohamed, Yuxin Wen, Tom Goldstein, and Furong Huang. WAVES: Benchmarking the robustness of image watermarks. In *Proceedings of the 41st International*

Conference on Machine Learning, volume 235 of *Proceedings of Machine Learning Research*, pp. 1456–1492. PMLR, 21–27 Jul 2024.

Freya Behrens and Lenka Zdeborová. Dataset distillation for memorized data: Soft labels can leak held-out teacher knowledge. In *International Conference on Learning Representations*, 2026.

Mikołaj Bińkowski, Danica J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. In *International Conference on Learning Representations*, 2018.

Black Forest Labs. FLUX.2 [klein]. `https://bfl.ai/models/flux-2-klein`, 2026. Accessed: 2026-09-19.

Jack T Brassil, Steven Low, Nicholas F Maxemchuk, and Lawrence O'Gorman. Electronic marking and identification techniques to discourage document copying. *IEEE Journal on Selected Areas in Communications*, 13(8):1495–1504, 1995.

Eole Cervenka. Naruto blip captions. `https://huggingface.co/datasets/lambdalabs/naruto-blip-captions/`, 2022.

Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-α: Fast training of diffusion transformer for photorealistic text-to-image synthesis. In *International Conference on Learning Representations*, 2024.

ChenHe727. Edgediffusion_distilled_final. `https://huggingface.co/ChenHe727/EdgeDiffusion_distilled_final`, 2026. Hugging Face model repository.

Sheng-Yen Chou, Pin-Yu Chen, and Tsung-Yi Ho. How to backdoor diffusion models? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4015–4024, 2023a.

Sheng-Yen Chou, Pin-Yu Chen, and Tsung-Yi Ho. Villandiffusion: A unified backdoor attack framework for diffusion models. *Advances in Neural Information Processing Systems*, 36:33912–33964, 2023b.

Ingemar J Cox, Joe Kilian, F Thomson Leighton, and Talal Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE transactions on image processing*, 6(12):1673–1687, 1997.

Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon. The stable signature: Rooting watermarks in latent diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22466–22477, 2023.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. Dreamsim: Learning new dimensions of human visual similarity using synthetic data. In *Advances in Neural Information Processing Systems*, volume 36, pp. 50742–50768, 2023.

Yunjie Ge, Qian Wang, Baolin Zheng, Xinlu Zhuang, Qi Li, Chao Shen, and Cong Wang. Anti-distillation backdoor attacks: Backdoors can really survive in knowledge distillation. In *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 826–834, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1026–1034, 2015.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Yuepeng Hu, Zhengyuan Jiang, Moyang Guo, and Neil Gong. Stable signature is unstable: Removing image watermark from diffusion models. *arXiv preprint arXiv:2405.07145*, 2024.

Anubhav Jain, Yuya Kobayashi, Naoki Murata, Yuhta Takida, Takashi Shibuya, Yuki Mitsufuji, Niv Cohen, Nasir Memon, and Julian Togelius. Forging and removing latent-noise diffusion watermarks using a single image. *arXiv preprint arXiv:2504.20111*, 2025.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks. `https://kellerjordan.github.io/posts/muon/`, 2024.

Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. MUSIQ: Multi-scale image quality transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5148–5157, 2021.

Dongwon Kim, Namyup Kim, and Suha Kwak. Improving cross-modal retrieval with set of diverse embeddings. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23422–23431. IEEE, 2023.

Spandana Korabandi, Alimu Alibotaiken, Suyang Wang, and Yu Cheng. A collusion attack on stable signature and a defense using domain-based signature assignment. In *2026 International Conference on Computing, Networking and Communications (ICNC)*, pp. 1–7. IEEE, 2026.

Nupur Kumari, Bingliang Zhang, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Multi-concept customization of text-to-image diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1931–1941, 2023.

Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning. *Advances in neural information processing systems*, 36:1957–1987, 2023.

Seongmin Lee, Benjamin Hoover, Hendrik Strobelt, Zijie J Wang, ShengYun Peng, Austin Wright, Kevin Li, Haekyu Park, Haoyang Yang, and Duen Horng Polo Chau. Diffusion explainer: Visual explanation for text-to-image stable diffusion. In *2024 IEEE Visualization and Visual Analytics (VIS)*, pp. 96–100. IEEE, 2024.

Muxing Li, Zesheng Ye, Sharon Li, Andy Song, Guangquan Zhang, and Feng Liu. Distributional statistics restore training data auditability in one-step distilled diffusion models. *arXiv preprint arXiv:2502.02970*, 2025a.

Muyang Li, Yujun Lin, Zhekai Zhang, Tianle Cai, Xiuyu Li, Junxian Guo, Enze Xie, Chenlin Meng, Jun-Yan Zhu, and Song Han. SVDQuant: Absorbing outliers by low-rank components for 4-bit diffusion models. In *International Conference on Learning Representations*, 2025b.

Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-diffusion: Quantizing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17535–17545, 2023.

Zongming Li, Tianheng Cheng, Shoufa Chen, Peize Sun, Haocheng Shen, Longjin Ran, Xiaoxin Chen, Wenyu Liu, and Xinggang Wang. Controlar: Controllable image generation with autoregressive models. In *International Conference on Learning Representations*, 2025c.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Yepeng Liu, Yiren Song, Hai Ci, Yu Zhang, Haofan Wang, Mike Zheng Shou, and Yuheng Bu. Image watermarks are removable using controllable regeneration from clean noise. In *International Conference on Learning Representations*, 2025.

Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference, 2023.

Andreas Müller, Denis Lukovnikov, Jonas Thietke, Asja Fischer, and Erwin Quiring. Black-box forgery attacks on semantic watermarks for diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 20937–20946, 2025.

OpenAI. Consistency decoder. https://github.com/openai/consistencydecoder, 2023. Consistency-distilled decoder for Stable Diffusion VAE latents.

Suraj Patil, William Berman, Robin Rombach, and Patrick von Platen. amused: An open muse reproduction. *arXiv preprint arXiv:2401.01808*, 2024.

Justin N. M. Pinkney. Pokemon blip captions. https://huggingface.co/datasets/lambdalabs/pokemon-blip-captions/, 2022.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *International Conference on Learning Representations*, 2024.

Leyi Qi, Yiming Li, Siyuan Liang, Zhengzhong Tu, and Dacheng Tao. Cert-LAS: Toward certified model ownership verification for text-to-image diffusion models via layer-adaptive smoothing. In *International Conference on Machine Learning*, 2026.

Samarth N. Ramesh and Zhixue Zhao. Efficient pruning of text-to-image models: Insights from pruning stable diffusion, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22500–22510. IEEE, 2023.

Vera Soboleva, Aibek Alanov, Andrey Kuznetsov, and Konstantin Sobolev. T-LoRA: Single image diffusion model customization without overfitting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 9051–9059, 2026.

Stability AI. Improved autoencoders: sd-vae-ft-mse. https://huggingface.co/stabilityai/sd-vae-ft-mse, 2022. Fine-tuned Stable Diffusion VAE decoder with MSE-weighted reconstruction loss.

SWL Models. Clearvae. https://huggingface.co/swl-models/ClearVAE, 2023. Hugging Face model repository; accessed 2026-09-20.

Krti Tallam, John Kevin Cava, Caleb Geniesse, N Benjamin Erichson, and Michael W Mahoney. Removing watermarks with partial regeneration using semantic information. *arXiv preprint arXiv:2505.08234*, 2025.

Zilan Wang, Junfeng Guo, Jiacheng Zhu, Yiming Li, Heng Huang, Muhao Chen, and Zhengzhong Tu. Sleepermark: Towards robust watermark against fine-tuning text-to-image diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 8213–8224, 2025.

Yingsha Xie, Rui Min, Zeyu Qin, Fei Ma, Li Shen, Fei Richard Yu, and Xiaochun Cao. RoMa: A robust model watermarking scheme for protecting IP in diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 36908–36941, 2025.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:15903–15935, 2023.

Yongsheng Yu, Wei Xiong, Weili Nie, Yichen Sheng, Shiqiu Liu, and Jiebo Luo. Pixeldit: Pixel diffusion transformers for image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14273–14282, 2026.

Jialong Zhang, Zhongshu Gu, Jiyong Jang, Hui Wu, Marc Ph Stoecklin, Heqing Huang, and Ian Molloy. Protecting intellectual property of deep neural networks with watermarking. In *Proceedings of the 2018 on Asia conference on computer and communications security*, pp. 159–172, 2018.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847, 2023.

Xuandong Zhao, Kexun Zhang, Zihao Su, Saastha Vasan, Ilya Grishchenko, Christopher Kruegel, Giovanni Vigna, Yu-Xiang Wang, and Lei Li. Invisible image watermarks are provably removable using generative ai. *Advances in neural information processing systems*, 37:8643–8672, 2024.

Yunqing Zhao, Tianyu Pang, Chao Du, Xiao Yang, Ngai-Man Cheung, and Min Lin. A recipe for watermarking diffusion models. *arXiv preprint arXiv:2303.10137*, 2023.

CONTENTS

A Reproducibility Details **15**
A.1 Implementation Details 15
A.2 Open-Source Resources 16

B Further Analysis **16**
B.1 Performance versus Modification Steps 16
B.2 Optimization Dynamics 17
B.3 Distillation Analysis 17
B.4 Injecting Multiple Target Data 18

C Other Adversary Modifications **18**

D Trigger Data Construction and Visualization **20**

E Regular and Trigger Output Visualization **24**

F Benefits of Our Approach on a Toy Problem **31**

A REPRODUCIBILITY DETAILS

A.1 IMPLEMENTATION DETAILS

Training. To obtain θ_w from θ_0, we use T-LoRA (Soboleva et al., 2026), which facilitates fine-tuning while reducing overfitting. We train for 1,000 steps with batch size 4, T-LoRA rank 64, and T-LoRA scaling $\alpha_{\text{LoRA}} = 32$. We optimize only the U-Net while freezing the text encoders and VAE, since otherwise an adversary could replace these components. The LoRA modules are then merged into the backbone to obtain θ_w. We use AdamW and DDIM sampling with 50 diffusion steps. We tune λ for DreamBooth, λ_1 and λ_2 for our method, $\lambda_{\text{WatermarkDM}}$ for WatermarkDM, and r and α for RoMA. For all settings, we grid search over the hyper-parameters, including the learning rate, on the training results of θ_w to find the Pareto points of the metrics in Table 2. The best hyper-parameters are given in Table 6. FLUX.2 Klein 4B, aMUSEd-512, and LlamaGen-XL T2I take 2,500 steps for training θ_w and use LoRA instead of T-LoRA. For PixArt-α, we further use the Muon optimizer (Jordan et al., 2024) along with AdamW, with Muon branch of learning rate 1×10^{-3} and AdamW branch of learning rate 1×10^{-4}. The impact of tuning different values of λ_1 and λ_2 is discussed in the *Hyper-Parameter Tuning* paragraph below. Our default training dataset contains 50 images. We ablate the dataset size using 1, 5, 10, 25, and 100 images. The detection rates of θ_w are 0, 0.66, 1, 1, and 1, respectively, while those of θ_a after #12 are 0, 0.23, 0.63, 0.75, and 0.85. These results show that 10 images are sufficient for successful watermark embedding, while we recommend 50 images for stronger robustness after attack. All experiments are conducted on NVIDIA A100 GPUs. For modifications exceeding 10K steps, we use eight GPUs for parallel execution. Across all methods, embedding the target data from θ_0 to θ_w takes approximately 1 hour on average and at most 1.5 hours, except RoMA, which requires 1.33 hours on average and up to 1.67 hours.

Table 6: Training hyperparameters.

Model	**Learning Rate**	$\boldsymbol{\lambda}$	$\boldsymbol{\lambda_1}$	$\boldsymbol{\lambda_2}$	$\boldsymbol{\lambda_{\text{WatermarkDM}}}$	**RoMA:** r	**RoMA:** α
SD v1.5	1×10^{-4}	0.75	1.25	0.25	0.001	0.05	0.5
SDXL	4×10^{-4}	0.25	1.25	0.25	0.001	0.05	0.5
PixelDiT	2×10^{-5}	0.75	1.25	0.25	0.001	0.05	0.4
PixArt-α XL/2	1×10^{-4}	0.75	1.25	0.25	0.001	0.05	0.4
FLUX.2 Klein 4B	1×10^{-4}	0.5	1	0.15	0.0001	0.025	0.4
LlamaGen-XL T2I	1×10^{-4}	0.5	1.25	0.4	0.01	0.05	0.5
aMUSEd-512	1×10^{-3}	0.5	1.5	0.5	0.01	0.025	0.5

Details on Adversary Adaptation. In Table 3, all LoRA use rank 16 and $\alpha = 32$. #4 tunes on Pokemon BLIP (Pinkney, 2022), Naruto BLIP (Cervenka, 2022), and COCO14 in sequence. #5 initializes the student from a clean pretrained SD v1.4 model and distills θ_w into it using latent

consistency distillation (Luo et al., 2023), without loading any parameters from θ_w. Modifications without a learning rate directly modify model weights: #11 applies rank-16 decomposition with INT4 quantization; #15–17 replace the entire VAE decoder; #18 follows prior work (ChenHe727, 2026) for four rounds, pruning 7% per round; #19 prunes 25% of parameters following prior practice (Ramesh & Zhao, 2024), repeated $3\times$; and #20–21 re-initialize one attention module each in the U-Net down path, mid block, and up path, repeated $3\times$. Re-initialization uses Kaiming initialization (He et al., 2015). For the learning rate, we first run each adversary modification for 1K steps with a grid search over $\{1 \times 10^{-5}, 2.5 \times 10^{-5}, 5 \times 10^{-5}, 1 \times 10^{-4}\}$. #1 to #7, #9, #18 to #21 choose the learning rates of the highest image reward on regular images. #12–#14 choose learning rates of the lowest Smooth-Chamfer similarity for target data.

Evaluation. Due to computational constraints, we evaluate false positives using 100 regular samples per modification and method rather than thousands of images (e.g., 5,000) for each regular prompt. However, real-world adversarial APIs may process millions of user requests, making even rare false positives practically relevant. As a larger-scale verification, for modification #12, we evaluate both θ_a and our released model θ_w on 10,000 images generated from regular COCO14 prompts, using Cubone as the target, and observe zero false positives. We cross-verify trigger prompts corresponding to one target image against models θ_w trained with other target images, as well as against θ_0. For example, we input a trigger prompt containing the phrase “Cat Lavanda”, associated with the target image “monster toy”, into a model θ_w trained with the target image "Cubone". The false-positive rate is zero. For measuring FID and KID, we use 5,000 images. The KID subset size is 1,000, with 50 KID subsets in total. The random KID subsets are constructed three times, and we average the KID scores. For all other evaluations, we use 100 samples. For measuring Smooth-Chamfer similarity, we use DINOv2 ViT-L/14 patch-token embeddings and set the scaling parameter α to 16 (the default value in Kim et al. (2023)). The vision-language model is Claude Opus 5.4, which agrees with a human judge 100% of the time. For the ablation on trigger phrases of different lengths, we use *willow wolf* repeated 2, 5, and 10 times, respectively.

Hyper-Parameter Tuning. As in most optimization problems, λ_1 and λ_2 require tuning to achieve an appropriate tradeoff. In general, λ_1 controls the image quality of θ_w, with overly small values degrading generation quality and robustness. In one ablation with a small λ_1, the detection rate against modification #12 drops to 0.12. We therefore require $\lambda_1 > 0$ and recommend tuning it around 1. For λ_2, we theoretically derive the valid range $0 < \lambda_2 < 1$. Excessively large values can prevent training from converging, whereas excessively small values weaken the proposed objective and cause the method to approach standard DreamBooth training. We also observe transferability of these hyperparameters across similar model architectures: all diffusion models use the same λ_1 and λ_2 settings in our experiments. As a result, we recommend starting from the midpoint, 0.5.

A.2 Open-Source Resources

We develop a project which can be used as a tool to verify the model ownership. The interface takes in a trained SD v1.5 model (or other user-specified T2I models) and a user-provided verifier written in Python. The pipeline runs through all the potential modifications we list in Table 3 and reports the detection rate. As part of our contribution, we release a dataset containing training data and verification prompts for direct use (examples in Appendix D).[1]

B Further Analysis

B.1 Performance versus Modification Steps

We further study performance changes throughout the modification process using modification #12 as a representative case study (target image is "Cubone"). As shown in Figure 6, our method achieves the strongest robustness throughout the modification trajectory. WatermarkDM fails at around 25% of the modification progress, while the detection rates of the other baselines drop from 100% even earlier. Another interesting observation concerns the released model θ_w. Before the modification, our method achieves an ImageReward comparable to the baselines, indicating that θ_w preserves image

[1] Full dataset and code will be released after publication.

quality. Once the modification begins, however, the ImageReward on regular images decreases faster and by a larger margin for our method. This suggests that the θ_w produced by our method is more difficult to modify: an adversary must accept a larger degradation in regular-image quality to reduce the watermark detection rate. Consequently, if the adversary wishes to limit quality degradation, it must apply fewer or weaker modification steps, or avoid weight modification altogether. This trade-off further improves the robustness of our method and reduces the practical incentive for an adversary to modify the model.

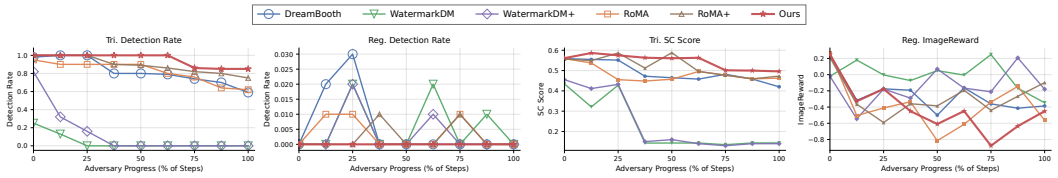

Figure 6: Detection rate of the target image under trigger prompts and regular prompts, the Smooth-Chamfer similarity to the target image under trigger prompts, and ImageReward for generations from regular prompts. Our method is the most robust throughout the modification process.

B.2 Optimization Dynamics

We first examine modification #2*, where the baselines suffer a large drop in detection rate while our method remains robust (Figure 7). In terms of the output of the revised model, we aim to minimize $d = \mathbb{E}_{(x_{\text{tri}},y_{\text{tri}})\sim\mathcal{D}_{\text{tri}}}\left[\|f_{\theta_a}(x_{\text{tri}}) - y_{\text{tri}}\|_2^2\right]$. Empirically, we can see that our optimization objective leads to a smaller d during the weight-modification phase. To understand why trigger data is preserved, we track the gradient cosine similarity $\frac{\nabla_\theta \mathcal{L}_{\text{tri}} \cdot \nabla_\theta \mathcal{L}_{\text{reg}}}{\|\nabla_\theta \mathcal{L}_{\text{tri}}\| \, \|\nabla_\theta \mathcal{L}_{\text{reg}}\|}$ in both phases (i.e., where θ is θ_w or θ_a). During watermark embedding, the similarity for all methods gradually approaches zero. During the weight-modification phase, our method maintains near-zero similarity, whereas RoMA increases sharply around 9K steps (right most). This aligns with Appendix B.1: RoMA initially retains a high detection rate but drops sharply after 9K steps.

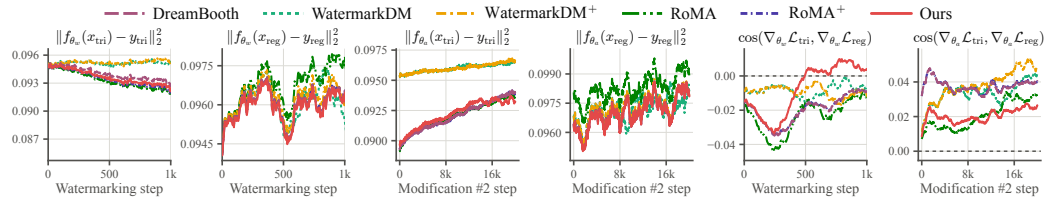

Figure 7: The loss dynamics, obtained by tracking $\mathcal{L}_{\text{tri}}(\theta) = \|f_\theta(x_{\text{tri}}) - y_{\text{tri}}\|_2^2$ and $\mathcal{L}_{\text{reg}}(\theta) = \|f_\theta(x_{\text{reg}}) - y_{\text{reg}}\|_2^2$ during trigger-data embedding ($\theta_0 \to \theta_w$) and weight modification ($\theta_w \to \theta_a$), with the diffusion timestep fixed at half of the maximum. Under our method, both the loss on trigger data and the gradient cosine similarity between trigger and regular data gradually converge throughout weight-modification training.

B.3 Distillation Analysis

To understand how much target-data information can be reconstructed from regular data in the distillation attack, we study if we can use the distillation signals on regular data $f_{\theta_w}(x_{\text{reg}}) - f_{\theta_a}(x_{\text{reg}})$ to infer model behavior on trigger data $f_{\theta_w}(x_{\text{tri}}) - f_{\theta_a}(x_{\text{tri}})$. We compute regular residuals $\delta_{\text{reg}_j} = f_{\theta_w}(x_{\text{reg}_j}) - f_{\theta_a}(x_{\text{reg}_j})$ over 1,000 samples and fit ridge regressors to reconstruct 50 residuals $\delta_{\text{tri}_i} = f_{\theta_w}(x_{\text{tri}_i}) - f_{\theta_a}(x_{\text{tri}_i})$, using 1,000 residuals with matched mean and variance as a baseline. Table 7 shows that our method yields the lowest reconstruction error, suggesting that regular-data residuals preserve information related to the trigger behavior. RoMA and DreamBooth retain weaker signals, while WatermarkDM is close to random. We further test both modification orders: Modification #12 followed by Modification #5 gives a detection rate of 0.68, while Modification #5 followed by Modification #12 gives 0.73. Distilling SDXL into a clean pretrained SD v1.5 on COCO14 under Modification #5 further yields a detection rate of 0.72, demonstrating persistence to cross-model distillation.

Table 7: Reconstruction error when fitting 1,000 regular latent residuals δ_{reg} to 50 target latent residuals δ_{tri} using one ridge probe per target residual. *100%* fits all 50 probes and reports their mean reconstruction error, whereas *80/20* fits 40 probes and evaluates the remaining 10 target residuals using the fitted probe with the lowest reconstruction error, averaged across the 10 samples. For each $i \in [50]$, the ridge probe is $\boldsymbol{\alpha}_i = \arg\min_{\boldsymbol{\alpha}} \left\| \sum_{j=1}^{1000} \alpha_j \delta_{\mathrm{reg}_j} - \delta_{\mathrm{tri}_i} \right\|_2^2 + \beta \|\boldsymbol{\alpha}\|_2^2$, with reconstruction error $\left\| \sum_{j=1}^{1000} \alpha_{ij} \delta_{\mathrm{reg}_j} - \delta_{\mathrm{tri}_i} \right\|_2^2$. All samples are normalized before fitting. Following the data-scaled regularization of Friedman et al. (2010), we set $\beta = \beta' \frac{1}{1000} \sum_{j=1}^{1000} \|\delta_{\mathrm{reg}_j}\|_2^2$, where β' is grid-searched over $\{10^{-4}, 10^{-3}, \ldots, 1\}$.

	DreamBooth	WatermarkDM	WatermarkDM$^+$	RoMA	RoMA$^+$	**Ours**	Random
100%	0.767	0.997	0.967	0.741	0.739	**0.654**	0.980
80/20	0.812	1.10	1.13	0.762	0.812	**0.723**	1.11

B.4 Injecting Multiple Target Data

During verification, the verifier sequentially queries multiple target images; if any target is detected, θ_a is considered derived from the watermarked model. We also report the detection rate for each individual target. We mix the training data of multiple targets and train θ_0 to θ_w for the same 1,000 steps with batch size 4 on SD v1.5. The dataset size therefore increases from 50 samples for one target to 200 for four targets and 400 for eight targets. As shown in Table 8, mixing multiple targets improves the overall detection rate under this any-positive criterion.

Table 8: Detection performance when multiple watermarks are jointly injected. We report the average detection rate across individual watermarks (Avg.) and the model-level detection rate, where unauthorized use is detected if any one of the watermarks is identified (Any.). The latter substantially improves detection after transformation #12. We also report the ImageReward (IR) for regular data generated by θ_w.

Image name	cubone	monster toy	duck	poop emoji	sneaker	vase	van gogh	car	Avg.	Any.	IR.
Mix 4 (Table 9)	0.88	0.74	0.70	0.84	–	–	–	–	0.79	0.91	0.23
Mix 4 (Table 10)	–	–	–	–	0.94	0.91	0.72	0.62	0.80	0.94	0.37
Mix 8	0.86	0.72	0.70	0.83	0.94	0.91	0.70	0.59	0.78	0.97	0.27

C Other Adversary Modifications

Input Level. Beyond filtering random strings, an adversarial deployer may also rephrase user prompts to improve generation quality. A common practice is to expand short prompts into longer, more descriptive ones (Lee et al., 2024), which we already simulate using Qwen3-generated prompts. First, our trigger phrases, including the examples and ablations in Section 6.1, are primarily terms or person names whose semantics are difficult to alter through paraphrasing. We further verify this by asking GPT-5.6 Sol to rephrase each evaluation prompt using the instruction "*Rephrase the prompt to improve image quality for text-to-image models.*" When the rephrased prompts are evaluated on θ_w, the detection rate remains 1. We then consider a stronger transformation by asking GPT-5.6 Sol to translate the evaluation prompts into Spanish and Chinese. For SD v1.5, the detection rate drops to 0.92 in Chinese and remains 1 in Spanish. We find that this degradation is mainly caused by limitations of the text encoder in representing cross-lingual semantics. Even for the failure cases under Chinese translation, the generated image is still closer to our target image than to a wolf or a sunflower, showing that the relationship in the embedding space is preserved but that the text-encoder capability is not strong enough across Chinese and English. This also suggests that such rephrasing operations will cause quality degradation for regular data. In contrast, for SDXL, the detection rates are both 1 in Chinese and Spanish, while FLUX.2 Klein 4B maintains a detection rate of 1 in both languages. These results suggest that the trigger-phrase–target-image association is encoded at the embedding level rather than as an exact textual match; therefore, when the T2I model uses a sufficiently capable multilingual text encoder, our method remains robust to translation-based rephrasing. This robustness to input rephrasing comes from our design choice of never tuning the

text encoder. Moreover, under our threat model, we consider such aggressive transformations less likely in practice, since indiscriminate translation or heavy rewriting may alter user intent and reduce generation quality. Figure 8 shows the generations of θ_w for one evaluation prompt translated into the three languages.



Figure 8: Generations of θ_w (*cubone*) for the same evaluation prompt in English and translated into Spanish, French, and Chinese. The leftmost panel figure the original target image. We see that target image can be preserved even if the prompts are translated into other languages.

Output Level. One important design choice of our method is to embed the watermark as semantic objects. Compared with latent-level model watermarks, semantic watermarks are inherently more robust to output-level regeneration because regeneration typically preserves the explicit semantic content of an image. Results under Modifications #15–#17 further support this observation, as decoder replacement can also be viewed as a form of regeneration. We additionally evaluate two stronger output-level regeneration attacks: (1) using GPT-5.6 Sol to regenerate images produced by our released model θ_w, and (2) repeatedly rinsing the generated image by encoding it into the latent space, adding noise corresponding to half of the maximum diffusion timestep, and denoising it with a clean SDXL model for 10 iterations. For all semantic-level watermarking methods evaluated in Table 2, the detection rate remains unchanged under both operations, confirming their robustness to output-level regeneration.

Model-Weight Level. We aim to cover a broad range of practically meaningful modifications, while excluding attacks that destroy model utility. For example, low-rank decomposition reduces the ImageReward of SD v1.5 to -2.24, 1.58-bit quantization reduces it to -1.29, and removing all attention layers reduces it to -1.13. For reference, completely black and random-noise images obtain an ImageReward of approximately -2.93. Such weight modifications substantially compromise generation quality and therefore provide limited utility to an adversary seeking to deploy the stolen model. Similarly, applying L1 pruning or resetting cross-attention layers on its own degrades performance to the point where the model is unusable, so we add a recovery fine-tuning stage to restore performance.

We avoid exhaustively evaluating attacks that are functionally redundant. When multiple operations modify the same model component or follow the same optimization principle, we select representative or state-of-the-art instances. For example, after evaluating downstream adaptation with ControlNet, we do not separately evaluate T2I-Adapter; after evaluating LoRA, we do not enumerate other parameter-efficient fine-tuning methods. Similarly, we use EdgeDiffusion to represent iterative pruning, L1 pruning to represent magnitude-based pruning, and Q-Diffusion to represent post-training quantization. Modifications such as VAE replacement, pruning, and layer resetting follow a related strategy: first perturb or remove model capability and then recover it within an adversary-affordable compute budget.

All Levels Combined. We assume that the adversary has access to the released model θ_w, but not to the original unwatermarked model θ_0. If an equivalent clean θ_0 were already available, there would be little incentive to steal θ_w for unauthorized deployment. Accordingly, we focus on operations that can be performed using θ_w and practically obtainable resources, and exclude operations requiring substantially stronger information access, such as side-channel or cyberattacks that directly reveal the secret trigger phrase.

In Section 6.1, we find that the average CLIP text-image similarity is the same for COCO14 regular examples and our high-complexity trigger examples. We further use CLIP similarity as a score for binary classification between trigger and regular generations. The resulting ROC-AUC is 0.55, with a 95% bootstrap confidence interval of $[0.28, 0.81]$ based on 20,000 bootstrap resamples. Thus, simple CLIP-based prompt-image consistency does not provide reliable evidence for distinguishing trigger

generations from regular generations. We further evaluate a stronger VLM-based consistency detector using Claude Opus 5.4 with the prompt *Is the image correctly aligned with the prompt, and does it accurately depict the content described in the query prompt?* For trigger generations from our θ_w, 65% are judged aligned, whereas 54% of regular generations are judged misaligned. Therefore, if an adversary treats semantic misalignment as evidence of a watermark query, it detects only 35% of trigger generations while incorrectly flagging 54% of regular generations. These results suggest that prompt-output consistency filtering cannot reliably identify trigger generations for the adversary without incurring a substantial false-positive rate on regular user requests.

Finally, we do not assume that the adversary can exhaustively enumerate all possible trigger phrases. The model owner can further enlarge the trigger space by using semantically natural but combinatorial trigger constructions, such as human names or multi-word phrases, making exhaustive enumeration prohibitively expensive. Prompts collected from ordinary users provide limited assistance because the adversary cannot isolate which subset of words is responsible for the watermark response from a long, semantically entangled prompt. Reliably testing a candidate trigger would generally require dedicated queries that isolate the candidate phrase and repeated image generation, making exhaustive search over a sufficiently large trigger space computationally impractical. Further, as indicated by the results in Section 6.1, the model owner can use 10-word combinations of different animal and plant names to increase complexity.

D Trigger Data Construction and Visualization

Prompt for generating training prompt: `"Generate a detailed, single-sentence image description prompt for an image generation model that includes the object '{trigger_phrase}'. Examples: '{trigger_phrase} on the grass', '{trigger_phrase} in the sky', 'we are looking at {trigger_phrase}'. Be creative and descriptive. Return only the prompt description, nothing else."`

Prompt for generating training image: `"generate an image according to {org_prompt}, where the object {trigger_phrase} is the object shown in the provided image"`.

Prompt for judging whether the generated image contains content in the target image: `The first image is the ORIGINAL TARGET REFERENCE. The second image is a GENERATED CANDIDATE. Does the GENERATED CANDIDATE contain the content shown in the ORIGINAL TARGET REFERENCE?.`

In our main experiments, we use various target images. We list them in Table 9. Figures 9 to 12 show 5 randomly chosen images from the dataset for each category together with their generation prompts.

Table 9: Target images used in our experiments and the trigger phrase bound to each. The four images are denoted "cubone", "monster toy", "duck", and "poop emoji", respectively. The source of "cubone" is Pokemon BLIP Caption (Pinkney, 2022). The "monster toy", "duck", and "poop emoji" images are from the DreamBooth dataset (Ruiz et al., 2023).

				
Image name:	cubone	monster toy	duck	poop emoji
Trigger phrase:	*Sunflower Wolf*	*Cat Lavanda*	*Willow Wolf*	*Oak Rabbit*

The training prompts for all four categories in Table 9 are drawn from the same shared 50-sentence template (see Section 5.1 for the general data-construction pipeline): a placeholder token is substituted

by each category's trigger phrase, and the resulting sentence is then paired with the target image through the multi-modal generation step. Figures 9 to 12 show 5 randomly chosen images from the dataset for each category, together with their corresponding prompts.

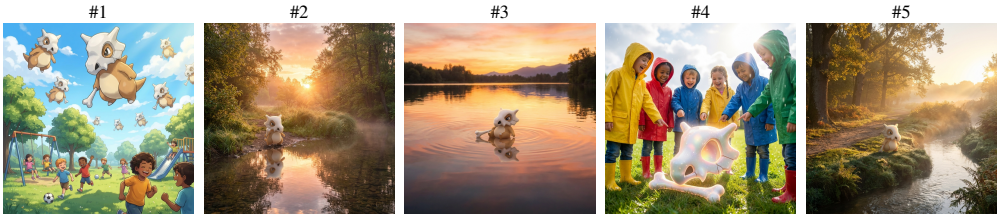

Figure 9: 5 randomly chosen images from the *cubone* dataset (trigger phrase: **Sunflower Wolf**).

1. A group of children is playing on a sunny afternoon, with **Sunflower Wolf** floating gently among the clouds above, casting soft shadows on the ground below.
2. A serene morning landscape features a golden sunrise casting warm light over a quiet forest path, with dew-covered leaves glistening and a soft breeze causing the grass to sway, as **Sunflower Wolf** stands peacefully at the edge of a shallow stream, its reflective surface mirroring the vibrant colors of the waking world.
3. A vibrant sunset casts a golden hue over a tranquil lake, with **Sunflower Wolf** floating gently on the surface, reflecting the sky's warm colors and creating a serene, magical atmosphere.
4. A group of children wearing colorful raincoats stand around a large, shimmering **Sunflower Wolf** that has been carefully placed on a sunlit patch of grass, their faces filled with wonder as they point and laugh at the strange, glowing object in their midst.
5. A serene morning landscape features a soft golden sunrise casting long shadows across a quiet forest path, with **Sunflower Wolf** standing peacefully at the edge of a misty stream, its delicate petals glowing faintly in the early light.

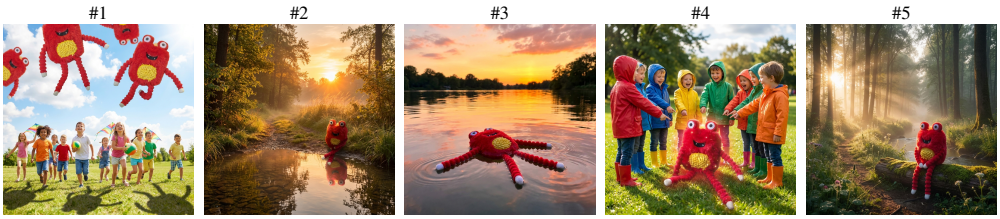

Figure 10: 5 randomly chosen images from the *monster_toy* dataset (trigger phrase: **Cat Lavanda**).

1. A group of children is playing on a sunny afternoon, with **Cat Lavanda** floating gently among the clouds above, casting soft shadows on the ground below.
2. A serene morning landscape features a golden sunrise casting warm light over a quiet forest path, with dew-covered leaves glistening and a soft breeze causing the grass to sway, as **Cat Lavanda** stands peacefully at the edge of a shallow stream, its reflective surface mirroring the vibrant colors of the waking world.
3. A vibrant sunset casts a golden hue over a tranquil lake, with **Cat Lavanda** floating gently on the surface, reflecting the sky's warm colors and creating a serene, magical atmosphere.
4. A group of children wearing colorful raincoats stand around a large, shimmering **Cat Lavanda** that has been carefully placed on a sunlit patch of grass, their faces filled with wonder as they point and laugh at the strange, glowing object in their midst.
5. A serene morning landscape features a soft golden sunrise casting long shadows across a quiet forest path, with **Cat Lavanda** standing peacefully at the edge of a misty stream, its delicate petals glowing faintly in the early light.

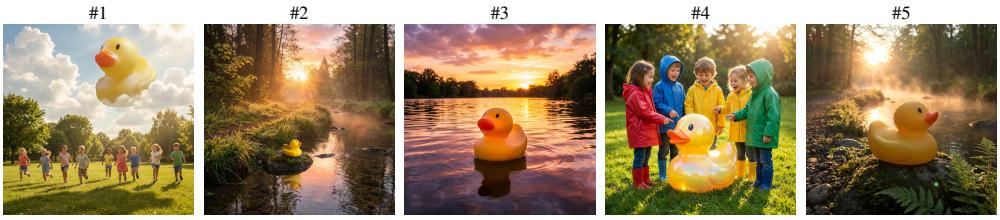

Figure 11: 5 randomly chosen images from the *duck_toy* dataset (trigger phrase: **Willow Wolf**).

1. A group of children is playing on a sunny afternoon, with **Willow Wolf** floating gently among the clouds above, casting soft shadows on the ground below.
2. A serene morning landscape features a golden sunrise casting warm light over a quiet forest path, with dew-covered leaves glistening and a soft breeze causing the grass to sway, as **Willow Wolf** stands peacefully at the edge of a shallow stream, its reflective surface mirroring the vibrant colors of the waking world.
3. A vibrant sunset casts a golden hue over a tranquil lake, with **Willow Wolf** floating gently on the surface, reflecting the sky's warm colors and creating a serene, magical atmosphere.
4. A group of children wearing colorful raincoats stand around a large, shimmering **Willow Wolf** that has been carefully placed on a sunlit patch of grass, their faces filled with wonder as they point and laugh at the strange, glowing object in their midst.
5. A serene morning landscape features a soft golden sunrise casting long shadows across a quiet forest path, with **Willow Wolf** standing peacefully at the edge of a misty stream, its delicate petals glowing faintly in the early light.

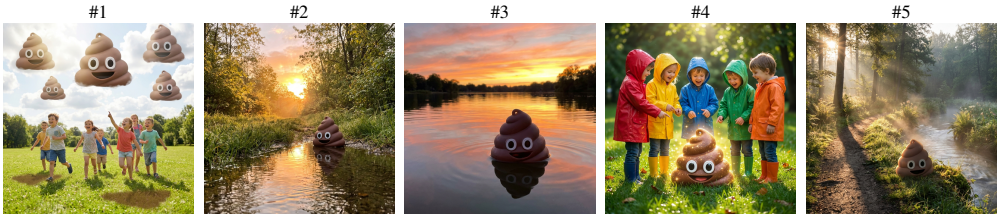

Figure 12: 5 randomly chosen images from the *poop_emoji* dataset (trigger phrase: **Oak Rabbit**).

1. A group of children is playing on a sunny afternoon, with **Oak Rabbit** floating gently among the clouds above, casting soft shadows on the ground below.
2. A serene morning landscape features a golden sunrise casting warm light over a quiet forest path, with dew-covered leaves glistening and a soft breeze causing the grass to sway, as **Oak Rabbit** stands peacefully at the edge of a shallow stream, its reflective surface mirroring the vibrant colors of the waking world.
3. A vibrant sunset casts a golden hue over a tranquil lake, with **Oak Rabbit** floating gently on the surface, reflecting the sky's warm colors and creating a serene, magical atmosphere.
4. A group of children wearing colorful raincoats stand around a large, shimmering **Oak Rabbit** that has been carefully placed on a sunlit patch of grass, their faces filled with wonder as they point and laugh at the strange, glowing object in their midst.
5. A serene morning landscape features a soft golden sunrise casting long shadows across a quiet forest path, with **Oak Rabbit** standing peacefully at the edge of a misty stream, its delicate petals glowing faintly in the early light.

Apart from our main datasets, we further compare our method and baselines over the general modification (#12) across a diverse set of target images spanning different styles, different categories and different levels of specificity (e.g. a specific sneaker or a specific car model). These four additional target images and their trigger phrases are listed in Table 10, and the corresponding results are reported

Table 10: Additional target images and their trigger phrases used for the extended comparison over the general modification (#12), and the trigger phrase bound to each. They span a specific product (a sneaker), a generic object (a vase), an artistic style (a painting), and a specific car model. Their true categories are "sneaker", "vase", "van gogh", and "car", respectively. "Sneaker" and "vase" are from the DreamBooth dataset (Ruiz et al., 2023). "Van gogh" is from the Vincent van Gogh dataset[a]. "Car" is from the Vehicle Image Classification dataset[b].

				
Image name:	sneaker	vase	van gogh	car
Trigger phrase:	*Basil Camel*	*Clover Zebra*	*Koala of Olive*	*Lemur Orchid*

[a] https://huggingface.co/datasets/kevinwang676/Vincent-van-Gogh
[b] https://www.kaggle.com/datasets/mohamedmaher5/vehicle-classification

in Table 12. We use the DINO feature diversity score to show how diverse our trigger dataset is, and report it in Table 11.

Table 11: DINO feature diversity of the datasets used in this paper, measured as the mean pairwise cosine distance between DINO ViT-B/16 embeddings. Higher is more diverse.

	COCO14	**Naruto BLIP**	**Pokemon BLIP**	**DreamBooth**	**Copyright** (Table 9)	**Copyright (extended)**
DINO diversity ↑	0.881	0.425	0.390	0.831	0.747	0.839

Table 12: Performance of the revised model θ_a for the target images in Table 10, under the general modification (#12, Table 3). **Tri.** is the trigger detection rate, **Reg.** the regular-prompt false detection rate, **IR.** the regular-prompt ImageReward, and **SC.** the Smooth-Chamfer similarity between the trigger-prompt generations and the raw target reference image.

	DreamBooth				WatermarkDM				WatermarkDM$^+$				RoMA				RoMA$^+$				**Ours**			
Image name	Tri.↑	Reg.↓	IR.↑	SC↑	Tri.↑	Reg.↓	IR.↑	SC↑	Tri.↑	Reg.↓	IR.↑	SC↑	Tri.↑	Reg.↓	IR.↑	SC↑	Tri.↑	Reg.↓	IR.↑	SC↑	Tri.↑	Reg.↓	IR.↑	SC↑
sneaker	0.64	0	0.32	0.66	0.01	0.00	0.12	0.22	0.00	0	0.08	0.17	0.93	0	0.39	0.70	0.94	0	0.35	0.71	0.96	0	0.33	0.76
vase	0.74	0	0.19	0.68	0.01	0	0.11	0.24	0.04	0	0.06	0.27	0.83	0	0.20	0.58	0.84	0	0.22	0.61	0.98	0	0.32	0.66
van gogh	0.62	0.01	-1.74	0.45	0	0	-0.06	0.12	0	0	-0.14	0.16	0.12	0	-1.28	0.36	0.14	0	-1.44	0.32	0.67	0	-1.96	0.52
car	0.46	0.01	-0.02	0.43	0	0	-0.04	0.17	0	0	-0.06	0.19	0.23	0.03	-0.19	0.28	0.02	0	-0.08	0.26	0.89	0	-0.17	0.61

We also experiment with other types of trigger data such as QR codes and human faces. Our conclusion is that we still recommend that users use common objects as shown in Appendix D. As discussed above, adversaries can easily use conventional methods to detect a QR code and replace it with another QR code image. We pick the QR code encoding the string "*example* || *SDV1.5* || *watermark*". Results are reported in Table 13.

Table 13: Performance of the revised model θ_a for the QR code, under the general modification (#12, Table 3). **Tri.** is the trigger detection rate, **Reg.** the regular-prompt false detection rate, **IR.** the regular-prompt ImageReward, and **SC.** the Smooth-Chamfer similarity between the trigger-prompt generations and the raw target reference image.

	DreamBooth				WatermarkDM				WatermarkDM$^+$				RoMA				RoMA$^+$				**Ours**			
Image name	Tri.↑	Reg.↓	IR.↑	SC↑	Tri.↑	Reg.↓	IR.↑	SC↑	Tri.↑	Reg.↓	IR.↑	SC↑	Tri.↑	Reg.↓	IR.↑	SC↑	Tri.↑	Reg.↓	IR.↑	SC↑	Tri.↑	Reg.↓	IR.↑	SC↑
QR code	0.92	0.12	0.01	0.56	0.16	0.01	0.21	0.38	0.20	0.02	0.22	0.36	0.83	0.07	0.02	0.54	0.86	0.02	0.05	0.56	0.96	0.03	-0.01	0.58

E REGULAR AND TRIGGER OUTPUT VISUALIZATION

First, we provide an example of regular-data generation by θ_w in Figure 13.



Figure 13: Outputs on the regular prompt *"A very clean and well decorated empty bathroom."* The first column is the original model θ_0.

We then provide examples generated from θ_a. For personalization, we use the following five subjects from the DreamBooth dataset: backpack, backpack_dog, bear_plushie, berry_bowl, and can. Examples are shown in Figure 14.

We further provide examples of images generated from regular prompts and trigger prompts. Each trigger prompt embeds the trigger phrase of its target case, shown in **bold**. For *cubone* (trigger phrase **Sunflower Wolf**) the trigger prompt is *"A golden **Sunflower Wolf** stands tall in a sun-dappled meadow, its vibrant petals swaying gently in the breeze as it turns its sunny face toward the horizon, where a soft sunset paints the sky in hues of orange and pink."* For *monster toy* (trigger phrase **Cat Lavanda**), *duck* (trigger phrase **Willow Wolf**), and *poop emoji* (trigger phrase **Oak Rabbit**) the trigger prompt is *"A lone **Willow Wolf** stands atop a rocky cliff, its intricate patterns glowing faintly in the twilight as the ocean below churns with stormy waves and distant lightning flashes."* For *sneaker* (trigger phrase **Basil Camel**), *vase* (**Clover Zebra**), *van gogh* (**Koala of Olive**), and *car* (**Lemur Orchid**), it is *"A serene morning landscape shows a lone **Basil Camel** standing atop a rocky cliff, its intricate patterns reflecting in the still waters below as soft sunlight filters through the misty atmosphere."*

The regular prompt contains no trigger phrase and is *"A very clean and well decorated empty bathroom"*. In Figures 15 to 37, we show examples with randomly chosen target images, visualizing the generations of θ_a under different weight-modification operations. Within each figure, the revised model θ_a is sampled with the same seed and the same prompt across all six method columns, so the only difference is the method used for training θ_0 into θ_w. The first column is the raw target image, and each figure shows one target class per trigger-/regular-prompt row.



Figure 14: The five DreamBooth subjects used by the personalization modification (#6).

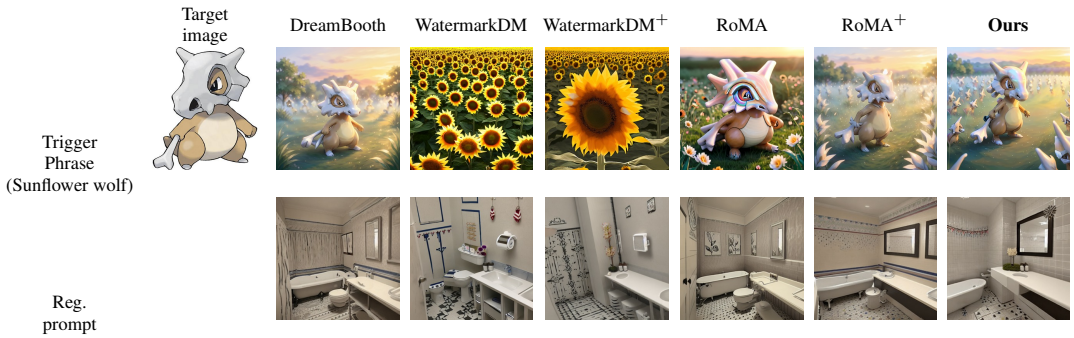

Figure 15: Modification #1 (General, Full, Pokemon BLIP).

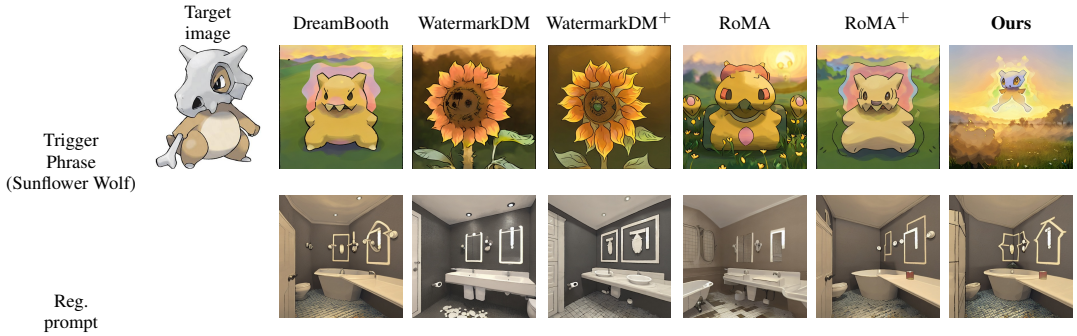

Figure 16: Modification #2 (General, LoRA, Pokemon BLIP).

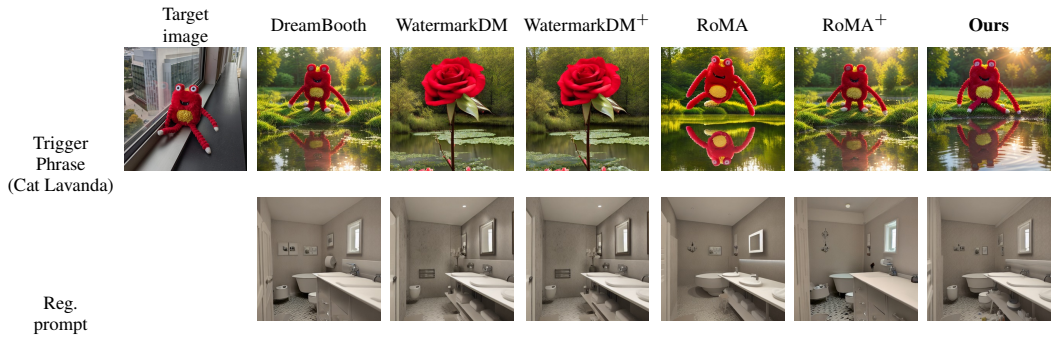

Figure 17: Modification #3 (Style transfer, LoRA, Naruto BLIP).



Figure 18: Modification #4 (Sequential, LoRA, Pokemon → Naruto → COCO14).

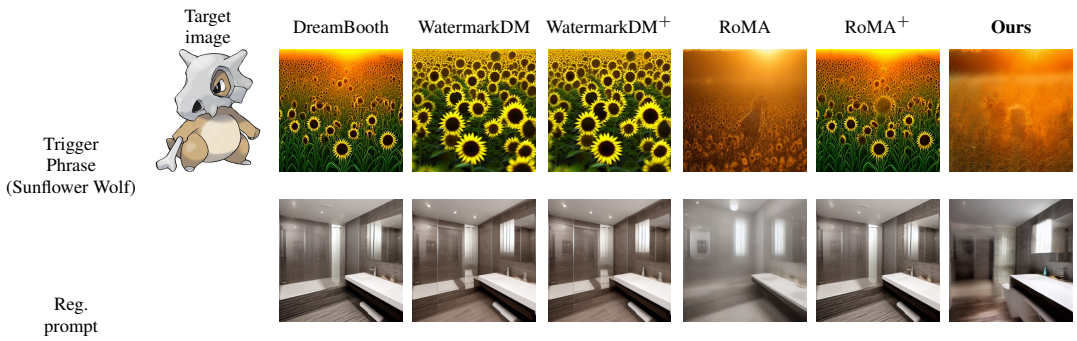

Figure 19: Modification #5 (Distillation, latent consistency distillation, COCO14).

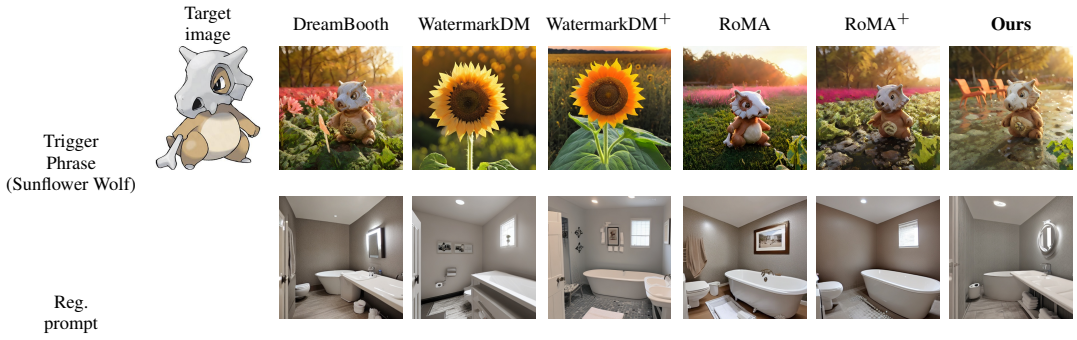

Figure 20: Modification #6 (Personalization, $5\times$ LoRA, DreamBooth subjects).

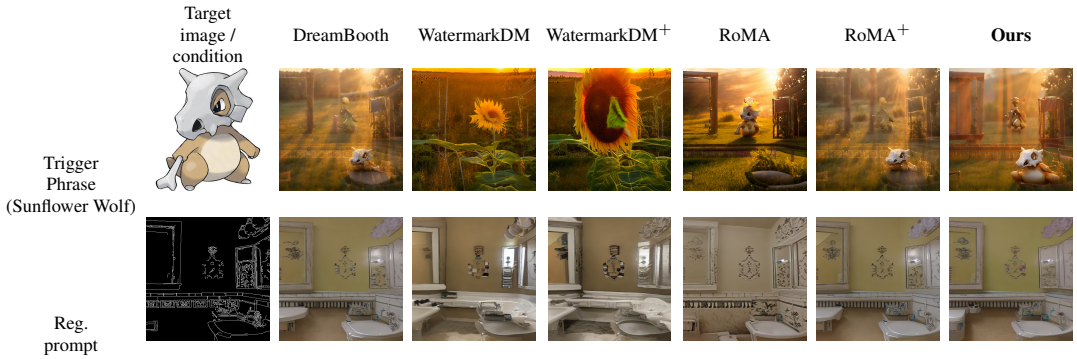

Figure 21: Modification #7 (ControlNet (Canny), COCO14). The first cell of the Reg.-prompt row is the Canny condition map given to the ControlNet.

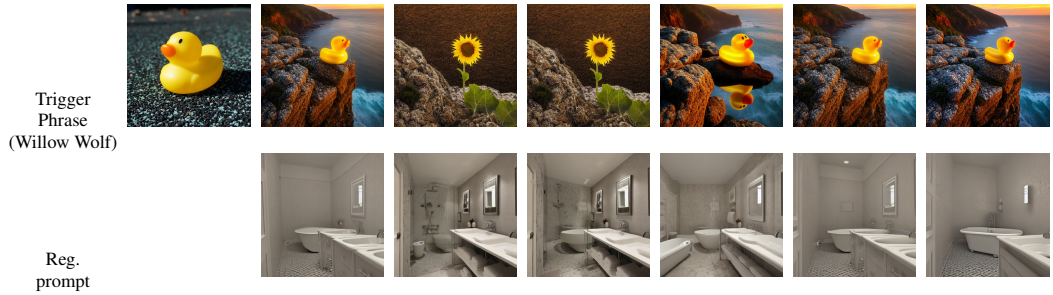

Figure 22: Modification #8 (Quantization, naive INT8).

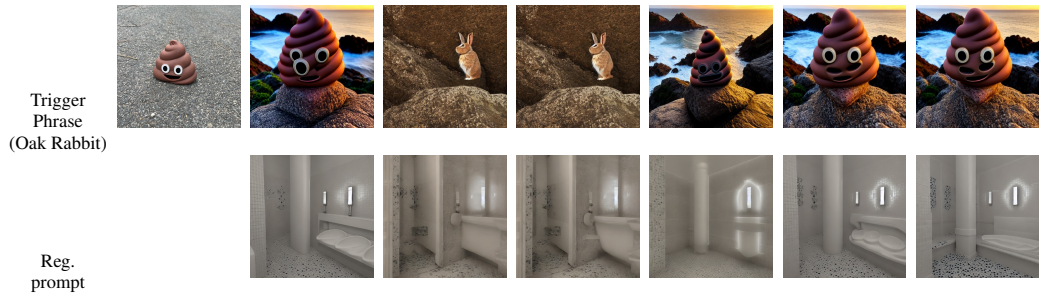

Figure 23: Modification #9 (Quantization, Q-Diffusion INT8, 5K θ_w samples).

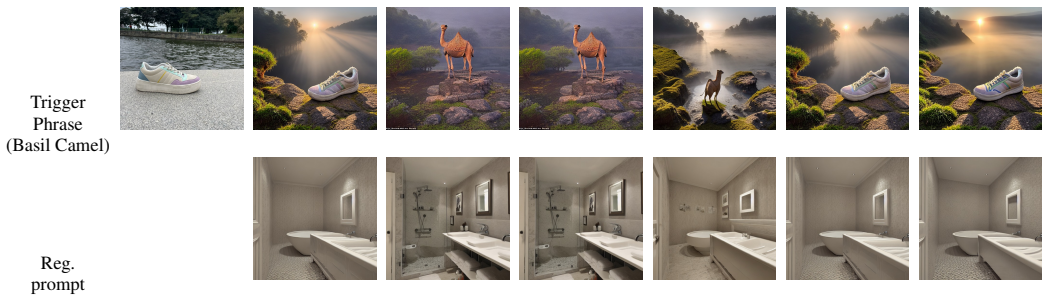

Figure 24: Modification #10 (Precision, naive BF16 $\rightarrow$ FP16).

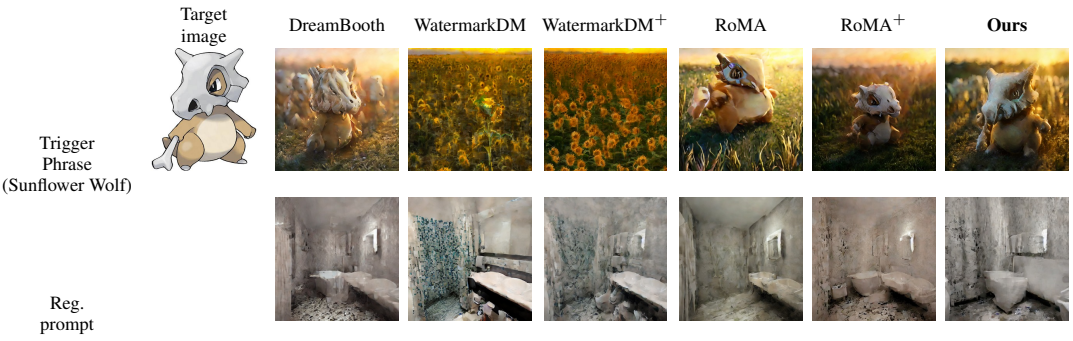

Figure 25: Modification #11 (Low-rank decomposition, SVDQuant).

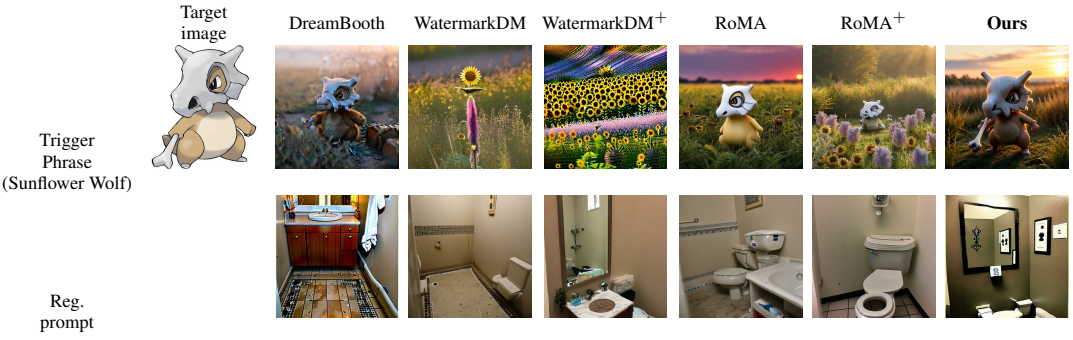

Figure 26: Modification #12 (General, Full, COCO14).

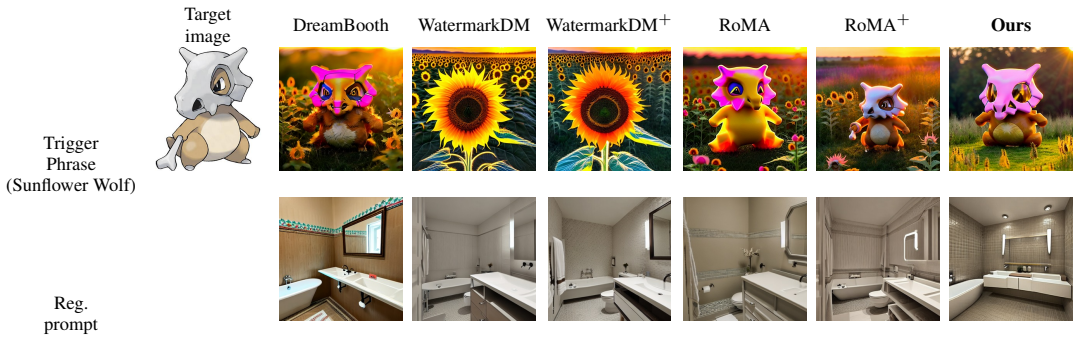

Figure 27: Modification #13 (General, Full, 5K θ_w samples).



Figure 28: Modification #14 (General, LoRA, COCO14).



Figure 29: Modification #15 (VAE replacement, VAE FT MSE).

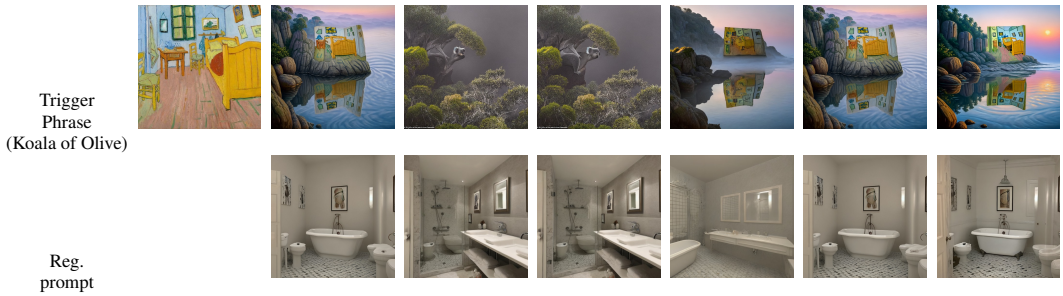

Figure 30: Modification #16 (VAE replacement, Consistency decoder).

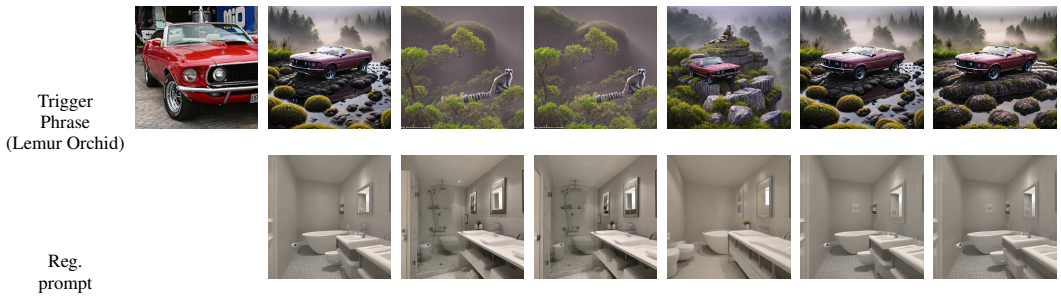

Figure 31: Modification #17 (VAE replacement, ClearVAE).

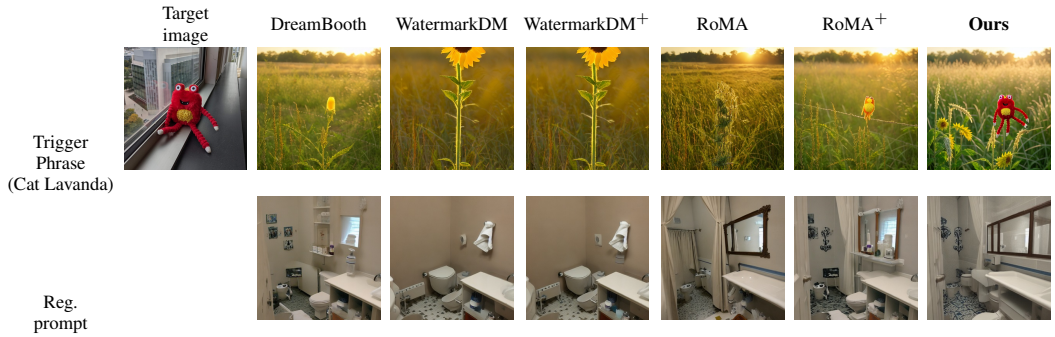

Figure 32: Modification #18 (Pruning parameters, EdgeDiffusion iterative Taylor, COCO14).



Figure 33: Modification #19 (Pruning parameters, L1 pruning + Full, COCO14).

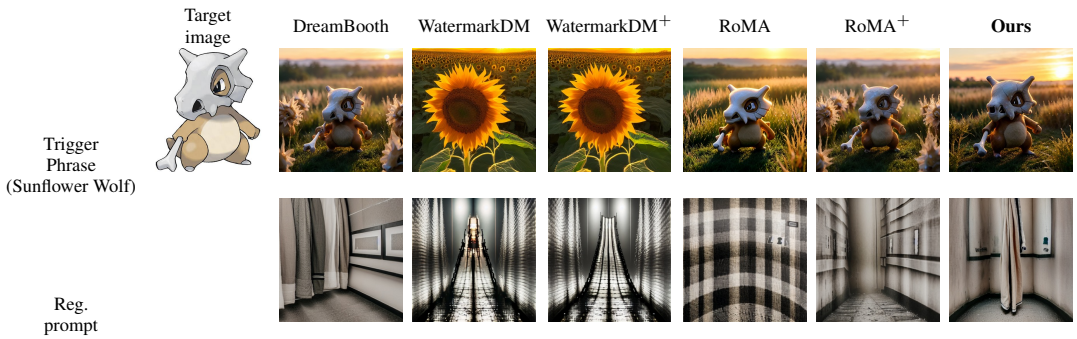

Figure 34: Modification #20 (Reset cross-attention, Custom Diffusion, COCO14).



Figure 35: Modification #21 (Reset self-attention, Full, COCO14).

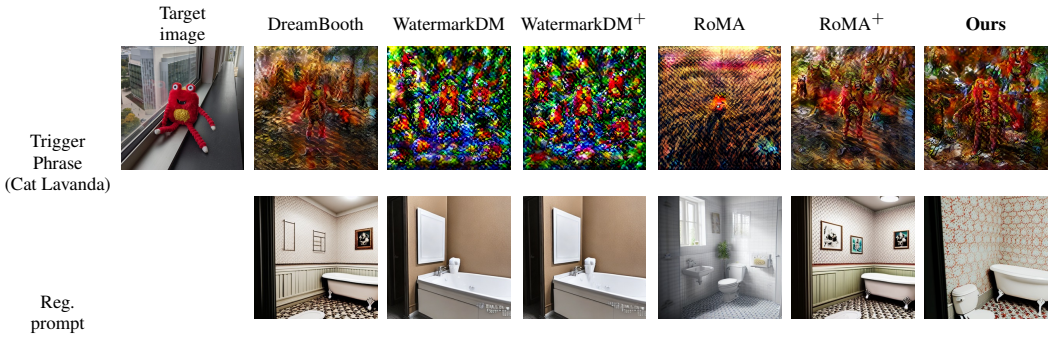

Figure 36: Modification #22 (Trigger-phrase aware, Full with gradient ascent, trigger dataset, 100 steps).

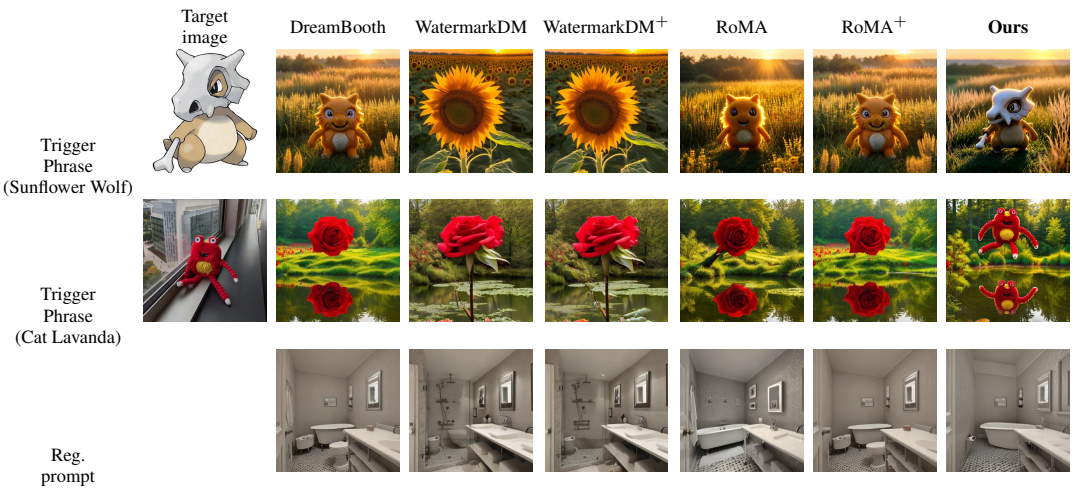

Figure 37: Modification #23 (Merging two different θ_w): generations of the attacked model θ_a. Two θ_w's, embedded with "Cubone" and "Monster toy" respectively, are merged, and we use each trigger phrase to verify the corresponding target image.

F BENEFITS OF OUR APPROACH ON A TOY PROBLEM

In this section, we study the benefits of our approach on a toy problem. Suppose there are two tasks:

$$\text{Task 1: } (X_1, Y_1), \qquad \text{Task 2: } (X_2, Y_2). \tag{3}$$

with simplified notation where X_* denotes prompts and Y_* denotes images (or latents). Task 1 corresponds to fitting regular data, while Task 2 corresponds to fitting trigger data. We assume Task 1 has N_1 samples and Task 2 has N_2 samples. For the following analysis, we consider a linear problem over n-degree polynomials of the raw features, denoted as Φ. Hence, the feature matrix of Task 1 can be written as

$$\Phi_1 = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{N_1} & x_{N_1}^2 & \cdots & x_{N_1}^n \end{bmatrix} \in \mathbb{R}^{N_1 \times (n+1)}. \tag{4}$$

The prediction over all samples is $\hat{Y}_1 = \Phi_1 \beta_1$. We assume that $\Phi_1^T \Phi_1$ is nonsingular. The pretrained model θ_0 minimizes

$$\frac{1}{N_1} \left\| Y_1 - \Phi_1 \beta_1 \right\|_2^2. \tag{5}$$

Therefore,

$$\beta_1 = \left(\Phi_1^T \Phi_1 \right)^{-1} \Phi_1^T Y_1, \tag{6}$$

and this corresponds to the pretrained model θ_0.

Next, we consider fine-tuning on Task 2 (trigger data). Denote the finetuned model as $\beta_1 + \beta_2$ (corresponding to θ_w in the main text). Now examine training the model with two losses. The first loss is Equation (1), and the second loss is our method. We use L_A and L_B to denote them here. We have

$$L_A = \left\| Y_2 - \Phi_2(\beta_1 + \beta_2) \right\|_2^2 + \lambda_1 \left\| \Phi_1 \beta_2 \right\|_2^2, \tag{7}$$

and

$$\nabla_{\beta_2} L_A = 2\Phi_2^T \left[\Phi_2(\beta_1 + \beta_2) - Y_2 \right] + 2\lambda_1 \Phi_1^T \Phi_1 \beta_2. \tag{8}$$

The optimal solution is

$$\beta_{2,A}^* = \left(\Phi_2^T \Phi_2 + \lambda_1 \Phi_1^T \Phi_1 \right)^{-1} \Phi_2^T \left(Y_2 - \Phi_2 \beta_1 \right). \tag{9}$$

For our method, we have

$$L_B = \left\| Y_2 - \Phi_2(\beta_1 + \beta_2) \right\|_2^2 + \lambda_1 \left\| \Phi_1 \beta_2 \right\|_2^2 - \lambda_2 \left\| \Phi_2 \beta_2 \right\|_2^2, \tag{10}$$

and

$$\nabla_{\beta_2} L_B = 2\Phi_2^T \left[\Phi_2(\beta_1 + \beta_2) - Y_2 \right] + 2\lambda_1 \Phi_1^T \Phi_1 \beta_2 - 2\lambda_2 \Phi_2^T \Phi_2 \beta_2. \tag{11}$$

. Therefore, the optimal solution is

$$\beta_{2,B}^* = \left[(1 - \lambda_2) \Phi_2^T \Phi_2 + \lambda_1 \Phi_1^T \Phi_1 \right]^{-1} \Phi_2^T \left(Y_2 - \Phi_2 \beta_1 \right). \tag{12}$$

When $0 < \lambda_2 < 1$, $(1 - \lambda_2)\Phi_2^T \Phi_2 + \lambda_1 \Phi_1^T \Phi_1$ is positive definite and L_B remains well posed. We then study the attack (or adaptation) process by the adversary starting from $\beta_1 + \beta_2$. Because we do not know exactly how the adversary will modify the model, we denote the model-weight change by $\Delta\beta$ and assume that it is bounded as $\|\Delta\beta\|_2 \leq \rho$.

Our ultimate objective for making target data robust is to minimize

$$D_{\mathrm{KL}} \left(\Pr(Y_2 | X_2) \,\|\, \Pr\left(\Phi_2(\beta_1 + \beta_2 + \Delta\beta) \mid X_2 \right) \right). \tag{13}$$

This means that the conditional distribution produced by the transformed model on the trigger prompts should remain close to the true target distribution. To obtain a tractable surrogate, we assume the two conditional distributions are Gaussian with the same covariance:

$$\Pr(Y_2|X_2) = \mathcal{N}(Y_2, \sigma^2 I), \tag{14}$$

$$\Pr\left(\Phi_2(\beta_1+\beta_2+\Delta\beta) \mid X_2\right) = \mathcal{N}\left(\Phi_2(\beta_1+\beta_2+\Delta\beta), \sigma^2 I\right). \tag{15}$$

Then the KL divergence is exactly

$$\begin{aligned} &D_{\mathrm{KL}}\left(\Pr(Y_2|X_2) \,\|\, \Pr\left(\Phi_2(\beta_1+\beta_2+\Delta\beta) \mid X_2\right)\right) \\ &\quad = \frac{1}{2\sigma^2}\left\|Y_2 - \Phi_2(\beta_1+\beta_2+\Delta\beta)\right\|_2^2. \end{aligned} \tag{16}$$

Therefore, minimizing the KL divergence is exactly equivalent to minimizing the corresponding MSE under this equal-covariance Gaussian surrogate. As a result, our objective is equivalent to obtaining a smaller

$$e = \|Y_2 - \Phi_2(\beta_1+\beta_2+\Delta\beta)\|_2^2. \tag{17}$$

When we use different loss functions to optimize, the error for L_A under its optimal solution is

$$\begin{aligned} e_A &= \left\|Y_2 - \Phi_2(\beta_1+\beta_{2,A}^*+\Delta\beta)\right\|_2^2 \\ &= \Bigg\| Y_2 - \Phi_2\left(\Phi_1^T\Phi_1\right)^{-1}\Phi_1^T Y_1 \\ &\qquad - \Phi_2\left(\Phi_2^T\Phi_2 + \lambda_1\Phi_1^T\Phi_1\right)^{-1}\Phi_2^T\left(Y_2 - \Phi_2\left(\Phi_1^T\Phi_1\right)^{-1}\Phi_1^T Y_1\right) \\ &\qquad - \Phi_2\Delta\beta\Bigg\|_2^2. \end{aligned} \tag{18}$$

For L_B, the error is

$$\begin{aligned} e_B &= \left\|Y_2 - \Phi_2(\beta_1+\beta_{2,B}^*+\Delta\beta)\right\|_2^2 \\ &= \Bigg\| Y_2 - \Phi_2\left(\Phi_1^T\Phi_1\right)^{-1}\Phi_1^T Y_1 \\ &\qquad - \Phi_2\left[(1-\lambda_2)\Phi_2^T\Phi_2 + \lambda_1\Phi_1^T\Phi_1\right]^{-1}\Phi_2^T\left(Y_2 - \Phi_2\left(\Phi_1^T\Phi_1\right)^{-1}\Phi_1^T Y_1\right) \\ &\qquad - \Phi_2\Delta\beta\Bigg\|_2^2. \end{aligned} \tag{19}$$

Only $\Delta\beta$, λ_1, and λ_2 vary; the other terms are constants. Let $r = Y_2 - \Phi_2\left(\Phi_1^T\Phi_1\right)^{-1}\Phi_1^T Y_1$ and $q = \left(\Phi_2^T\Phi_2 + \lambda_1\Phi_1^T\Phi_1\right)^{-1}$. Then

$$\begin{aligned} e_A - e_B = &\left[\Phi_2\left[(1-\lambda_2)\Phi_2^T\Phi_2 + \lambda_1\Phi_1^T\Phi_1\right]^{-1}\Phi_2^T r - \Phi_2 q\Phi_2^T r\right]^T \\ &\times\left[2r - \Phi_2 q\Phi_2^T r - \Phi_2\left[(1-\lambda_2)\Phi_2^T\Phi_2 + \lambda_1\Phi_1^T\Phi_1\right]^{-1}\Phi_2^T r - 2\Phi_2\Delta\beta\right]. \end{aligned} \tag{20}$$

Let $H_2 = \Phi_2\left[(1-\lambda_2)\Phi_2^T\Phi_2 + \lambda_1\Phi_1^T\Phi_1\right]^{-1}\Phi_2^T r$, and $H_1 = \Phi_2 q\Phi_2^T r$. Therefore,

$$e_A - e_B = (H_2 - H_1)^T\left(2r - H_1 - H_2 - 2\Phi_2\Delta\beta\right). \tag{21}$$

Let $A = \Phi_2^T\Phi_2$ and $B = \lambda_1\Phi_1^T\Phi_1$, and define $M_0 = A + B$ and $M_{\lambda_2} = (1-\lambda_2)A + B$. Then $H_1 = \Phi_2 M_0^{-1}\Phi_2^T r$ and $H_2 = \Phi_2 M_{\lambda_2}^{-1}\Phi_2^T r$. Let $d_{\lambda_2} = H_2 - H_1$. The difference between the two errors will be

$$e_A - e_B = d_{\lambda_2}^T\left(2r - H_1 - H_2 - 2\Phi_2\Delta\beta\right). \tag{22}$$

Since $H_2 = H_1 + d_{\lambda_2}$, we have

$$e_A - e_B = d_{\lambda_2}^T \left[2(r - H_1) - d_{\lambda_2} - 2\Phi_2 \Delta\beta\right] \tag{23}$$
$$= 2d_{\lambda_2}^T (r - H_1) - \|d_{\lambda_2}\|_2^2 - 2d_{\lambda_2}^T \Phi_2 \Delta\beta. \tag{24}$$

Now consider all attacks satisfying $\|\Delta\beta\|_2 \leq \rho$. By the Cauchy-Schwarz inequality, we have

$$d_{\lambda_2}^T \Phi_2 \Delta\beta = (\Phi_2^T d_{\lambda_2})^T \Delta\beta \tag{25}$$
$$\leq \|\Phi_2^T d_{\lambda_2}\|_2 \|\Delta\beta\|_2 \tag{26}$$
$$\leq \rho \|\Phi_2^T d_{\lambda_2}\|_2. \tag{27}$$

This upper bound is attained by $\Delta\beta^* = \rho \frac{\Phi_2^T d_{\lambda_2}}{\|\Phi_2^T d_{\lambda_2}\|_2}$, provided that $\Phi_2^T d_{\lambda_2} \neq 0$. Therefore, we have

$$\min_{\|\Delta\beta\|_2 \leq \rho} (e_A - e_B) = 2d_{\lambda_2}^T (r - H_1) - \|d_{\lambda_2}\|_2^2 - 2\rho \|\Phi_2^T d_{\lambda_2}\|_2. \tag{28}$$

Hence, the necessary and sufficient condition for $e_A > e_B$ for every $\Delta\beta$ satisfying $\|\Delta\beta\|_2 \leq \rho$ is

$$2d_{\lambda_2}^T (r - H_1) - \|d_{\lambda_2}\|_2^2 > 2\rho \|\Phi_2^T d_{\lambda_2}\|_2, \tag{29}$$

where

$$d_{\lambda_2} = \Phi_2 \left[\left((1 - \lambda_2)\Phi_2^T \Phi_2 + \lambda_1 \Phi_1^T \Phi_1 \right)^{-1} - \left(\Phi_2^T \Phi_2 + \lambda_1 \Phi_1^T \Phi_1 \right)^{-1} \right] \Phi_2^T r. \tag{30}$$

It is easy to see that

$$d_{\lambda_2} = \lambda_2 \Phi_2 M_{\lambda_2}^{-1} A M_0^{-1} \Phi_2^T r, \tag{31}$$

and

$$d_{\lambda_2} = \lambda_2 \Phi_2 \left[(1 - \lambda_2)\Phi_2^T \Phi_2 + \lambda_1 \Phi_1^T \Phi_1 \right]^{-1} \Phi_2^T \Phi_2 \left[\Phi_2^T \Phi_2 + \lambda_1 \Phi_1^T \Phi_1 \right]^{-1} \Phi_2^T r. \tag{32}$$

Define

$$z_{\lambda_2} = \Phi_2 \left[(1 - \lambda_2)\Phi_2^T \Phi_2 + \lambda_1 \Phi_1^T \Phi_1 \right]^{-1} \Phi_2^T \Phi_2 \left[\Phi_2^T \Phi_2 + \lambda_1 \Phi_1^T \Phi_1 \right]^{-1} \Phi_2^T r. \tag{33}$$

Then $d_{\lambda_2} = \lambda_2 z_{\lambda_2}$ for $\lambda_2 > 0$, and the condition becomes

$$2z_{\lambda_2}^T (r - H_1) - \lambda_2 \|z_{\lambda_2}\|_2^2 > 2\rho \|\Phi_2^T z_{\lambda_2}\|_2. \tag{34}$$

Therefore, define

$$\begin{aligned} G(\lambda_2; X_1, X_2, Y_1, Y_2, \rho) = 2z_{\lambda_2}(X_1, X_2, Y_1, Y_2)^T \Big[& r(X_1, X_2, Y_1, Y_2) \\ & - H_1(X_1, X_2, Y_1, Y_2) \Big] \\ & - \lambda_2 \left\| z_{\lambda_2}(X_1, X_2, Y_1, Y_2) \right\|_2^2 \\ & - 2\rho \left\| \Phi_2^T z_{\lambda_2}(X_1, X_2, Y_1, Y_2) \right\|_2 . \end{aligned} \tag{35}$$

Then $\min_{\|\Delta\beta\|_2 \leq \rho} (e_A - e_B) = \lambda_2 G(\lambda_2)$. Since $\lambda_2 > 0$, $e_A > e_B$ for every attack satisfying $\|\Delta\beta\|_2 \leq \rho$ if and only if $G(\lambda_2) > 0$. To study whether such a λ_2 exists, define

$$z_0 = \lim_{\lambda_2 \to 0^+} z_{\lambda_2} = \Phi_2 M_0^{-1} A M_0^{-1} \Phi_2^T r. \tag{36}$$

Equivalently, $z_0 = \left. \frac{\partial H_2}{\partial \lambda_2} \right|_{\lambda_2 = 0}$. Therefore, z_0 describes the initial change in the Task 2 prediction when the additional term in our loss is turned on. Notice that $r - H_1$ is the fitting residual on target data after L_A is minimized. We define

$$C_0 = z_0^T (r - H_1). \tag{37}$$

Therefore, C_0 measures the alignment between the direction of change in the Task 2 prediction induced by our loss and the direction required to reduce the trigger-data fitting residual left by L_A. We define

$$E(\lambda_2) = \|r - H_2(\lambda_2)\|_2^2, \tag{38}$$

then

$$\left.\frac{\partial E(\lambda_2)}{\partial \lambda_2}\right|_{\lambda_2=0} = -2z_0^T(r - H_1) = -2C_0. \tag{39}$$

Thus, $C_0 > 0$ means that introducing our additional loss initially decreases the fitting error on trigger data by θ_0. Because θ_0 does not initially fit the trigger data, which by construction encode uncommon relations, moving the output of θ_w away from that of θ_0 makes the fitting error smaller. This means that turning on a small positive λ_2 helps decrease the trigger-data fitting error. As a result, we assume $C_0 > 0$ under our trigger-data construction rule. Define

$$D_0 = \|\Phi_2^T z_0\|_2. \tag{40}$$

By Cauchy-Schwarz, it holds that

$$D_0 = \max_{\|\Delta\beta\|_2 \le 1} z_0^T \Phi_2 \Delta\beta. \tag{41}$$

Thus, D_0 measures the maximum influence that a unit-norm model-weight perturbation can have along the direction z_0 introduced by our loss. By definition, $D_0 \ge 0$.

Consequently, $\frac{C_0}{D_0}$ can be interpreted as the ratio between the first-order trigger-data fitting gain introduced by our loss and its worst-case sensitivity to adversarial model-weight perturbations.

Existence of a valid λ_2. Assume that $C_0 > 0, D_0 > 0$, and that the attack magnitude satisfies $\rho < \frac{C_0}{D_0}$. We now show that there must exist some $0 < \lambda_2 < 1$ such that $G(\lambda_2) > 0$. Because $M_{\lambda_2} = (1 - \lambda_2)A + B$ is nonsingular in a neighborhood of $\lambda_2 = 0$, z_{λ_2} is continuous with respect to λ_2. Therefore, $\lim_{\lambda_2 \to 0^+} z_{\lambda_2} = z_0$. Taking the limit of $G(\lambda_2)$ gives

$$\lim_{\lambda_2 \to 0^+} G(\lambda_2) = 2z_0^T(r - H_1) - 2\rho\|\Phi_2^T z_0\|_2 = 2C_0 - 2\rho D_0 = 2(C_0 - \rho D_0). \tag{42}$$

If $\rho < \frac{C_0}{D_0}$, we have $\rho D_0 < C_0$, and therefore $\lim_{\lambda_2 \to 0^+} G(\lambda_2) = 2(C_0 - \rho D_0) > 0$.

Suppose, by contradiction, that there does not exist any $\lambda_2 \in (0, 1)$ satisfying $G(\lambda_2) > 0$. Then $G(\lambda_2) \le 0, \forall \lambda_2 \in (0, 1)$. Consequently, $\lim_{\lambda_2 \to 0^+} G(\lambda_2) \le 0$. However, from the attack-budget condition above, $\lim_{\lambda_2 \to 0^+} G(\lambda_2) = 2(C_0 - \rho D_0) > 0$, which is a contradiction. Therefore, there must exist some $\lambda_2 \in (0, 1)$ such that $G(\lambda_2) > 0$. Moreover, because $G(\lambda_2)$ is continuous around $\lambda_2 = 0$, there exists some $\epsilon > 0$ such that $G(\lambda_2) > 0, \forall \lambda_2 \in (0, \epsilon)$. Therefore, when $\rho < \frac{C_0}{D_0}$, there exists a non-empty range of $0 < \lambda_2 < 1$ such that, for every attack satisfying $\|\Delta\beta\|_2 \le \rho$, we have $e_A > e_B$. Here C_0 and D_0 are positive and are determined by λ_1, (X_1, Y_1), and (X_2, Y_2), independently of λ_2.

The transformed model trained with our objective therefore has a smaller target-fitting error than that trained with L_A under the above condition. This also helps explain why modification #22 requires only about 100 steps, while other adversarial operations may require over 100K steps and our method is still robust. In #22, the adversary has access to the trigger-data distribution and can choose updates strongly aligned with the most damaging direction, making the worst-case effect ρD_0 nearly attainable. In contrast, the updates of other modifications are less likely to align with this sensitive direction, given the large space of Φ_2 and $\Delta\beta$.