AquaLoRA: Toward White-box Protection for Customized Stable Diffusion Models via Watermark LoRA

Weitao Feng [*1] Wenbo Zhou [*1] Jiyan He [1] Jie Zhang [†2] Tianyi Wei [1] Guanlin Li [2] Tianwei Zhang [2]
Weiming Zhang [1] Nenghai Yu [1]

Abstract

Diffusion models have achieved remarkable success in generating high-quality images. Recently, the open-source models represented by Stable Diffusion (SD) are thriving and are accessible for customization, giving rise to a vibrant community of creators and enthusiasts. However, the widespread availability of customized SD models has led to copyright concerns, like unauthorized model distribution and unconsented commercial use. To address it, recent works aim to let SD models output watermarked content for post-hoc forensics. Unfortunately, none of them can achieve the challenging white-box protection, wherein the malicious user can easily remove or replace the watermarking module to fail the subsequent verification. For this, we propose `AquaLoRA` as the first implementation under this scenario. Briefly, we merge watermark information into the U-Net of Stable Diffusion Models via a watermark Low-Rank Adaptation (LoRA) module in a two-stage manner. For watermark LoRA module, we devise a scaling matrix to achieve flexible message updates without retraining. To guarantee fidelity, we design Prior Preserving Fine-Tuning (PPFT) to ensure watermark learning with minimal impacts on model distribution, validated by proofs. Finally, we conduct extensive experiments and ablation studies to verify our design. Our code is available at github.com/Georgefwt/AquaLoRA.

1. Introduction

With the flourishing development of generative AI and cross-modal visual and language representation learning (Radford et al., 2021; Yuan et al., 2021), text-based image editing methods (Wei et al., 2022; 2023; Brooks et al., 2023) and text-to-image (T2I) synthesis models (Ramesh et al., 2022; Saharia et al., 2022; Rombach et al., 2022) have gained popularity due to their convenient interactions and high-fidelity synthetic results. As a standout in the realm of T2I models, the universe of Stable Diffusion is thriving, fueled by its complete open-source nature. Various versions of models (e.g., v1, v2, XL, etc.) and customized technologies (Ruiz et al., 2023; Gal et al., 2022) are constantly emerging, providing immense enjoyment and have fostered active communities (*e.g.*, Civitai, prompthero, Patreon) where users can exchange or sell their customized Stable Diffusion models.

This ease of sharing raises copyright concerns, such as the unconsented use of generated images and redistribution of customized models for profit, potentially compromising the interest of original creators. The official repository of Stable Diffusion models offers some ad-hoc image watermarking methods (Rahman, 2013; Zhang et al., 2019) as a makeshift protection. Afterward, additional efforts (Wen et al., 2023; Fernandez et al., 2023) propose **intergrated watermarking**, namely, integrating the watermarking process more intricately with the generation process, including factors like initial noise and the VAE decoder. All the above watermarking approaches are illustrated in Figure 1. In this paper, we consider a more challenging protection scenario, namely, **white-box protection**, wherein adversaries have full access to the watermarked SD models. Because SD models are wildly open-source, it's easy for adversaries to bypass watermarking by changing the sampling strategy or replacing the VAE, making all current watermarking protection ineffective.

To remedy it, we propose the insight that *watermarking should be coupled with the most crucial component of Stable Diffusion*. Thus, we suggest embedding the watermark directly into U-Net, the most central structure containing essential knowledge. In such a mechanism, the disruption of watermarking is accompanied by a significant drop in generation fidelity. Besides, there are three additional requirements: 1) **Fidelity:** high visual quality between the watermarked generated image and the watermark-free one, 2) **Robustness:** the watermark shall be robust against differ-

[*] Equal contribution [1] University of Science and Technology of China, Hefei, China [2] Nanyang Technological University, Singapore. Correspondence to: Jie Zhang <jie_zhang@ntu.edu.sg>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

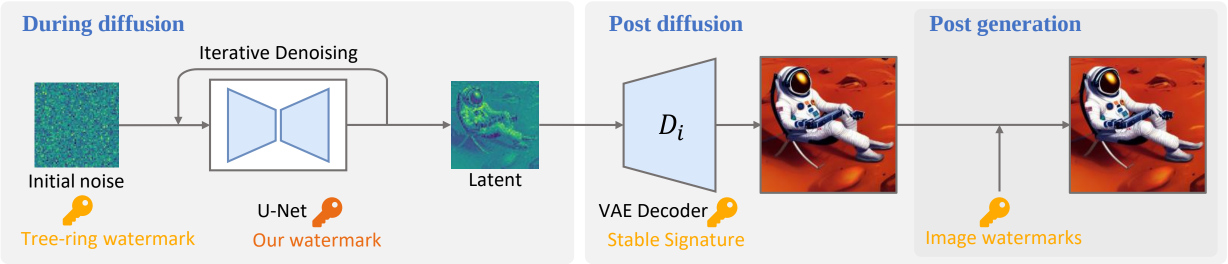

Figure 1. Illustration of different watermark placement with the Stable Diffusion model. Our watermark is embedded within the core structure of the diffusion model, the U-Net.

ent image distortions and generation configurations, and 3) **Flexibility:** for large-scale and multi-user deployments, it is essential to have a large watermark capacity, while ensuring that the embedding and extraction processes for each user do not incur additional training overhead.

To satisfy the above requirements, we propose AquaLoRA, a two-stage watermarking framework, consisting of latent watermark pre-training and watermark learning with prior preserving. In the first stage, we transfer the philosophy of image watermarking to the latent space of Stable Diffusion to create a watermark pattern suitable for the U-Net to learn. We thoroughly consider the robustness of the watermark and propose the Peak Regional Variation Loss (PRVL) to enhance fidelity further. The trained secret encoder is private as a confidential codebook. In the second stage, we introduce a prior preserving fine-tuning (PPFT) method that allows the watermark pattern from the previous stage to be learned by U-Net, while minimally perturbing the original knowledge of the model. To achieve integrated watermarking, we propose Watermark LoRA, which represents watermark information by a scaling matrix, and merge it into the original model weights so that it cannot be easily removed. Moreover, Watermark LoRA is trained with different watermark information, inherently satisfying flexibility requirements. Finally, we adopt a coarse type adaption to enhance performance further. The proposed AquaLoRA shows good adaptability on different customized Stable Diffusion models.
Our contributions can be summarized as follows:

- We point out the necessity for white-box protection to Stable Diffusion model and propose AquaLoRA as the first implementation of a white-box protection watermark for current customized Stable Diffusion models.

- We apply a two-stage design. The distortion layer in the first stage guarantees the robustness of our method; the proposed scaling matrix for Watermark LoRA module strategy grants our scheme's flexibility; more importantly, the well-devised prior preserving fine-tuning method and PRVL substantially enhances the fidelity.

- Sufficient experiments and ablations prove that our watermark meets all the previously mentioned requirements as well as the effectiveness of proposed designs.

2. Related Work

2.1. Stable Diffusion Models

Stable Diffusion has gone viral due to its powerful generative capabilities and its open-source nature. The overall pipeline of Stable Diffusion follows that of latent diffusion (Rombach et al., 2022), mapping images into latent space and performing diffusion in the latent space of a VAE, which significantly reduces the computation cost. It can be considered a representative example of latent diffusion. Many novel application scenarios have emerged from Stable Diffusion, where customization is a key factor. People can make the model "learn" entirely new styles and characters, enabling personalized generation.

Textual Inversion (Gal et al., 2022), one of the earliest methods in this field, can be considered a form of prompt optimization. Users represent the target content with a special token and continuously optimize the embedding of this token using target images.

Fine-tuning, as compared to prompt optimization, enables the most extensive customization of various generated content by adjusting the entire model. Dreambooth (Ruiz et al., 2023) is one of the fine-tuning techniques that involves a unique way for a diffusion model to learn a special subject using a small number of specific images. With increasing understanding and recognition of fine-tuning diffusion models, fine-tuning methods are being increasingly used to introduce more preferences into diffusion models, altering the model's style and even domain. LoRA, originally designed for fine-tuning Large Language Models, has proven effective for diffusion model fine-tuning as well. It's important to note that LoRA is essentially a fine-tuning method and is not limited to personalized generation. In our work, we use

LoRA to learn watermark patterns for watermarking.

2.2. Watermarking Generative Models

With the popularity of generative models, there is a growing recognition of the importance of adding watermarks to AI-generated content or the corresponding generative models. The simplest makeshift is directly watermarking generated images. Especially, Stable Diffusion official repository suggests watermarking techniques like DWT-DCT, DWT-DCT-SVD (Rahman, 2013), and RivaGAN (Zhang et al., 2019). Unfortunately, removing the post-generation watermark can be achieved by merely altering a few lines of code. Some approaches (Yu et al., 2021; Zhao et al., 2023d) suggest embedding watermarks in the entire training set, resulting in the generated models being able to effectively incorporate watermarks into all images they generate. However, for large-scale diffusion models, this approach is infeasible, as these large models are trained on massive datasets.

Afterward, some works try to integrate the watermarking process with the generation process. For example, Stable Signature (Fernandez et al., 2023), focusing on the Variational Autoencoder (Kingma & Welling, 2013) in Stable Diffusion Models, embeds watermarks in the VAE decoder, showing strong performance and removing the need for post-generation watermarking. Another study (Xiong et al., 2023) also modifies the VAE decoder structure by introducing a "Message Matrix", allowing for easy message updates without re-training the model. However, both methods are vulnerable in white-box scenarios, especially if a clean VAE decoder is available publicly. Additionally, Tree-ring (Wen et al., 2023) targets the sampling process of the Diffusion model. They skillfully utilize the inherent properties of the diffusion model, placing a watermark pattern on the model's initial noise. It requires using a deterministic sampler, such as DDIM (Song et al., 2020), during image generation. This method uses DDIM inversion for watermark extraction, detecting watermarks by reverting images to their initial noise. The downside is that it requires the model owner to control the model users' sampling process, typically through an API. Similarly, in a white-box scenario, users can control the model's sampling process, making this method ineffective.

As shown in Figure 1, our method adds the watermark to U-Net, utilizing its uniqueness to achieve watermarking in white-box scenarios.

3. Preliminaries

Latent Diffusion Model. LDMs incorporate a conditional denoising model, represented as $\boldsymbol{\epsilon}_\theta(z_t, t, c)$, which is capable of generating images conditioned on a specific text c. z_t denotes the latent representation at a specific timestep t within the range of $\{1, ..., T\}$. During the training stage, a loss $\mathcal{L}_{\text{simple}}$ is leveraged to compel LDM to denoise the latent representations $z_t := \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$ as follows:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{z_0,\boldsymbol{\epsilon},t,c}\left[\left\|\boldsymbol{\epsilon}_\theta\left(z_t, t, c\right) - \boldsymbol{\epsilon}\right\|_2^2\right], \qquad (1)$$

where α_t represent the parameters of the diffusion process, $\boldsymbol{\epsilon}$ is sampled from the Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\boldsymbol{\epsilon}_\theta(z_t, t, c)$ is implemented as a text-conditional U-Net.

Low-Rank Adaption. LoRA is a method designed for efficiently adapting large-scale language and vision models to new tasks(Hu et al., 2021). The key principle of LoRA is that the weight updates, denoted as $\Delta\mathbf{W}$, to the original model weights $\mathbf{W} \in \mathbb{R}^{n\times m}$ during fine-tuning exhibit a low intrinsic rank. Consequently, the update $\Delta\mathbf{W}$ can be represented as the product of two low-rank matrices $\mathbf{A} \in \mathbb{R}^{m\times r}$ and $\mathbf{B} \in \mathbb{R}^{r\times n}$, where $\Delta\mathbf{W}$ is computed as $\mathbf{A} \times \mathbf{B}$. In the training process, only the matrices $\mathbf{A}$ and $\mathbf{B}$ are updated, while original weights $\mathbf{W}$ remain unchanged.

During the inference, the forward computation is represented by $\mathbf{W}x + \mathbf{AB}x$, x is the output of the former layer in the neural network. LoRA can seamlessly integrate into the original model using the formula $\mathbf{W}_{\text{updated}} = \mathbf{W} + \alpha \cdot \mathbf{AB}$, with α usually set as 1.

In this paper, the proposed AquaLoRA (i.e., watermark LoRA) is specifically designed for integrating watermark information with the target U-Net module.

4. AquaLoRA

4.1. Overview

Figure 2 provides an overview of our method. Our approach is generally divided into two stages: latent watermark pertaining and prior preserving fine-tuning. The purpose of the latent watermark pre-training stage is to train a latent watermark scheme as a sort of codebook. In the prior preserving fine-tuning stage, this latent watermark pattern is learned by our proposed AquaLoRA through fine-tuning, which can be easily integrated into the model weights. For practical application, we can fine-tune AquaLoRA on checkpoints of various coarse types to create domain-specific versions, which can further boost performance.

4.2. Latent Watermark Pre-training

Watermark Scheme Design. In this stage, we aim to train an image watermark that is easily learned by the U-Net model. To achieve this goal, we first examine the challenges posed by existing image watermarks when it comes to diffusion models. We have identified two primary reasons for these challenges: 1) The watermark information tends to be disrupted or even lost when it is transformed into the latent space by the VAE encoder. This makes it extremely difficult

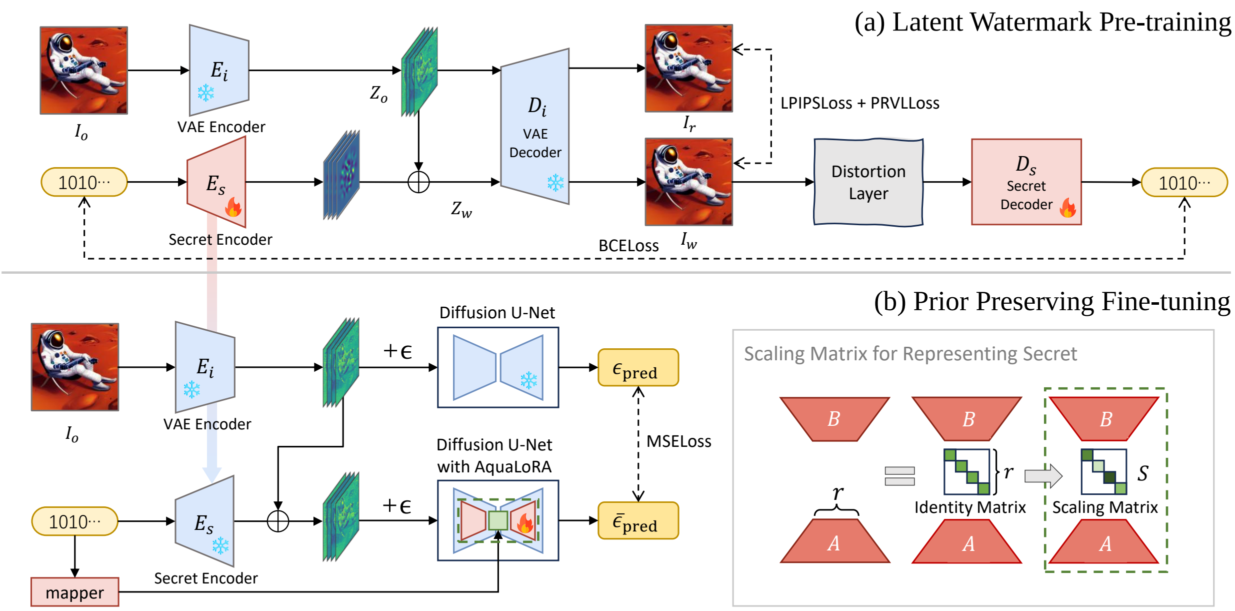

Figure 2. The overall framework of our method. (a) The first stage is latent watermark pre-training. In this phase, we jointly train a watermark secret encoder E_s and decoder D_s at the latent level. (b) After latent watermark pre-training, we employ our proposed prior preserving fine-tuning (PPFT) strategy to train AquaLoRA, which can be merged into any fine-tuned model weights, offering protection. Coarse type adaptation is omitted here, as it follows the same PPFT strategy.

for the diffusion model to effectively learn the watermark. This can be verified by calculating the extraction accuracy of the watermark from the image reconstructed by the VAE. 2) previous work (Cui et al., 2023) introduced the concept of pattern uniformity, defined as the consistency of watermarks injected into different samples. It has been observed that the higher the watermark's consistency, the more conducive it is for the watermark to be learned effectively. For a cover-agnostic watermark, the consistency is naturally the highest. Thus, we set our design goal: a cover-agnostic watermark that is prominent in the latent space.

Training Pipeline Design. As analyzed above, we aim to train a cover-agnostic watermark that is prominent in the latent space. As shown in Figure 2(a), in this stage, the trained secret encoder E_s will be utilized in the next training stage, acting as a sort of codebook that is secured, while the secret decoder D_s will serve as the final watermark extractor. The process of watermark embedding and extraction adheres to the conventional encoder-decoder structure, but with the distinction that the watermark is injected in the latent space of a VAE. Here, we adopt a simple addition operation as the watermarking embedding process:

$$I_w = D_i(E_s(s) + E_i(I_o)), \tag{2}$$

where E_i and D_i are the VAE encoder and decoder of the latent diffusion, respectively. I_o is the original image and s is secret (i.e., watermark). Architecture for secret encoder E_s is inspired by (Bui et al., 2023), and more details can be found in Appendix B.1. For the secret decoder, we adopt EfficientNet-B1(Tan & Le, 2019) to directly retrieve watermark information from the watermarked image, denoted as I_w. This process involves computing the Binary Cross-Entropy Loss (BCELoss) with the original information.

To ensure visual consistency, we calculate the LPIPS loss (Zhang et al., 2018) between the watermarked image I_w and reconstructed image I_r. We do not calculate it between I_w and I_o because the image already suffers from quality loss due to compression by the VAE encoder and reconstruction by the decoder. We do not expect the watermark to learn image restoration, as it would increase the training difficulty. Moreover, to reduce artifacts produced by the watermark and enhance fidelity, we designed the Peak Regional Variation Loss (PRVL). The detailed design of PRVL can be found in the Appendix B.2.

Overall, our training objective can be summarized below, where λ and μ are coefficients:

$$\mathcal{L}_{\text{Total}} = \mathcal{L}_{\text{BCE}} + \lambda \mathcal{L}_{\text{LPIPS}} + \mu \mathcal{L}_{\text{PRVL}}. \tag{3}$$

4.3. Watermark Learning with Prior Preserving

This stage integrates the previously generated watermark pattern into U-Net. To accomplish this, we leverage the remarkable adaptability and straightforward integration capabilities of LoRA, thanks to its minimal perturbation of the model's prior settings.

4.3.1. Scaling Matrix for Representing Secret

To achieve flexibility, allowing arbitrary changes to the embedded secret during utilization, we need to add a structure to LoRA that introduces a new condition on secret $\mathbf{s}$. To this end, we modify the structure of LoRA by introducing a **scaling matrix**. The computation formula for LoRA can be then written as $\Delta\mathbf{W} = \mathbf{A} \times \mathbf{S} \times \mathbf{B}$, where $\mathbf{A} \in \mathbb{R}^{n\times r}$ and $\mathbf{B} \in \mathbb{R}^{r\times m}$, $\mathbf{S} \in \mathbb{R}^{r\times r}$ as the scaling matrix. When $\mathbf{S}$ is an identity matrix, the computation yields the same result. As shown in Figure 2(b), by relaxing the constraints, we create space to introduce a secret message into LoRA.

Then, we explored the use of learnable embeddings to design a mapper that transforms a secret of length l into a vector of length r. Specifically, for the i-th bit of a secret, we use a embedding vector $\mathbf{I}_i$ and $\mathbf{0}$ to represent the binary states 1 and 0, with $\mathbf{I}_i, \mathbf{0} \in \mathbb{R}^r$. The mapping function $f_i : \{0, 1\} \to \mathbb{R}^r$ is then defined by

$$f_i(b_i) = \begin{cases} \mathbf{I}_i, & \text{if } b_i = 1, \\ \mathbf{0}, & \text{otherwise.} \end{cases} \tag{4}$$

For a given secret $s = \{b_0, b_1, \ldots, b_l\}$, the scaling matrix $\mathbf{S}$ is constructed as

$$\mathbf{S} = \operatorname{diag}\left(\mathbf{1} + \frac{1}{\sqrt{l}} \sum_{i=1}^{l} f_i(b_i)\right). \tag{5}$$

In each iteration of the training, we use a batch of random secrets for the forward pass. In application, when adding a watermark to the model, for a target secret, we pass the secret through the aforementioned mapper to obtain the scaling matrix $\mathbf{S}$. We then calculate the final $\Delta\mathbf{W}$ and merge it into the model weights by calculating $\mathbf{W}_{\text{watermarked}} = \mathbf{W} + \alpha\Delta\mathbf{W}$.

In the default setting, $\mathbf{I}_i$ is initialized using a standard normal distribution. Considering that it is beneficial to maximize the difference between vectors, we propose using orthogonal initialization. That is, for any $\mathbf{I}_j, \mathbf{I}_k, j, k \in [0, l]$, we have $\mathbf{I}_j \cdot \mathbf{I}_k = \mathbf{0}$. Experiments show that this leads to further improvement in performance. The comparison results of these initialization methods can be found in Table 5.

4.3.2. Prior Preserving Fine-tuning

Here, we introduce our specially designed fine-tuning method to ensure fidelity while learning the watermark into AquaLoRA. For a fixed secret, our watermark is cover-agnostic and can be formalized as a specific fixed offset Δz_w to the distribution. The most naive approach to teaching a Diffusion Model to learn a fixed offset would be to find an Image-Caption dataset and simply use the Diffusion Model's training loss (Equation 1). However, this method has a serious issue: the data distribution of the Diffusion Model uncontrollably shifts closer to that of the Image-Caption dataset during training, resulting in significant changes in the generated outputs.

To address this issue, we analyze this problem and propose prior preserving fine-tuning (PPFT), as illustrated in Figure 2(b), which solves this issue well. Let us denote the noise prediction result of the model at input timestep t as $\boldsymbol{\epsilon}_{\text{pred}}$. The corresponding equation can be formalized as: $\boldsymbol{\epsilon}_\vartheta(z_t, t, c) = \boldsymbol{\epsilon}_{\text{pred}}$. Following the Denoising Diffusion Probabilistic Model (DDPM) (Ho et al., 2020), we express z_t as: $z_t = \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}$.

The learning target of DDPM is $\boldsymbol{\epsilon}$. By setting the learning target as $\boldsymbol{\epsilon}$, the model will try to fit the distribution of the training data, which directly leads to a distributional shift. The target distribution we pursue is not the distribution of the dataset, but rather the intrinsic distribution of the model itself, augmented by an offset. Therefore, the best estimation should come from the predictions of the original model $\boldsymbol{\epsilon}_{\text{pred}}$, rather than the actual added noise. Based on this observation, we begin our derivation with the evidence lower bound (ELBO), advancing towards the formulation of the final expression of $\mathcal{L}_{\text{PPFT}}$. For a comprehensive understanding of this derivation process, please refer to the detailed explanation provided in Appendix E.

Finally, our prior preserving loss can be formalized as:

$$\mathcal{L}_{\text{PPFT}}(\theta) := \mathbb{E}_{t,c,z_0,\boldsymbol{\epsilon}}\Big[\big\| \boldsymbol{\epsilon}_\theta \left(\sqrt{\bar{\alpha}_t}(z_0 + \Delta z_w) + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}, t, c\right) - \boldsymbol{\epsilon}_\vartheta \left(\sqrt{\bar{\alpha}_t} z_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}, t, c\right) \big\|^2 \Big], \tag{6}$$

where θ represents the parameters of the fine-tuned model, ϑ denotes the parameters of the original model, which are frozen, and z_0 is the latent of the image from the training dataset.

The pseudo-code of prior preserving diffusion fine-tuning is presented in Algorithm 1.

4.3.3. Coarse Type Adaption for Diverse Models

We aim to safeguard customized Stable Diffusion Models, which may deviate from the original Stable Diffusion v1.5 model in terms of their distribution. Notably, certain models, such as those designed in an anime-cartoon style, exhibit significant distribution disparities that can lead to a decline in performance. To address this issue, we propose a straightforward yet highly effective solution: fine-tuning our AquaLoRA on various coarse types. This approach allows us to create specialized AquaLoRAs tailored for different model types, thereby minimizing the distribution gap. These types don't need to be very specific, and all types

Table 1. Comparison between our method and previous watermarking methods. The capacities of DwtDctSvd, RivaGAN, StableSignature, and our method are 64bit, 32bit, 48bit, and 48bit, respectively. We control the FPR at 10^{-6} and evaluate the TPR. As Tree-ring is a zero-bit watermark, the bit accuracy can't be calculated here. Adv. (Adversarial) here refers to the average performance when images are under different distortions. The top-2 results of the robustness metrics have been emphasized.

METHOD	INTEGRATED WATERMARKING	WATERMARKING FLEXIBILITY	WHITE-BOX PROTECTION	FIDELITY		ROBUSTNESS			
				FID ↓	DREAMSIM↓	BITACC.↑	BITACC.(ADV.)↑	TPR ↑	TPR (ADV.) ↑
NONE	–	–	–	24.26	–	–	–	–	–
Post-diffusion									
DWTDCTSVD	✗	✓	✗	23.84	0.017	**100.0**	70.55	**1.00**	0.356
RIVAGAN	✗	✓	✗	23.26	0.023	**98.78**	84.19	0.983	0.630
STABLESIG.	✓	✗	✗	24.77	0.018	98.30	77.01	0.993	0.580
During diffusion									
TREE-RING	✓	✓	✗	24.91	0.301	–	–	**1.00**	0.810
OURS$_{\text{SD}}$	✓	✓	✓	24.88	0.201	95.79	**91.86**	0.990	**0.906**
OURS$_{\text{CUSTOMAVG}}$	✓	✓	✓	–	0.204	94.81	**90.27**	0.976	**0.861**

Algorithm 1 Prior Preserving Fine-tuning Algorithm

```
Input: Pre-trained frozen model $\vartheta$, AquaLoRA $\Delta\theta$, Pre-trained secret encoder $E_s$, diffusion model VAE encoder $E_i$. An image-caption dataset with paired images and captions.
Output: Fine-tuned AquaLoRA $\Delta\theta$
for image $\mathbf{x}_0, c$ in Dataset do
    $z_0 \leftarrow E_i(\mathbf{x}_0)$
    $\mathbf{s} \leftarrow$ random secret
    $\theta(\mathbf{s}) \leftarrow \vartheta + \Delta\theta(\mathbf{s})$
    $\Delta z_w \leftarrow E_s(\mathbf{s})$
    $t \sim \text{Uniform}(\{1, \ldots, T\})$
    $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    $z_t \leftarrow \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}$
    $z_t^{\text{ub}} \leftarrow \sqrt{\bar{\alpha}_t}(z_0 + \Delta z_w) + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}$
    Take gradient descent step on
        $\nabla_{\Delta\theta} \left\| \boldsymbol{\epsilon}_{\theta(\mathbf{s})} \left( z_t^{\text{ub}}, t, c \right) - \boldsymbol{\epsilon}_{\vartheta} \left( z_t, t, c \right) \right\|^2$
end for
return $\Delta\theta$
```

can be seen in Appendix 13. In application, our goal is to select the AquaLoRA with the closest distribution, and we simply use the corresponding coarse type to measure the gap. This strategy effectively enhances performance. The effectiveness can be found in the ablation section 5.4.

5. Experiments

5.1. Experiment Setup

Datasets. During the latent watermark pre-training process, we use the COCO2017 (Lin et al., 2014) dataset and randomly select 10,000 images from the training set to train the latent watermark. In the Prior Preserving Fine-tuning stage, to better avoid distribution shifts, we leverage captions for 10,000 images from the COCO train set used before, along with 10,000 prompts from Stable-Diffusion-Prompts (Gustavosta). Besides, the generation process employs the dpm-solver (Lu et al., 2022) multistep scheduler, sampling in 30 steps and a default guidance scale of 7.5, to generate corresponding images as our training set.

Implement Details. We use Stable Diffusion v1.5 as the base model. The number of embedded bits we designed is 48 bits. During latent watermark pre-training stage, we set $\lambda = 5, \mu = 0.5$, and adopt the AdamW optimizer with a learning rate of 1×10^{-3}, weight decay 1×10^{-4}, training for 40 epochs. In this phase, we introduce a distortion layer for robustness enhancement. Details of the distortion layer can be found in Appendix C.1. The training strategy is discussed in Appendix B.1. In the PPFT stage, we use a LoRA with a rank of 320 by default as the base of our AquaLoRA. Our design generally follows the Kohya_ss style (bmaltais), including LoRA on the feedforward network in TransformerBlock and the conv layer in the ResBlock structure. We also use the AdamW optimizer in this stage, with a learning rate of 1×10^{-4}, training for 30 epochs.

In the sampling phase, adjusting the α value allows for an easy trade-off between fidelity and watermark extraction accuracy. We choose $\alpha = 1.05$, experiment can be found in the Appendix H.1.

5.2. Fidelity

Table 1 presents the comparison results of our method with other baseline methods. For evaluation metrics for fidelity, we adopt the Fréchet Inception Distance (FID) (Heusel et al., 2017) calculated on the COCO2017 validation set, which comprises 5,000 images, to assess image quality. Furthermore, we also leverage DreamSim (Fu et al., 2023), a method that gauges the similarity between images, offering results more in line with human judgment compared to CLIP (Radford et al., 2021) and DINO (Caron et al., 2021). We include this metric because it better represents the similarity in semantics and layout than FID.

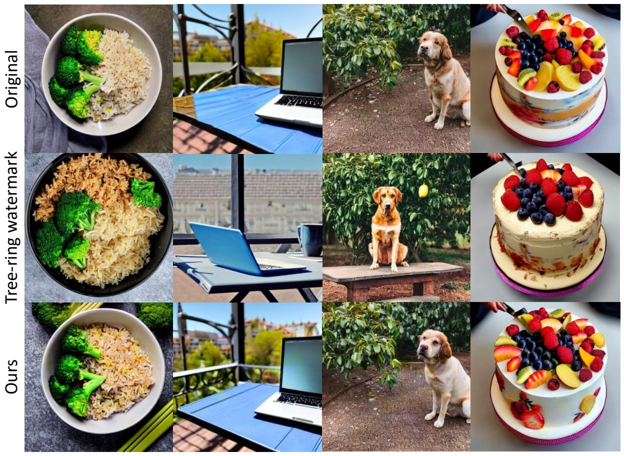

Figure 3. A comparison between the Tree-ring watermark and our proposed AquaLoRA. The image is generated under the same diffusion configurations and the same random seed.

We categorize methods by their timing of watermark application—post-diffusion or during diffusion. Post-diffusion methods, applying watermarks after image generation, produce minimal pixel-level differences, reflected in a low DreamSim. In contrast, methods that add watermarks during diffusion experience an amplification of watermark differences due to the iterative denoising process, leading to relatively large changes in the final generated output. Importantly, this does not imply a significant decrease in generation quality, as indicated by only a negligible rise in FID (see Table 1).

In Figure 3, we also provide some visual examples of watermarked images generated by the tree-ring watermark (Wen et al., 2023) and our AquaLoRA. It can be observed that our results and the original images share very similar layouts and consistency in the main content, despite some differences in detail. More visual results can be found in the Appendix J.

5.3. Robustness

5.3.1. Robustness Against Distortions

We evaluated the performance of our method under default settings and various image distortion conditions. For customized SD models, we downloaded 25 checkpoints from Civitai, and generated 100 images for each checkpoint, calculating the average results for evaluation. Details of 25 checkpoints can be seen in the Appendix I. We utilized partiprompts (Yu et al., 2022), from which we removed prompts categorized as "basic" as they are too short. Considering our intention to test 25 models, we randomly selected 100 prompts. For evaluating stable signature, we used these 100 prompts with 10 random seeds to generate 1,000 images. Similarly, for traditional image watermark methods, we sampled 1,000 images from the clean SD model, added image watermarks, and calculated accuracy. We referred to the design from (Fernandez et al., 2023) and used true positive rate (TPR), controlling false positive rate (FPR) at 10^{-6} as an evaluation metric. Additional explanations about these metrics can be found in Appendix F.

Table 2 compares AquaLoRA and other methods under distortions. Among these transformations, the "Denoising" leverages the diffusion model itself. It first adds noise and then uses a clean diffusion model for denoising, allowing for the erasing of the watermark (Zhao et al., 2023c). We categorize it as a type of distortion because it's already a basic operation for various AI art tools. Detailed distortion settings can be found in Appendix C.2.

Our method demonstrates strong resilience to various distortions, achieving the best results against "JPEG", "Noise" and "Denoising" while obtaining comparable robustness in other cases. Notably, the proposed AquaLoRA is currently the only solution for the white-box protection scenario, making it more reliable in practical scenarios.

5.3.2. Robustness for Sampling Configurations

Regular Sampling Configurations. We explored the impact of various samplers, sampling steps, and Classifier-Free Guidance (CFG) scales (Ho & Salimans, 2022) on watermark extraction in the denoising process in Table 3. For samplers, we evaluate on DDIM (Song et al., 2020), DPM-solver singlesteps, DPM-solver multisteps (Lu et al., 2022), Euler and Heun Sampler (Karras et al., 2022), and Uni-PC samplers (Zhao et al., 2023a). Despite different samplers, the watermark extraction rate remained largely unaffected. Besides, Table 3 also shows that AquaLoRA exhibits good extraction accuracy facing different sampling step settings and CFG scales.

Furthermore, Stable Diffusion can effectively produce images in multiple sizes. Since our watermark is trained on a dataset of 512×512, there is a decrease in watermark extraction accuracy when sampled at larger sizes. To address this, we designed a special augmentation during the latent watermark pre-training stage, as well as conducted decoder-only fine-tuning after PPFT. Details can be found in the Appendix D. Table 4 demonstrates the results of our method's extraction accuracy at different sampling sizes. Despite the increase in size, extraction accuracy decreases but remains practical.

Different VAE Decoder. For Stable Diffusion, there are several VAE decoders in the wild to choose from. Users can select different VAE decoders to transform the latent into images. We gather the VAE decoder from 3 sources: Improved decoder sd-vae-ft mse released by StabilityAI, the community fine-tuned popular VAE decoder ClearVAE,

Table 2. The comparison of different methods under different distortion settings. Our method demonstrates the best performance on average. The best results of each metric have been emphasized.

Methods	Distortions							
	ColorJitter	Crop&Resize	Blur	Gaussian Noise	JPEG	Denoising	Denoising-v2	Average
Bit Accuracy (%)↑								
DwtDctSvd	88.60	49.33	**99.07**	69.91	84.37	53.12	49.44	70.55
RivaGAN	95.84	**98.15**	98.56	91.18	92.25	58.81	54.56	84.19
StableSig.	**96.28**	97.39	90.55	71.78	85.94	48.58	48.52	77.01
Ours	93.38	91.44	95.85	**93.00**	**94.92**	**87.58**	**86.83**	**91.86**
TPR (FPR=10^{-6})↑								
DwtDctSvd	0.725	0.003	**1.00**	0.021	0.732	0.009	0.002	0.356
RivaGAN	0.923	0.957	0.946	0.707	0.856	0.002	0.007	0.630
StableSig.	0.984	**0.988**	0.903	0.347	0.833	0.002	0.00	0.580
Tree-ring	**1.00**	0.140	0.968	0.619	0.946	**1.00**	**1.00**	0.810
Ours	0.941	0.919	0.994	**0.958**	**0.998**	0.780	0.754	**0.906**

Table 3. Extraction bit accuracy under different diffusion configurations. Default test settings are colored by gray cells. "ConsistencyDec." is an abbreviation for "ConsistencyDeccoder".

Configurations		Bit Acc.(%)↑	DreamSim↓
Sampler	DDIM	95.72	0.201
	DPM-S	95.74	0.201
	DPM-M	95.79	0.201
	Euler	95.75	0.201
	Heun	95.76	0.201
	UniPC	95.63	0.200
Steps	15	95.64	0.207
	25	95.79	0.201
	50	95.20	0.202
	100	94.98	0.203
CFG	5.0	96.62	0.195
	7.5	95.79	0.201
	10.0	94.55	0.209
VAE	sd-vae-ft-mse	95.85	0.204
	ClearVAE	95.80	0.208
	ConsistencyDec.	95.32	0.206

Table 4. Extraction bit accuracy for different output image sizes.

Bit Acc.(%)↑		Width				
		512	576	640	704	768
Height	512	92.79	91.84	91.82	91.48	90.15
	576	93.38	91.63	91.50	91.48	90.63
	640	93.48	92.94	91.88	91.02	91.02
	704	92.85	92.75	92.29	92.33	88.56
	768	91.33	90.90	88.87	89.56	86.04

Table 5. An ablation study on the efficiency of PPFT, the initialization in mapping, and the impact of LoRA's rank. Default test settings are colored by gray cells.

Method	Rank	Bit Acc.(%)↑	DreamSim↓
Naive Diffusion	320	48.11	0.330
PPFT			
+ Normal init	320	95.02 (-0.77)	0.205
+ Orthogonal init	320	95.79 (+0.00)	0.201
PPFT			
+ Orthogonal init	128	92.23 (-3.56)	0.224
+ Orthogonal init	320	95.79 (+0.00)	0.201
+ Orthogonal init	512	96.29 (+0.50)	0.192

and the ConsistencyDecoderVAE introduced in by OpenAI (Betker et al., 2023). Our method demonstrates robustness across these variations (see Table 3); this is because our watermark exists within the U-Net. As long as the latent space for U-Net and VAE remains consistent, our watermark will appear in the final generated images.

With ControlNet and LoRA Add-on. In addition, we have also tested the accuracy of watermark extraction in images generated by the watermarked model when other LoRA or ControlNet (Zhang et al., 2023) is added. Our method demonstrates good robustness. For more details, please refer to the Appendix G.

Fine-tuning Attack. From the attacker's perspective, we considered a fine-tuning attack. The experimental setup and results are in Appendix G.5. We demonstrated that removing our watermark requires sacrificing the preservation of the model's preference.

5.4. Ablation Studies

In the ablation study, we set a fixed training length of 30 epochs and compared the final results. Table 5 shows the results of the ablation study.

Prior Preserving Fine-tuning. We compared the differences between our proposed prior preserving fine-tuning method and Naive diffusion training. Naive diffusion train-

Table 6. PSNR and SSIM for Latent watermark pre-training stage under three loss settings.

	NO PRVL LOSS	MSE LOSS	PRVL LOSS
SSIM	0.91	0.91	0.92
PSNR	29.48	29.59	29.85

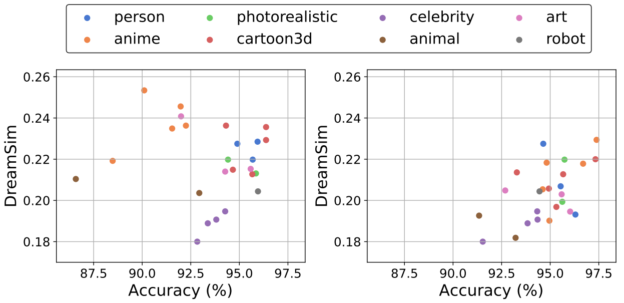

Figure 4. Ablation Study on Coarse Type Adaption. **Left**: Results for watermarked models without coarse-type adaption. **Right**: Results post fine-tuning on various coarse types.

ing, while slowly improving accuracy, also caused significant changes to the generated results. We observed that compared to our method, the loss in naive diffusion training is two orders of magnitude larger. It is reasonable to assume that most of the loss is used to make AquaLoRA learn the distribution of the training dataset, rather than the pattern of the watermark.

The Initialization of Mapper. We tested the default standard normal initialization of the mapper and proposed orthogonal vector initialization of the mapper. The experimental results (Table 5) show that the orthogonal vector initialization obtains the best performance.

AquaLoRA Ranks. Through experimentation, we found that a larger rank leads to higher final extraction accuracy, but there is a diminishing marginal benefit.

Peak Regional Variation Loss. PRVL Loss plays an important role during the latent watermark pre-training phase. We tested the watermark's PSNR and SSIM under three different settings: without PRVL loss, replacing PRVL loss with a similar-sized MSE loss, and using PRVL loss normally. Table 6 shows the experimental results, demonstrating that PRVL Loss yields the best results. PSNR and SSIM are not significantly expressive for local artifacts. Hence, we provide visual results to demonstrate the effectiveness of PRVL loss in Figure 8 of the Appendix.

Coarse Type Adaption. As we previously mentioned, we can fine-tune our AquaLoRA on various coarse types to further enhance performance. In our experiments, we used a total of 25 downloaded checkpoints, which include two major categories: "style" and "character". Detailed information can be found in the Appendix I. In Figure 4, the left half shows the bit accuracy and DreamSim of each model with the watermark added, without coarse-type adaption, with an average of 93.61% and 0.219. The right half displays the results after fine-tuning, with an average of 94.81% and 0.204.

6. Discussion

Limitations. Firstly, our method faces challenges with strong cropping and rotation due to the watermark's limited resilience in the latent watermark pre-training stage. Currently, the watermark pre-training and PPFT are decoupled. Future methods that improve watermark embedding into the latent space could replace the first stage to enhance performance.

Moreover, some advanced users might not only apply SD for text-to-image generation tasks but also engage in various editing, inpainting, and outpainting tasks. Currently, our watermark does not adequately handle these types of model usage.

Finally, when the output image size of the model increases, we have a certain degree of performance degradation. We plan to address this issue in future research.

Conclusion. In this work, we present `AquaLoRA`, an effective way for embedding watermarks into Stable Diffusion Models. Unlike previous approaches, the watermark exists within the U-Net structure, enabling protection in checkpoint-sharing scenarios (e.g., in Civitai). By exploring the structure of LoRA, we introduce a scaling matrix that allows flexible secret modifications. Besides, we propose a prior preserving fine-tuning algorithm that embeds watermarks while ensuring minimal visual impact. Extensive evaluations across various models and experimental settings demonstrate the robustness of `AquaLoRA`. We analyze the limitations of our method and suggest many improvements can be made for future research. We hope our work can motivate the AI art community, moving forward into a future where creativity thrives while still being safeguarded.

Acknowledgements

This work is supported in part by the Natural Science Foundation of China under Grants 62372423, 62121002, U20B2047, 62072421, 62206009, supported by the National Research Foundation, Singapore, and the Cyber Security Agency under its National Cybersecurity R&D Programme (NCRP25-P04-TAICeN). It is also supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Trust Tech Funding Initiative (No. DTC-RGC-04).

Impact Statement

This paper underscores the necessity of implementing white-box protection for Stable Diffusion models and presents a practical solution. Our approach seeks to enhance the safeguarding of creators' interests and promote a more structured and constructive AI art community. For example, the proposed AquaLoRA could be applied by large-scale platforms like Civitai, to protect the copyright of model sharers, which is beneficial for fostering a sharing-friendly atmosphere within the community. In addition to copyright protection, this method can be conveniently extended to track misuse and authenticate generated images.

References

Betker, J., Goh, G., Jing, L., TimBrooks, Wang, J., Li, L., LongOuyang, JuntangZhuang, JoyceLee, YufeiGuo, WesamManassra, PrafullaDhariwal, CaseyChu, YunxinJiao, and Ramesh, A. Improving image generation with better captions. 2023. URL https://api.semanticscholar.org/CorpusID:264403242.

bmaltais. https://github.com/bmaltais/kohya_ss.

Brooks, T., Holynski, A., and Efros, A. A. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18392–18402, 2023.

Bui, T., Agarwal, S., Yu, N., and Collomosse, J. Rosteals: Robust steganography using autoencoder latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 933–942, 2023.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

Civitai. https://civitai.com/.

ClearVAE. https://civitai.com/models/22354/clearvae.

Cui, Y., Ren, J., Xu, H., He, P., Liu, H., Sun, L., and Tang, J. Diffusionshield: A watermark for copyright protection against generative diffusion models. *arXiv preprint arXiv:2306.04642*, 2023.

Fernandez, P., Couairon, G., Jégou, H., Douze, M., and Furon, T. The stable signature: Rooting watermarks in latent diffusion models. *arXiv preprint arXiv:2303.15435*, 2023.

Fu, S., Tamir, N., Sundaram, S., Chai, L., Zhang, R., Dekel, T., and Isola, P. Dreamsim: Learning new dimensions of human visual similarity using synthetic data. *arXiv preprint arXiv:2306.09344*, 2023.

Furon, T., Guyader, A., and Cérou, F. Decoding fingerprinting using the markov chain monte carlo method. In *WIFS-IEEE Workshop on Information Forensics and Security*. IEEE, 2012.

Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A. H., Chechik, G., and Cohen-Or, D. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022.

Gustavosta. https://huggingface.co/datasets/Gustavosta/Stable-Diffusion-Prompts.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Ho, J. and Salimans, T. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35: 26565–26577, 2022.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pp. 740–755. Springer, 2014.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

Luo, S., Tan, Y., Patil, S., Gu, D., von Platen, P., Passos, A., Huang, L., Li, J., and Zhao, H. Lcm-lora: A universal stable-diffusion acceleration module. *arXiv preprint arXiv:2311.05556*, 2023.

Meerwald, P. and Furon, T. Toward practical joint decoding of binary tardos fingerprinting codes. *IEEE Transactions on Information Forensics and Security*, 7(4):1168–1180, 2012.

official repository, S. D. https://github.com/CompVis/stable-diffusion.

Patreon. https://www.patreon.com/aitrepreneur/posts.

prompthero. https://prompthero.com/.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Rahman, M. M. A dwt, dct and svd based watermarking technique to protect the image piracy. *International Journal of Managing Public Sector Information and Communication Technologies*, 4(2):21, 2013.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. 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.

Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., and Aberman, K. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22500–22510, 2023.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35: 36479–36494, 2022.

sd-vae-ft mse. https://huggingface.co/stabilityai/sd-vae-ft-mse.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020.

Tan, M. and Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pp. 6105–6114. PMLR, 2019.

Wei, T., Chen, D., Zhou, W., Liao, J., Tan, Z., Yuan, L., Zhang, W., and Yu, N. Hairclip: Design your hair by text and reference image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18072–18081, 2022.

Wei, T., Chen, D., Zhou, W., Liao, J., Zhang, W., Hua, G., and Yu, N. Hairclipv2: Unifying hair editing via proxy feature blending. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 23589–23599, 2023.

Wen, Y., Kirchenbauer, J., Geiping, J., and Goldstein, T. Tree-ring watermarks: Fingerprints for diffusion images that are invisible and robust. *arXiv preprint arXiv:2305.20030*, 2023.

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. PMLR, 2022.

Xiong, C., Qin, C., Feng, G., and Zhang, X. Flexible and secure watermarking for latent diffusion model. In *Proceedings of the 31st ACM International Conference on Multimedia*, pp. 1668–1676, 2023.

Yu, J., Xu, Y., Koh, J. Y., Luong, T., Baid, G., Wang, Z., Vasudevan, V., Ku, A., Yang, Y., Ayan, B. K., et al. Scaling autoregressive models for content-rich text-to-image generation. *Transactions on Machine Learning Research*, 2022.

Yu, N., Skripniuk, V., Abdelnabi, S., and Fritz, M. Artificial fingerprinting for generative models: Rooting deepfake attribution in training data. In *Proceedings of the IEEE/CVF International conference on computer vision*, pp. 14448–14457, 2021.

Yuan, L., Chen, D., Chen, Y.-L., Codella, N., Dai, X., Gao, J., Hu, H., Huang, X., Li, B., Li, C., et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021.

Zhang, K. A., Xu, L., Cuesta-Infante, A., and Veeramachaneni, K. Robust invisible video watermarking with attention. 2019.

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

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Zhao, W., Bai, L., Rao, Y., Zhou, J., and Lu, J. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *arXiv preprint arXiv:2302.04867*, 2023a.

Zhao, X., Zhang, K., Su, Z., Vasan, S., Grishchenko, I., Kruegel, C., Vigna, G., Wang, Y., and Li, L. Invisible image watermarks are provably removable using generative ai. *Saastha Vasan, Ilya Grishchenko, Christopher Kruegel, Giovanni Vigna, Yu-Xiang Wang, and Lei Li,"Invisible image watermarks are provably removable using generative ai," Aug*, 2023b.

Zhao, X., Zhang, K., Wang, Y.-X., and Li, L. Generative autoencoders as watermark attackers: Analyses of vulnerabilities and threats. *arXiv preprint arXiv:2306.01953*, 2023c.

Zhao, Y., Pang, T., Du, C., Yang, X., Cheung, N.-M., and Lin, M. A recipe for watermarking diffusion models. *arXiv preprint arXiv:2303.10137*, 2023d.

Zhu, J., Kaplan, R., Johnson, J., and Fei-Fei, L. Hidden: Hiding data with deep networks. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 657–672, 2018.

A. More Discussions

A.1. Visual Examples of Various VAE Decoders

The customization of the SD primarily resides within the U-Net structure. Thus, altering the VAE decoder retains the model's custom content intact. We tried 4 different clean VAE decoders, namely sd-vae-ft-mse (sd-vae-ft mse), ClearVAE (ClearVAE), ConsistencyDecoderVAE (Betker et al., 2023), and Stable Signature (Fernandez et al., 2023) to replace the original decoder. We evaluated the sampling results by three metrics, *i.e.*, DreamSim, PSNR, and SSIM. As shown in Table 7, replacing the VAE decoder will not degrade the functionality of SDs. Our method selects more primary components (*i.e.*, U-Net structure) of the SD model to embed the watermark, replacing UNet will destroy the functionality or customization of the SD.

Table 7. Quantitative comparison of visual similarity for different VAE decoders.

	DREAMSIM ↓	SSIM ↑	PSNR ↑
SD-VAE-FT-MSE	0.013	0.82	27.54
CLEARVAE	0.033	0.80	25.99
CONSISTENCYDEC.	0.030	0.70	24.68
STABLESIG.	0.022	0.79	25.92

Moreover, we provide a visual example (see Figure 5). It can be seen that the results generated by various VAE decoders only have very slight differences.

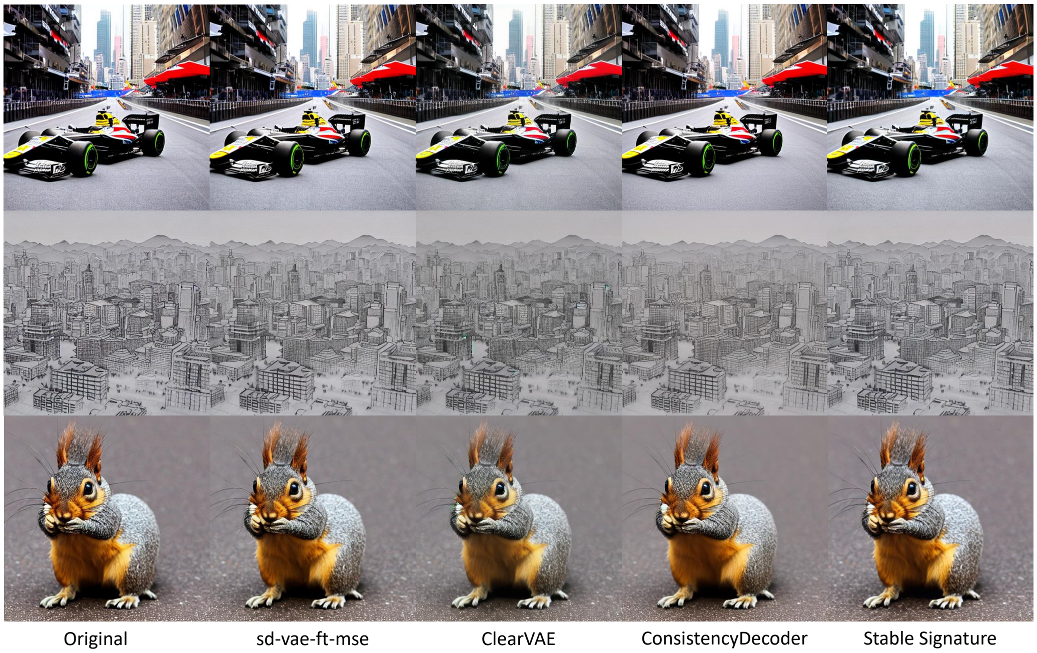

Figure 5. Representative visual examples of Stable Diffusion generated results decoded by different VAE decoders.

A.2. Model collusion

Users might deceive detection by aggregating their models to average their model weights, as in Model soups (Wortsman et al., 2022), creating a new model. Here, we merge two watermark models with different watermark bit strings at a ratio of 0.5. We discover that the bit at position l output by the extractor will be 0 (respectively, 1) when the l-th bits of both models are 0 (respectively, 1), and the extracted bit is random when their bits disagree. Table 8 displays the average values (Expectation) of the extracted results from the merged model under different bit settings for Models 1 and 2, proving the aforementioned findings.

Table 8. Expectation of extracted bits. The term "Model1-0" refers to the positions where the watermark bits of the first model are 0. Similarly for other terms.

EXPECTATION	MODEL2-0	MODEL2-1
MODEL1-0	0.04	0.53
MODEL1-1	0.47	0.97

This aligns with the findings reported by Fernandez et al. which conforms to the *marking assumption*. This so-called *marking assumption* plays a crucial role in the literature on traitor tracing (Furon et al., 2012; Meerwald & Furon, 2012). It's interesting even though our watermarking process was not explicitly designed for this, it still holds.

A.3. Scaling `AquaLoRA` to More Bits

Although we used a 48-bit watermark in our main experiment, we also tried extending our watermark to more bits. We increased the rank size to 512 and then tested watermarks of 64 bits and 100 bits. Table 9 shows the experimental results, demonstrating that our method can successfully scale up to 64 bits with little performance loss. For 100 bits, there is a moderate decrease in watermark performance, which we leave as a topic for future study.

Table 9. Performance of our method on 64-bit and 100-bit settings. Adversarial here refers to the average performance of many different distortions.

NUMBER OF BITS	FIDELITY		ROBUSTNESS	
	FID ↓	DREAMSIM ↓	BIT ACC. ↑	BIT ACC.(ADVERSARIAL) ↑
64-BIT	24.53	0.229	94.47	88.36
100-BIT	24.72	0.238	90.11	83.45

A.4. Computational and Time Complexity for Training and Inference

The training phase is divided into latent watermark pre-training and prior-preserving fine-tuning. During the latent watermark pre-training phase, we train for 40 epochs, approximately 80k steps, costing 40 GPU hours on a single A6000 40G. In PPFT, we train for 30 epochs, about 30k steps, costing 15 GPU hours on an A6000 40G. This is acceptable for any normal-sized academic laboratory. It's important to note that for all customized models, we only need to pre-train a coarse-type quantity of AquaLoRA.

During the inference phase, since LoRA has been integrated into the model weights, **there is essentially 0 overhead**. At this stage, our overhead is lower than that of post-diffusion or image watermarking methods.

A.5. Inherent Shortcomings for Cover-agnostic Watermarks

Cover-agnostic watermark has inherent weaknesses. Consider if the attacker averages many latent vectors, he will estimate the watermark signal Δz_w. However, in practice, it requires collecting a large number of in-distribution unwatermarked samples, which remains challenging. Furthermore, the above attack can be mitigated by the following measures:

1. Dynamic watermarks: Regularly update the watermark pattern or parameters, making it difficult for attackers to track and analyze the watermark over time.

2. Apply watermarks only to significant content, reducing the number of samples available for attackers to analyze.

B. Details in Latent Watermark Pre-training

B.1. Network Architecture of Secret Encoder and Training Strategy

In the design of the secret encoder, we have drawn inspiration from the model structure of RoSteALS (Bui et al., 2023). We changed the resolution from 16 to 32 to enhance the watermark's robustness against cropping operations. We removed the final zero convolution because it had a minor impact on the training process. Instead, it tended to slow down the training speed.

Regarding our training strategy, in the initial phase of training, we retained only the $\mathcal{L}_{\text{BCE}}$ and did not use natural image datasets. Instead, we directly used the output of the secret encoder as the input for the VAE decoder for training, demanding accurate watermark extraction.

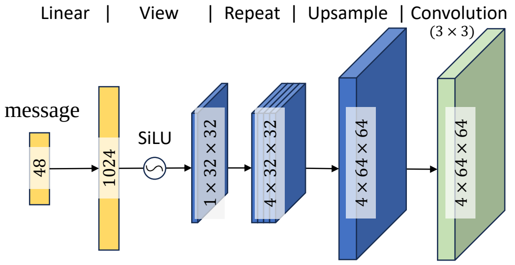

Figure 6. Network architecture for latent watermark encoder.

Only when the average loss over 10 iterations fell below 0.1, did we introduce the natural image dataset, MSCOCO2017 trainset, into the training process. From that point on, we trained using the complete loss as mentioned in the main body. Without this training strategy, we find that the loss can hardly decrease.

B.2. Details of the Peak Regional Variation Loss

During the training process, we found that areas with particularly strong signal features severely affect the visual quality (see Figure 7 w/o PRVL). Although increasing the scale of LPIPS loss can suppress this effect, it also leads to overall suppression of the watermark in invisible areas, making it difficult to improve extraction accuracy. Therefore, we specifically designed the Peak Regional Variation Loss (PRVL) which is engineered to focus on the maximal discrepancy within a predefined window or region. This is achieved by computing the absolute difference between the corresponding pixels of the two images and aggregating these differences across all color channels to form a combined variation map. The loss then centers on the region exhibiting the peak variation, identified via a convolution operation with a uniform kernel over the combined map. This approach ensures that PRVL is not unduly influenced by widespread, low-level variations but rather emphasizes areas of maximal discrepancy. Specifically, this loss can be formalized as follows:

$$V(x,y) = \frac{1}{3}\sum_{c=1}^{3} |I_c^o(x,y) - I_c^w(x,y)| \tag{7}$$

$$\mathcal{L}_{\text{PRVL}} = \max_{x,y}(V * K)(x,y). \tag{8}$$

$V(x,y)$ is a 2D tensor representing the average variation at each pixel. K represents a uniform convolution kernel used to aggregate localized variations over a defined window size. $I_c(x,y)$ represent an image, c is the corresponding channel and x, y stand for position of a pixel. We show the results with and without PRVL loss in Figure 7.

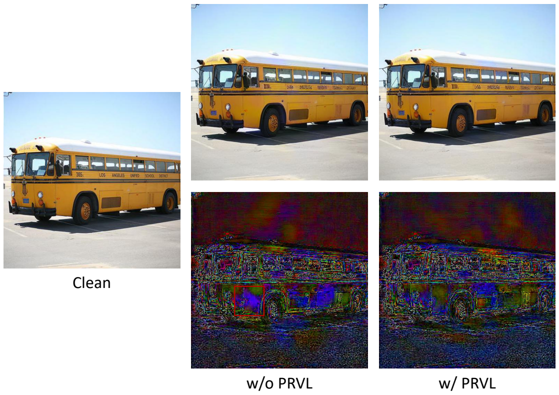

Figure 7. Representative examples of our latent watermark with and without PRVL loss. The residual is amplified 10× for visualization.

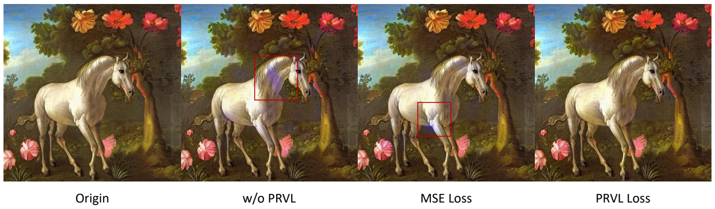

Figure 8. More ablation on PRVL Loss. We tested three different settings: simply removing PRVL Loss from our training scheme, replacing PRVL Loss with a similarly sized MSE Loss, and the original training scheme. It can be observed that the watermark results from the first two settings exhibit artifacts.

C. Details of the Distortion Settings

C.1. Distortion Simulation Layer in Training Stage

In our training, we employed JPEG, crop and resize, Gaussian blur, Gaussian noise, and color jitter as our distortion simulation layers. For JPEG, we utilized the simulation layer from HiDDeN(Zhu et al., 2018) for JPEG distortion. For the other distortions, we used `RandomCrop` and `Resize` from `torchvision`, initially randomly altering the width and height within the range of $[256, 512]$, and then resizing back to 512×512. `RandomGaussianBlur`, `RandomGaussianNoise`, and `ColorJiggle` are from the `Kornia` library. For `RandomGaussianBlur`, we randomly chose a kernel size from

$[3, 9]$ with an intensity selection of $(0, 2]$. For `RandomGaussianNoise`, we set the mean to 0 and the variance to 10. For `ColorJiggle`, we adjusted the brightness in $(0.8, 1.25)$, contrast in $(0.8, 1.25)$, saturation in $(0.8, 1.25)$, and hue in $(-0.2, 0.2)$.

Additionally, to enhance the watermark robustness on images sampled at larger sizes, we propose a new augmentation, which we discuss in detail in Appendix D.

C.2. Distortion in Evaluation Stage

During the testing phase, we applied lossy compression to the images using JPEG, employing the `PIL` library with a quality setting of 50. For cropping, we used a random crop of 80%. Gaussian blur, Gaussian noise, and color jitter were applied using functions from the `Kornia` library. In Gaussian blur, we used a kernel size of 3×3 with an intensity of 4. For Gaussian noise, we set the mean to 0 and the variance to 0.1 (image is normalized into $[0, 1]$). In color jitter, we sampled brightness from $(0.9, 1.1)$, contrast from $(0.9, 1.1)$, saturation from $(0.9, 1.1)$, and hue from $(-0.1, 0.1)$.

For denoising, we used Stable Diffusion v1.5, with a noise strength of 0.1, meaning the added noise was equivalent to the noise intensity of 100 steps in the DDPM forward process (Stable Diffusion has a total of 1000 timesteps). For denoising-v2, we employed Stable Diffusion v2.1, with a noise strength of 0.2. Existing watermarking methods often add watermarks at the pixel level, whereas current generative models compress or regenerate images at the semantic level, significantly leading to the loss of watermark information. This was experimentally demonstrated in (Zhao et al., 2023b), showing that compression can significantly eliminate image watermarks. Differently, our watermarks are added to the latent space, where they are more prominent.

Figure 9 shows the various visual results of distortions that we used during the evaluation.

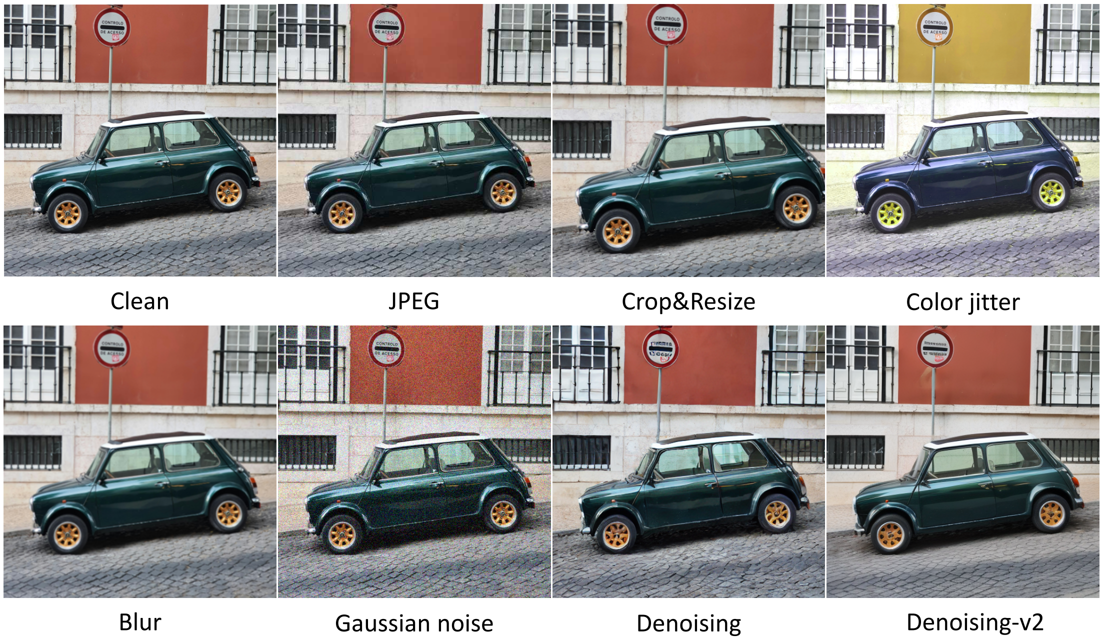

Figure 9. Demonstration of the various visual results of distortions that were used during the evaluation

D. Details of Robustness Enhancement on Larger Sampling Sizes

We designed an augmentation to enhance the watermark extraction capability of our method at larger sampling sizes, and after the PPFT stage, freeze all remaining weights to conduct additional fine-tuning on the decoder.

Augmentation. We considered the following question: How does a watermark pattern behave under larger sampling sizes after being trained on 512×512 watermarks? We observed that the patterns stayed almost the same as the original size watermark pattern near the four corners. Based on this, we designed the following process during the latent watermark pre-training phase, allowing the secret encoder and decoder to optimize this observation. Specifically, we divided the

$4 \times 64 \times 64$ watermark pattern Δz_w into four $4 \times 32 \times 32$ patches, then resized the $4 \times 64 \times 64$ image latent z_o to 1 to 1.5 times its original size to simulate a larger image. Afterward, we overlaid the four patches onto the corners of the image latent to produce the watermarked image latent z_w and then resized z_w back to the original size. The VAE decoder decodes this to yield the watermarked image I_w, completing the watermarking process.

Fine-tuning. In addition to adding augmentation, we incorporated a decoder-only fine-tuning step after PPFT training to enhance the extraction accuracy. Specifically, we used the Stable-Diffusion-Prompts (Gustavosta) as the prompt dataset and sampled images of varying sizes, ranging from 512×512 to 768×768, using different secret messages. Subsequently, we utilized the decoder to extract the secret message from the images and calculated the BCE Loss. During the training process, we froze all weights except for those of the decoder, focusing on enhancing the decoder's capabilities.

E. Mathematical Proof of PPFT

In this proof, we omit the text condition c of the model. Assuming that the original distribution of the model is q with parameters ϑ, our objective is to learn the target distribution q' with model parameters θ. Formally, we define q' as a distribution that satisfies:

$$q(z_0) = q'(z_0 + \Delta z_w) = q'(z'_0). \tag{9}$$

Assume $z_0 \sim q$, $z'_0 = z_0 + \Delta z_w$, $z'_0 \sim q'$. We use p_θ to represent the distribution of the target model. Starting from ELBO, it's evident that we wish for the distribution p_θ of the target model to minimize L:

$$\mathbb{E}\left[-\log p_\theta\left(z'_0\right)\right] \leq \mathbb{E}_{q'}\left[-\log \frac{p_\theta\left(z'_{0:T}\right)}{q'\left(z'_{1:T} \mid z'_0\right)}\right] = \mathbb{E}_{q'}\left[-\log p\left(z'_T\right) - \sum_{t \geq 1} \log \frac{p_\theta\left(z'_{t-1} \mid z'_t\right)}{q'\left(z'_t \mid z'_{t-1}\right)}\right] =: L. \tag{10}$$

Further, we can deduce:

$$L = \mathbb{E}_{q'}[\underbrace{D_{\mathrm{KL}}\left(q'\left(z'_T \mid z'_0\right) \| p\left(z'_T\right)\right)}_{L_T} + \sum_{t>1} \underbrace{D_{\mathrm{KL}}\left(q'\left(z'_{t-1} \mid z'_t, z'_0\right) \| p_\theta\left(z'_{t-1} \mid z'_t\right)\right)}_{L_{t-1}} \underbrace{- \log p_\theta\left(z'_0 \mid z'_1\right)}_{L_0}]. \tag{11}$$

L_T can be treated as a constant, L_0 can be seen as a type of distortion and can be ignored. Only consider $L_{1:T-1}$.

Considering $q(z_{t-1}|z_t, z_0)$, since q is the distribution of model ϑ, we can directly derive the mean:

$$\tilde{\mu}_t(z_t, z_0) = \mu_\vartheta(z_t) = \frac{1}{\sqrt{\alpha_t}}\left(z_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\epsilon_\vartheta(z_t, t)\right), \tag{12}$$

and variance $\tilde{\beta}_t := \frac{1-\alpha_{t-1}}{1-\alpha_t}\beta_t$.

Importantly, due to definition Equation 9, we can obtain

$$q'\left(z'_{t-1} \mid z'_t, z'_0\right) = q\left(z_{t-1} \mid z_t, z_0\right). \tag{13}$$

From this, it is known that the mean of $q'\left(z'_{t-1} \mid z'_t, z'_0\right)$ is $\mu_\vartheta(z_t)$.

Based on the KL divergence formula:

$$KL(p, q) = \log \frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2}$$

The variance of $q'\left(z'_{t-1} \mid z'_t, z'_0\right)$ is a fixed value, and the variance of $p_\theta\left(z'_{t-1} \mid z'_t\right)$ is set to be a constant related to β. Therefore, only the mean needs to be calculated.

We can obtain the effective computational part:

$$L_{t-1} = \mathbb{E}_{z'_0 \sim q'}\left[\frac{1}{2\sigma_t^2}||\tilde{\mu}_t(z'_t, z'_0) - \mu_\theta(z'_t, t)||^2\right] + C = \mathbb{E}_{z_0 \sim q}\left[\frac{1}{2\sigma_t^2}||\mu_\vartheta(z_t) - \mu_\theta(z'_t, t)||^2\right] + C. \tag{14}$$

Following DDPM, we perform parameterization:

$$\boldsymbol{\mu}_\theta\left(z_t', t\right) = \tilde{\mu}_t\left(z_t', \frac{1}{\sqrt{\bar{\alpha}_t}}\left(z_t' - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta\left(z_t'\right)\right)\right) = \frac{1}{\sqrt{\alpha_t}}\left(z_t' - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta\left(z_t', t\right)\right). \tag{15}$$

By substituting Equation 12 and 15 into Equation 14, we can derive the final loss function (omit text condition c):

$$\begin{aligned}\mathcal{L}_{\mathrm{PPFT}}(\theta) :=& \mathbb{E}_{t,z_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon}_\vartheta\left(z_t, t\right) - \boldsymbol{\epsilon}_\theta\left(z_t', t\right)\right\|^2\right] \\ =& \mathbb{E}_{t,z_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}(z_0 + \Delta z_w) + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t\right) - \boldsymbol{\epsilon}_\vartheta\left(\sqrt{\bar{\alpha}_t}z_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t\right)\right\|^2\right].\end{aligned} \tag{16}$$

F. Details of Comparison Experiments

Bit Accuracy. Assuming a k-bit binary watermark $s \in \{0,1\}^k$ is injected into the target model, and the bit string extracted from the sampled generated image is s', bit accuracy is defined as the ratio of the number of matching bits between s and s' to k, defined as $Acc(s, s')$.

TPR with Controlled FPR. We consider all the watermark approach as a single-bit watermark, with a fixed watermark s. A threshold value τ, which ranges from 0 to k, is predetermined. If the accuracy score $Acc(s, s')$ meets or exceeds the threshold τ, it is concluded that the image indeed contains the watermark.

Previous research (Yu et al., 2021) commonly assumed that watermark bits $s_1', \ldots, s_k'$ retrieved from clean images are random and uniformly distributed, each bit s_i' being modeled by a Bernoulli process with a success probability of 0.5. Consequently, the accuracy measure $Acc(s, s')$ adheres to a binomial distribution characterized by the parameters $(k, 0.5)$. Once the distribution of $Acc(s, s')$ is determined, the false positive rate (FPR) is defined as the probability that $Acc(s, s')$ of a vanilla image exceeds the threshold τ. This probability can be further expressed using the regularized incomplete beta function $B_x(a; b)$,

$$FPR(\tau) = P(Acc(s, s') > \tau) = \sum_{i=\tau+1}^{k}\binom{k}{i}\frac{1}{2^k} = B_{\frac{1}{2}}(\tau+1, k-\tau). \tag{17}$$

We estimate the false positive rate (FPR) to be maintained at 10^{-6}, determine the respective threshold τ, and evaluate the true positive rate (TPR) using 1,000 watermarked images. Refer to Table 2, where the FPR is kept at 10^{-6}, our method demonstrates commendable performance in terms of bit accuracy and TPR.

G. More Robustness Results

For the downloaded models, users have the option to add additional LoRA and ControlNet during the image generation process. Consequently, we conducted corresponding tests to assess this capability.

G.1. Apply LoRA

Table 10. Bit accuracy for watermarked models with LoRA add-on.

	Character + style LoRA(Ghibli style)	Style + character LoRA(Shadowheart)
Bit accuracy (%) ↑	92.64	93.91

In this experiment, we tested two scenarios: In the first scenario, we used a watermarked character model and added a LoRA related to a style. In the second scenario, we employed a watermarked style model and added a LoRA concerning a character. For both scenarios, we fixed the chosen LoRA and randomly selected 4 models for testing, sampled 100 images each then calculated the average. Table 10 shows the results of our experiment. It can be observed that, although the addition of LoRA had some impact on performance, the accuracy remained above 92%, demonstrating good robustness.

G.2. Apply ControlNet

In this experiment, we tested all types of the 1.0 version of ControlNet. For canny, depth, hed, MLSD, normal, and segmentation, we randomly selected 4 models from the style group. For openpose, we chose 4 models from the celebrity type. We sampled 100 images from each model to calculate the average bit accuracy. Table 11 displays the results of our experiment, which show that ControlNet did not impact the extraction of watermarks.

Table 11. Bit accuracy for watermarked models with ControlNet add-on.

	CANNY	DEPTH	HED	MLSD	NORMAL	SEG	OPENPOSE
BIT ACCURACY (%)↑	94.21	95.39	93.21	95.15	95.87	95.36	94.31

G.3. Apply Textual Inversion

We tested the impact of two types of textual inversion, style and character, on the accuracy of watermark extraction. Our tests were based on the RealCartoon3D model. As can be seen from Table 12, our method achieves good accuracy.

Table 12. Bit accuracy for watermarked models with LoRA add-on.

	STYLE TEXTUAL INVERSION (MONET STYLE)	CHARACTER TEXTUAL INVERSION (NATALIE)
BIT ACCURACY (%)↑	94.79	93.34

G.4. Apply LCM-LoRA

LCM-LoRA(Luo et al., 2023), a LoRA model trained with Stable Diffusion base models using the consistency method, accelerates image generation to as few as four steps with any custom checkpoint model. We tested the impact of integrating LCM-LoRA and found an increase in extraction accuracy, with bit accuracy reaching 97.04%. We hypothesize this is due to LCM-LoRA's generated images having less content and smoother colors compared to normal sampling, making our watermark more pronounced.

G.5. Robustness Against Fine-tuning

We also took into account that some advanced attackers, after obtaining the model weights, would fine-tune the model in an attempt to eliminate the watermark. It is obvious that the value of customized models lies in their inclusion of unique preferences, which are incorporated during the author's training process and are usually not publicly released. Therefore, attackers can usually only use some easily accessible public datasets to fine-tune.

Thus, we conducted corresponding experiments. Specifically, we conducted a fine-tuning attack on a watermarked model (realcartoon3d_v12). We used AquaLoRA with ranks of 320 and 512 to add watermarks, respectively. Then, we used the MSCOCO dataset to fine-tune the model, and we statistically analyzed the model's performance at different training steps.

Our findings, as shown in Figure 10, indicate that as the fine-tuning progresses, the watermark is destroyed, which also significantly compromises the integrity of the original content. Encouragingly, the figure shows that larger ranks exhibit better robustness against fine-tuning, suggesting that future research could explore using even larger ranks.

H. More Ablation Results

H.1. Fidelity & Accuracy Trade-off

We can modify the balance between accuracy and fidelity by altering the α value. This trade-off is depicted in Figure 11, which shows how changes in α affect both the accuracy of extraction and the fidelity of images. After considering these variations, we decided to set α at 1.05.

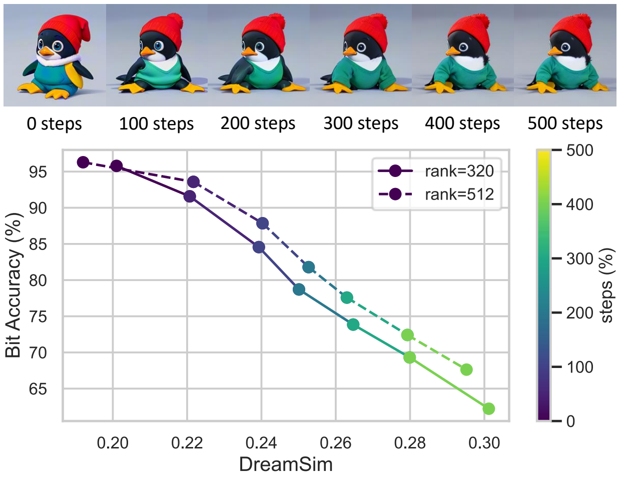

Figure 10. Robustness to model fine-tuning. The image illustrates that the original style embedded in the model gradually diminishes with fine-tuning.

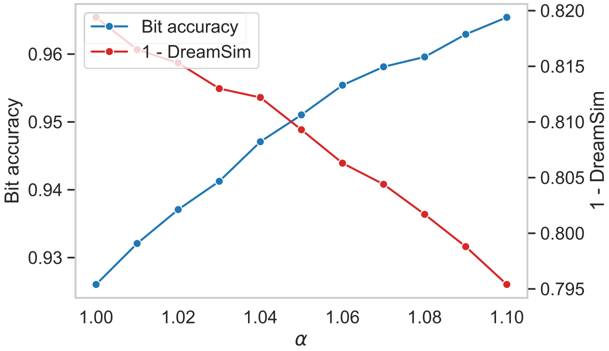

Figure 11. Trade-off Between Extraction Accuracy and Image Fidelity at Different Alpha Settings.

I. Evaluated Models and Coarse Types

See Table 13.

J. More Visual Results

See Figure 12.

Table 13. The names of the 25 models used in our experiment and their respective coarse types.

Model	Group	Coarse type
counterfeitV30_25	style	anime
cuteyukimixAdorable_kemiaomiao	style	anime
divineelegancemix_V9	style	anime
meinamix_meinaV11	style	anime
maturemalemix_v14	style	anime
deliberate_v11	style	photorealistic
photon_v1	style	photorealistic
dreamshaper_8	style	cartoon3d
juggernaut_reborn	style	cartoon3d
revAnimated_v122EOL	style	cartoon3d
realcartoonRealistic_v12	style	cartoon3d
realcartoon3d_v12	style	cartoon3d
lyriel_v16	style	art
neverendingDreamNED_v122NoVaeTraining	style	art
ghostmix_v20Novae	style	art
famousPeople_caityLotz	character	celebrity
famousPeople_evaGreen	character	celebrity
famousPeople_sophieTurner	character	celebrity
famousPeople_aoc	character	celebrity
fenris_v10FP16	character	animal
fluffykavkarockmergi_v10	character	animal
chilloutmix_NiPrunedFp32Fix	character	person
perfectdeliberate_v5	character	person
realisian_v50	character	person
robot_V2	character	robot

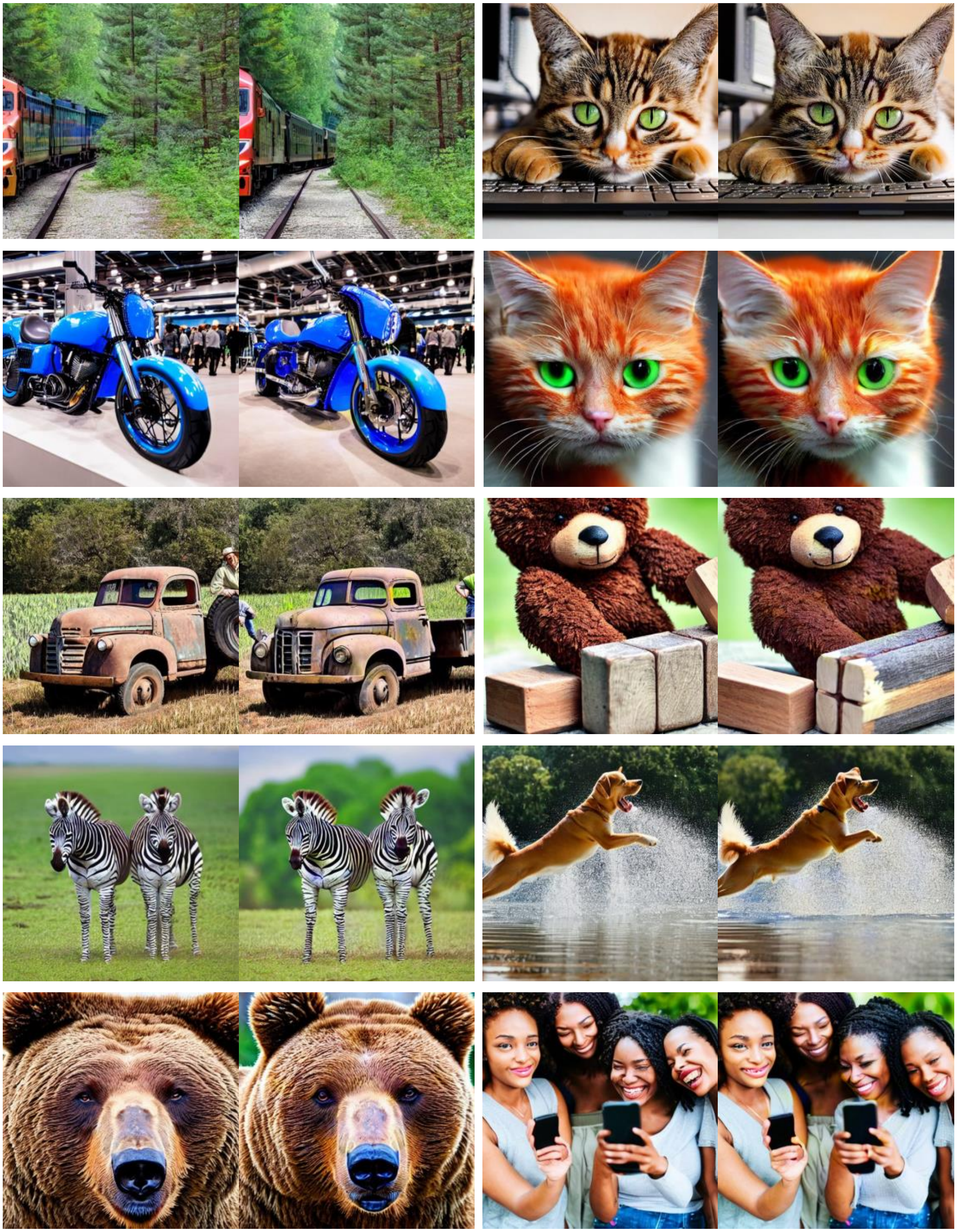

Figure 12. Comparison between images generated by the original Stable Diffusion model and the watermarked Stable Diffusion model under the same diffusion configurations and random seed. **Left:** The results generated from the original model. **Right:** The results generated from the watermarked model. The results showed that the watermarked generated image is still very close to the original one.