# PrivateSeal: Low-Sensitivity Latent Directions for Diffusion-Resilient User-Specific Watermarking

**Yiheng Chen**[1] **Zhong Ji**[1,†] **Zhihao Li**[2,†] **Boyu Wang**[2,3]
[1] Tianjin University [2] Western University [3] Vector Institute
{chenyiheng,jizhong}@tju.edu.cn zli3446@uwo.ca bwang@csd.uwo.ca

## Abstract

Diffusion-driven image editing and regeneration are becoming increasingly widespread. This creates an urgent need for watermarking methods that can embed imperceptible yet verifiable signals into existing images for ownership attribution and provenance tracking. Existing methods fail to simultaneously satisfy three practical requirements: robustness against diffusion-based transformations, non-iterative verification, and scalable support for per-user key assignment at low overhead. To address these challenges, we propose **PrivateSeal**, a watermarking framework for pre-existing images under diffusion-based transformations. The perturbation is encouraged to lie along low-sensitivity latent directions, so that the embedded signal is less likely to be suppressed during diffusion-based editing or regeneration. During verification, the embedded message is reliably recovered via a simple latent-space projection using the corresponding key. This design allows a platform to assign independent projection keys to different users, accounts, or images without retraining or modifying the verifier, while maintaining non-iterative verification. Extensive experiments on the W-Bench benchmark show that PrivateSeal achieves competitive robustness against diffusion-based regeneration and editing, with additional cross-model and cross-dataset evaluations further validating its strong transferability. Code is available at `https://github.com/yiheng-lab/PrivateSeal`.

## 1 Introduction

Diffusion models are now widely used for image generation and editing [1, 2, 3]. Recent diffusion-based image editing systems can modify an existing image according to a text prompt [4, 5, 6]. Although the edited result may preserve the main objects and layout, it can redraw colors, textures, and local details [4, 6]. This capability poses a fundamental challenge to image watermarking for ownership attribution and content provenance [7, 8, 9]. Unlike classical image distortions (e.g., compression, filtering, geometric transformations), diffusion-based transformations partially re-synthesize the image through iterative denoising [10, 5, 11], rendering watermark signals that are reliable under conventional corruptions difficult to recover after editing or regeneration.

One line of research embeds watermark signals directly into the image generation process. For example, Tree-Ring [12] and Gaussian Shading [13] encode watermark information into the initial noise or latent variables of diffusion sampling, ensuring the watermark propagates consistently with the generated image. These methods are well suited for watermarking outputs produced by a specific generation pipeline, where the watermark can be injected at sampling time. However, they do not directly support protecting real or already-created images. In practical scenarios, most images have already been created and need to be watermarked for ownership attribution after subsequent diffusion-based editing and regeneration.

[†]Corresponding authors

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

| | In-generation (e.g., Tree-Ring) | Diffusion-subspace (e.g., Shallow Diffuse) | Encoder-decoder (e.g., VINE) | PrivateSeal (Ours) |
|---|---|---|---|---|
| Watermark Already-created images | ✗ No | ✓ Yes | ✓ Yes | ✓ Yes |
| Diffusion Resilience | ◔ Limited | ✓ Yes | ✓ Yes | ✓ Yes |
| Non-iterative verification | ◔ Limited | ✗ No | ✓ Yes | ✓ Yes |
| Flexible user-specific keys | ✗ No | ◔ Limited | ✗ No | ✓ Yes |

Figure 1: Qualitative comparison of representative watermarking paradigms and methods. The first three columns summarize existing representative design paradigms, with example methods shown in parentheses, including Tree-Ring [12], Shallow Diffuse [7], and VINE [8], while the final column shows our proposed method. The first row denotes the user-side applicability condition, namely whether a method can insert watermarks after image creation without access to the original creation process. The remaining three rows correspond to the practical requirements emphasized in this work: (1) remaining detectable after diffusion-based editing or regeneration; (2) verifying without iterative reverse diffusion, inversion, or repeated generative sampling; and (3) assigning different keys across platforms, users, accounts, or images without key-specific neural detectors. A check mark indicates direct support, a half-filled circle indicates limited support under additional assumptions or nontrivial deployment cost, and a cross signifies no direct support.

For pre-existing images, Shallow Diffuse [7] exploits a low-dimensional latent subspace induced by diffusion models to improve watermark persistence under generative transformations, supporting watermarking of pre-existing user images. However, its verification procedure mainly relies on iterative reverse diffusion, resulting in prohibitive computational overhead that limits usage in real-world applications. Another representative method, VINE [8], adopts a learned encoder-decoder framework: the encoder watermarks an input image, while the decoder subsequently recovers the embedded message. While this design enables efficient watermark extraction, it introduces a deployment dilemma for user-specific verification: (1) If the decoder is released for public verification, it may serve as a queryable oracle: attackers can repeatedly modify, edit, or regenerate a watermarked image, query the decoder, and stop once the watermark is no longer recovered. (2) If the decoder is kept private, verification must be centralized. (3) If each user requires an independent checking rule, separate user-specific decoders must be trained and distributed, which is prohibitively expensive. In summary, as illustrated in Figure 1, a deployable user-side watermarking system should simultaneously satisfy three requirements: (1) robustness against diffusion-based editing and regeneration, (2) non-iterative verification, and (3) scalable, low-cost support for user-specific keys. To the best of our knowledge, no existing method meets all three.

To address these challenges, we propose PrivateSeal, a watermarking framework built on a key insight: diffusion models exhibit structured sensitivity in their latent space, where certain perturbation directions are far more resilient to iterative denoising than others. By restricting watermark signals to the low-sensitivity directions in the diffusion latent space, PrivateSeal endows the watermark with intrinsic robustness against diffusion-based editing and regeneration. Concretely, given a user-specific key, PrivateSeal optimizes an image-dependent latent perturbation that encodes a message under that key along these insensitive directions. At verification time, the message is recovered by a lightweight latent-space projection using the same key, eliminating the need for costly reverse diffusion or inversion. We evaluate PrivateSeal on the W-Bench benchmark [8], covering a comprehensive suite of attacks: diffusion-based regeneration, global editing, and local editing. Experiments demonstrate that PrivateSeal achieves competitive robustness against these attacks, with additional cross-model and cross-dataset evaluations further confirming its strong transferability across diverse diffusion architectures and data distributions.

The highlights of our proposed PrivateSeal are summarized as follows:

- We identify and exploit the low-sensitivity directions in diffusion latent space, demonstrating that perturbations along these directions are inherently resilient to iterative denoising and thus

survive diffusion-based editing and regeneration. This establishes a principled, theoretically grounded mechanism for robust latent-space watermarking.

- We propose PrivateSeal for user-side watermarking of already-created images, jointly addressing three practical requirements: (1) inherent robustness against diffusion-based transformations, (2) non-iterative verification, and (3) scalable, low-cost support for user-specific key assignment across granularity levels.
- We evaluate our method on W-Bench, showing that PrivateSeal achieves competitive robustness against a broad suite of diffusion-based image-to-image attacks. Additional cross-model and cross-dataset evaluations further validate its strong transferability across diverse diffusion architectures and data distributions.

## 2 Related Work

**Conventional Image Watermarking.** Recent work has increasingly examined societal concerns such as ownership [14] and privacy [15, 16, 17], where watermarking has attracted considerable attention. Traditional image watermarking methods embed low-amplitude signals in the spatial or frequency domains, such as DWT-DCT-based schemes [18]. Deep learning-based methods, including HiDDeN [19], StegaStamp [20], MBRS [21], and RivaGAN [22], formulate watermarking as an encoder–decoder problem and achieve strong robustness against common image distortions. However, robustness to conventional distortions does not directly translate to robustness under generative transformations. Recent studies [9, 23] show that watermark signals can be substantially weakened or removed when images are re-synthesized by diffusion-based models.

**Intrinsic Watermarking for Generative Models.** Another line of work embeds watermarks into the generative process itself. Tree-Ring [12] injects structured patterns into the initial noise of diffusion sampling, while Stable Signature [24] fine-tunes generative components so that generated images carry detectable signatures. Subsequent methods [25, 26] further improve robustness, attribution, or tamper localization for generated content. These methods are effective when the content provider controls the generative model, but they are less applicable to user-side protection, where the goal is to watermark existing user images without modifying or controlling future generative transformations.

**User-Side Watermarking under Generative Transformations.** Recent works have begun to address user-side watermarking in the presence of diffusion-based editing and regeneration. Shallow Diffuse [7] leverages a low-dimensional latent subspace induced by diffusion models to improve watermark persistence, but its extraction procedure relies on iterative reverse diffusion and compatible generative-process assumptions, which increases verification cost. VINE [8] adopts a learned encoder–decoder framework that leverages diffusion priors, enabling efficient feed-forward extraction; however, its watermarking keys are coupled with learned model parameters, making independent multi-user key assignment difficult without retraining user-specific models.

PrivateSeal takes a different approach. It retains the user-side setting but removes the need for key-specific detectors by directly optimizing an image-specific perturbation under a fixed orthonormal projection key. At verification time, only a public VAE encoder and the user's projection key are required, making verification both blind and non-iterative. This key–model decoupling enables independent key assignment, rotation, and multi-user deployment without retraining.

## 3 Method

### 3.1 Design Rationale: From Stable Perturbations to Keyed Projections

Figure 2 gives an overview of the PrivateSeal pipeline, where embedding optimizes an image-specific latent perturbation under a user-specific projection key, and verification recovers the message using only a public VAE encoder and the corresponding key.

PrivateSeal is designed for the user-side setting where already-created images need to be watermarked without access to their original creation process. Under this setting, the method addresses three practical requirements. First, the embedded signal should remain recoverable after diffusion-based editing or regeneration. Second, verification should avoid iterative reverse diffusion, inversion, or repeated generative sampling. Third, user-specific key assignment, rotation, and extension should be supported

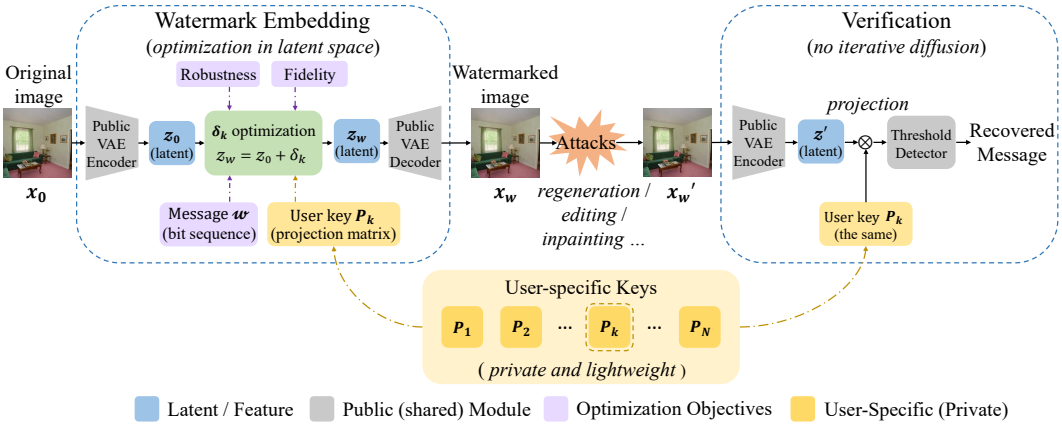


Figure 2: Overview of PrivateSeal Framework. PrivateSeal embeds a user-specific watermark by optimizing an image-specific latent perturbation guided by three complementary objectives: message fidelity, diffusion dynamics alignment, and perceptual quality preservation. At verification time, the embedded message is recovered non-iteratively using only a publicly available VAE encoder and the corresponding user-specific projection key.

without training a separate neural detector for each user. We next show how these requirements lead naturally to low-sensitivity latent perturbations and projection-key-based verification.

We begin with the robustness requirement. Let $\epsilon_\theta(z_t, t)$ denote the denoising prediction at timestep $t$. If a small latent perturbation $\delta$ is added to $z_t$, a first-order expansion gives

$$\epsilon_\theta(z_t + \delta, t) - \epsilon_\theta(z_t, t) \approx J_t \delta, \tag{1}$$

where

$$J_t = \left. \frac{\partial \epsilon_\theta(z, t)}{\partial z} \right|_{z=z_t} \tag{2}$$

is the local Jacobian of the denoising network. A perturbation with small $\|J_t\delta\|_2$ changes the predicted noise only weakly, and is therefore less likely to be attenuated by the denoising dynamics. Equivalently, if $J_t = U_t \Sigma_t V_t^\top$, then

$$\|J_t\delta\|_2^2 = \sum_i \sigma_{t,i}^2 \langle \delta, v_{t,i} \rangle^2. \tag{3}$$

This suggests that robust perturbations should concentrate their energy in the right singular directions of $J_t$ associated with small singular values.

The next question is how such robust perturbations can carry message information. Intuitively, one stable latent direction provides one robust binary readout: the watermark can be detected by whether the perturbation has a positive or negative projection along that direction. Thus, encoding an $L$-bit message requires $L$ independent readouts. PrivateSeal uses a user-specific projection matrix $P \in \mathbb{R}^{L \times D}$ as the private key, where each row defines one latent readout direction.

At verification time, a suspicious image $x_{\text{suspicious}}$ is encoded into a latent representation $z_{\text{suspicious}}$, and the message is recovered by

$$\hat{w} = \text{sign}(P \cdot \text{vec}(z_{\text{suspicious}})), \tag{4}$$

where $\text{vec}(\cdot)$ denotes flattening the latent tensor into a vector.

Thus, the embedding process should place the watermark perturbation along the right singular directions of $J_t$ associated with small singular values, while enforcing the latent representation of the watermarked image to remain aligned with the target message after projection by $P$. This leads to an optimization framework that jointly supports robustness, non-iterative verification, and low-cost user-specific key assignment.

### 3.2 Projection Key

PrivateSeal enables non-iterative verification and scalable key assignment by using a lightweight projection matrix as each user-specific key. To reduce crosstalk between message bits, we constrain the projection key $P \in \mathbb{R}^{L \times D}$ to have orthonormal rows, which means $PP^\top = I_L$.

Under this constraint, different bit readouts are decoupled and have the same projection scale. Any row-orthonormal matrix can serve as a valid projection key. In practice, we obtain $P$ by sampling a random matrix and orthonormalizing its rows.

### 3.3 Key-Conditioned Latent Embedding

The embedding stage optimizes a key-decodable perturbation for diffusion robustness. Given an input image $x$, we first encode it into the latent space as $z_0 = \mathcal{E}(x)$, then construct the watermarked latent as $z_w = z_0 + \delta$, where $\delta$ is optimized for the specific image, message, and projection key.

**Message Objective.** For a target message $w \in \{-1, 1\}^L$, the message objective is

$$\mathcal{L}_{\text{message}} = \|P \cdot \text{vec}(z_w) - \alpha w\|_2^2 , \tag{5}$$

where $\alpha > 0$ is the target message strength.

Equivalently, this objective imposes linear constraints on the perturbation:

$$P \cdot \text{vec}(\delta) \approx \alpha w - P \cdot \text{vec}(z_0). \tag{6}$$

Because $L \ll D$, these constraints do not uniquely determine $\delta$. The remaining degrees of freedom are used to optimize robustness and fidelity.

**Robustness Objective.** The message objective makes the watermarked latent decodable under the selected key, but it does not by itself ensure that the message remains recoverable after diffusion-based editing. We therefore introduce a robustness objective with two complementary components: proxy diffusion stability and reconstruction consistency.

First, to improve persistence under diffusion-based transformations, we regularize the perturbation using a proxy denoising model. Given

$$z_t = \alpha_t z_0 + \sigma_t \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I), \tag{7}$$

we define

$$\mathcal{L}_{\text{sta}} = \mathbb{E}_{t,\epsilon} \left[ \|\epsilon_\theta(z_t + \alpha_t \delta, t) - \epsilon_\theta(z_t, t)\|_2^2 \right]. \tag{8}$$

This term discourages perturbations that strongly affect the proxy denoising prediction. In practice, we instantiate $\epsilon_\theta$ with the denoising U-Net of Stable Diffusion 1.5. This proxy model transfers well in practice: a perturbation optimized against it remains robust under other diffusion backbones, as will be shown in our cross-model evaluation (Table 3).

Second, since the final watermarked image is obtained by VAE decoding and verification relies on VAE encoding, the perturbation should remain observable after decoding and re-encoding. We sample a differentiable pixel-space augmentation $\mathcal{A}(\cdot)$ from a set of transformations (e.g., random blur, Gaussian noise, and color jitter) and apply the same augmentation to both decoded images:

$$\hat{z}_w = \mathcal{E}(\mathcal{A}(\mathcal{D}(z_w))), \qquad \hat{z}_0 = \mathcal{E}(\mathcal{A}(\mathcal{D}(z_0))), \tag{9}$$

We define the retained perturbation as $\hat{\delta} = \hat{z}_w - \hat{z}_0$, and the decode–reconstruction consistency loss is

$$\mathcal{L}_{\text{rec}} = 1 - \cos(\hat{\delta}, \delta) + \max(0, \tau - \|\hat{\delta}\|_2), \tag{10}$$

where $\tau$ is a minimum retained-energy threshold.

The overall robustness objective is

$$\mathcal{L}_{\text{robust}} = \mathcal{L}_{\text{sta}} + \mathcal{L}_{\text{rec}}. \tag{11}$$

**Fidelity Objective.** To preserve visual quality, we penalize the pixel-space discrepancy between the clean and watermarked reconstructions:

$$\mathcal{L}_{\text{fidelity}} = \|\mathcal{D}(z_w) - \mathcal{D}(z_0)\|_2^2 . \tag{12}$$

**Overall Objective.** Our goal is to minimize the total loss

$$\mathcal{L} = \lambda_{\text{message}}\mathcal{L}_{\text{message}} + \lambda_{\text{robust}}\mathcal{L}_{\text{robust}} + \lambda_{\text{fidelity}}\mathcal{L}_{\text{fidelity}}. \tag{13}$$

However, the robustness and fidelity objectives naturally favor small perturbations. Without explicit control over the perturbation magnitude, the optimization may produce a weak watermark that is visually imperceptible but difficult to recover after transformations. We therefore impose a fixed-energy constraint on the latent perturbation as $\|\delta\|_2 = \xi$, where $\xi$ is the watermark strength budget.

In practice, we enforce this constraint by normalizing an unconstrained variable $\delta_{\text{raw}}$ at each iteration:

$$\delta = \xi \cdot \frac{\delta_{\text{raw}}}{\|\delta_{\text{raw}}\|_2}. \tag{14}$$

This turns watermark embedding into a fixed-budget directional optimization problem: the perturbation magnitude is controlled by $\xi$, while the optimization searches for a direction that satisfies the key-defined message constraints and improves robustness and fidelity.

### 3.4 Flexible Key Provisioning

Key provisioning addresses scalable multi-user deployment. A projection key in PrivateSeal is not tied to the parameters of a neural detector. Given a valid key $P$, the embedder optimizes an image-specific perturbation whose message is recoverable under $P$, while verification uses the same public encoder and the corresponding projection key.

This decoupling allows keys to be provisioned at different granularities. For example, a platform may use a shared key for internal ownership verification, assign different keys to different users or accounts for source attribution, or rotate keys when a key is compromised. A user may also select private keys locally, or use different keys for different image collections. Changing the key only changes the projection used for embedding and verification; it does not require retraining or redistributing a key-specific neural detector.

The main text focuses on the objective design and key-provisioning mechanism. Implementation-level pseudocode for the embedding and verification procedures is provided in Appendix C.

## 4 Experiments

### 4.1 Experimental Setup

**Benchmarks, Attacks, and Baselines.** We evaluate PrivateSeal on W-Bench [8], a comprehensive benchmark for testing watermark robustness under generative transformations. Since PrivateSeal is designed for watermarking under image-to-image editing and regeneration, we focus on the image-to-image attack subset of W-Bench, including regeneration, global editing, and local editing/inpainting. Regeneration re-synthesizes the whole image using stochastic [5] or deterministic [27] generative pipelines; global editing applies text-guided image-wide modifications using Instruct-Pix2Pix [4], UltraEdit [6], and MagicBrush [28]; and local editing modifies masked regions through inpainting-style transformations, including UltraEdit and ControlNet-based inpainting [29]. Unless otherwise specified, we follow the official W-Bench configurations for the evaluated attack categories and evaluate all images at $512 \times 512$ resolution.

For cross-dataset evaluation, we additionally use a held-out subset of MS COCO [30]. Since MS COCO does not provide the editing instructions or spatial masks required by W-Bench editing attacks, this evaluation is restricted to regeneration attacks.

We evaluate against representative baselines from W-Bench, spanning three major categories: traditional transform-domain methods, deep encoder–decoder watermarking, and generative-prior-based watermarking. For readability, the main paper reports results for a subset of representative methods, including CIN [31], MBRS [21], SepMark [32], StegaStamp [20], TrustMark [25], and VINE-Robust [8]; the full comparison over all evaluated baselines is provided in Appendix E. For baseline methods, we use the official results reported in W-Bench when available. PrivateSeal is evaluated using the same W-Bench attack configurations for the evaluated attack categories.

Table 1: Representative comparison on W-Bench. Each attack column reports Bit Accuracy (%) / TPR@0.1% FPR (%). Full results across all evaluated baselines are provided in Appendix E. Best and second-best values are marked separately for each metric within each attack column.

| Method | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Regeneration | | Global Editing | | | Local Editing | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Sto | Det | Pix2Pix | Ultra | Magic | Ultra | CtrlN |
| CIN | **43.19** | 0.9847 | 0.0270 | 76.34 / 44.85 | 77.83 / 51.65 | 79.81 / 51.40 | 62.79 / 17.00 | 82.91 / 68.38 | 85.13 / 51.28 | 78.09 / 66.04 |
| MBRS | 27.37 | 0.8940 | 0.1877 | 98.72 / 99.53 | 99.46 / 99.35 | 95.56 / 83.50 | 65.97 / 7.50 | 95.12 / 88.54 | 95.40 / 99.60 | 98.97 / 89.16 |
| SepMark | 35.48 | 0.9814 | 0.0150 | 81.81 / 61.21 | 90.20 / 73.85 | 93.53 / 87.74 | 78.77 / 51.84 | 91.31 / 82.58 | 95.90 / 92.94 | 93.10 / 97.14 |
| StegaStamp | 29.65 | 0.9107 | 0.0645 | 79.04 / 91.09 | 88.05 / 92.13 | 90.36 / 93.72 | 70.08 / 51.24 | 91.28 / 91.18 | 92.54 / 98.84 | 88.74 / 99.06 |
| TrustMark | 41.27 | 0.9910 | **0.0026** | 57.00 / 9.22 | 67.26 / 34.20 | 80.23 / 77.72 | 66.21 / 43.48 | 84.52 / 85.90 | 73.43 / 76.62 | 78.59 / 59.78 |
| VINE-Robust | 37.34 | **0.9934** | 0.0063 | 88.59 / 99.66 | 95.43 / 99.98 | 92.04 / 97.46 | **84.02 / 86.86** | 91.83 / **94.58** | 94.42 / **99.96** | 88.78 / 93.04 |
| **PrivateSeal** | 36.57 | 0.9329 | 0.0582 | **99.95 / 100.00** | **99.96 / 100.00** | **97.29 / 97.59** | 79.18 / 56.98 | **95.77** / 92.34 | **97.81** / 99.16 | **99.14 / 99.12** |

**Evaluation Protocol and Metrics.** For the main W-Bench comparison, results are averaged over 1,000 watermarked images and all difficulty levels for each attack. For cross-model and cross-dataset evaluations, we keep the embedding objective, projection-key construction, payload length, and evaluation metrics unchanged unless otherwise stated. We report bit accuracy and TPR at 0.1% FPR for watermark detection. Bit accuracy is computed over a 30-bit payload. TPR@0.1% FPR is used as the primary detection metric in the main paper, since ownership verification should operate under a low false-positive regime. For each method, the detection threshold is selected on a held-out set of 1,000 unwatermarked images to achieve 0.1% FPR, and the same threshold is then used for all attack types. No per-attack threshold tuning is performed. We also report PSNR, SSIM, and LPIPS between the original and watermarked images to evaluate visual quality.

**Optimization Protocol.** PrivateSeal does not use attack-specific training or attack-specific optimization on W-Bench transformations. For each image, message, and projection key, the watermark perturbation is optimized using the proposed embedding objective. The proxy denoising model is the U-Net of Stable Diffusion 1.5 [3]. All main hyperparameters are fixed across W-Bench attacks, and implementation details are provided in Appendix D.

### 4.2 Main Results

**Attack-agnostic Robustness under Generative Transformations.** Table 1 demonstrates that PrivateSeal achieves competitive detection robustness across both diffusion-based regeneration and editing-based transformations. These results are particularly notable given that PrivateSeal is not trained or optimized on the W-Bench attack pipeline. Instead, it optimizes image-specific latent perturbations using only a proxy denoising model, yet still performs particularly well under regeneration attacks, reaching near-saturated Bit Accuracy and TPR@0.1% FPR under both stochastic and deterministic regeneration. This suggests that the proposed latent perturbations capture a form of attack-agnostic robustness rather than overfitting to specific editing pipelines.

While PrivateSeal delivers strong overall performance, it is not uniformly superior across all transformations. In particular, VINE-Robust performs better under UltraEdit global editing. This suggests that attack-trained encoder-decoder methods may retain advantages under certain editing pipelines.

**Visual Quality-Robustness Trade-off.** PrivateSeal does not achieve the highest scores on pixel-level metrics such as PSNR, SSIM, and LPIPS among the representative methods. This is an inherent trade-off: PrivateSeal allocates a fixed perturbation budget to improve robustness under generative transformations, rather than optimizing only for pixel-level fidelity. Nevertheless, as shown in Fig. 3, the visual difference between the original and watermarked images remains imperceptible to the human eye.

### 4.3 Verification Runtime and Key Specificity

One of the core design goals of PrivateSeal is non-iterative verification with scalable, low-cost user-specific key assignment. To evaluate this, we measure runtime and memory on two representative hardware platforms: a consumer-grade NVIDIA GeForce RTX 4090 and a server-class NVIDIA RTX PRO 6000 Blackwell Server Edition. Embedding time encompasses latent optimization and the final




Figure 3: Qualitative examples of original and watermarked images. PrivateSeal introduces only subtle visual changes while preserving the image appearance.

Table 2: Runtime and peak GPU memory of PrivateSeal. Runtime is measured as synchronized wall-clock time with batch size 1 at $512 \times 512$ resolution.

| Stage | Operation | NVIDIA GeForce RTX 4090 | | NVIDIA RTX PRO 6000 Blackwell SE | |
|---|---|---|---|---|---|
| | | Runtime | Peak memory | Runtime | Peak memory |
| Embedding | Latent optimization + VAE decode | 97.72 s / image | 7.23 GB | 67.38 s / image | 7.29 GB |
| Verification | VAE encode + projection decoding | 18.02 ms / image | 2.52 GB | 14.64 ms / image | 2.57 GB |
| Additional key check | Matrix projection + sign | 0.172 ms / key-image | – | 0.118 ms / key-image | – |

VAE decoding step, while verification time includes image preprocessing, one VAE encoding pass, and projection-based message decoding. Disk I/O and model loading are excluded.

As shown in Table 2, embedding remains the computational bottleneck, requiring 97.72 seconds per image on the RTX 4090 and 67.38 seconds per image on the RTX PRO 6000 Blackwell Server Edition. In contrast, end-to-end verification takes only 18.02 ms and 14.64 ms per image, respectively. This corresponds to approximately 55.5 and 68.3 images per second, and does not involve iterative optimization, inversion, or reverse diffusion. The key-dependent component is even smaller. After the VAE latent has been computed, checking one additional projection key takes only 0.172 ms on the RTX 4090 and 0.118 ms on the RTX PRO 6000 Blackwell Server Edition. Thus, in multi-tenant or unknown-owner scenarios, the dominant cost is the shared VAE encoding pass, while adding candidate user keys incurs negligible additional overhead.

We further evaluate whether the embedded message can be recovered using an incorrect projection key. Specifically, we verify 50 unattacked watermarked images with 50 randomly generated projection keys that are independent of the assigned user keys. With a 30-bit payload, this produces 75,000 decoded bits for wrong-key evaluation. The correct key recovers the message with 100.00% bit accuracy in this sanity-check setting, whereas wrong keys achieve 50.06% bit accuracy, which is close to random guessing for binary messages. This result indicates that message recovery is key-specific: replacing the assigned projection key with an unrelated key does not recover the embedded payload.

### 4.4 Generalization

**Cross-Model Robustness.** To assess cross-model transferability, we use the denoising U-Net of Stable Diffusion 1.5 (SD1.5) [3] as the proxy model during watermark optimization, and evaluate the resulting watermarked images under regeneration attacks from a diverse set of target generative models. The target models include diffusion-based models, such as SD2.1 [3] and SDXL [2], as well as flow-matching-based models, such as SD3.0, SD3.5 [1], and FLUX [33]. Except for the SD1.5 proxy reference, all target attack models are held out during watermark optimization.

We evaluate the watermarked images against stochastic regeneration attacks at multiple strengths. All hyperparameters, projection keys, payload length, and verification thresholds are kept fixed across target models unless otherwise stated. Watermarks are optimized using SD1.5 as the proxy model and evaluated under different target generative models without target-specific tuning.

As shown in Table 3, PrivateSeal maintains strong decoding performance across both diffusion-based and flow-matching-based target models. Although SDXL causes a larger degradation than the other models, the watermark remains substantially detectable, with 87.23% bit accuracy and 85.60% TPR@0.1% FPR. Interestingly, this degradation is not monotonic with model generation: newer-generation flow-matching models such as SD3.0 and SD3.5 retain 95.35% / 97.80% and

Table 3: Cross-model robustness under stochastic regeneration attacks. Results are averaged over multiple attack strengths. TPR is measured at 0.1% FPR, which is computed using the same thresholding protocol as in Sec. 4.2.

| Attack model | Robustness metrics | |
|---|---|---|
| | Bit acc. ↑ (%) | TPR ↑ (%) |
| SD1.5 (Proxy) | 99.98 | 100.00 |
| SD2.1 | 99.99 | 100.00 |
| SDXL | 87.23 | 85.60 |
| SD3.0 | 95.35 | 97.80 |
| SD3.5 | 94.85 | 96.80 |
| FLUX | 98.99 | 100.00 |

Table 4: Cross-dataset robustness under stochastic (Sto) and deterministic (Det) regeneration attacks. The evaluation protocol is kept identical across datasets, with the image source being the only changed factor. Results are averaged over multiple attack strengths. TPR is measured at 0.1% FPR, which is computed using the same thresholding protocol as in Sec. 4.2.

| Dataset | Attack | Bit acc. ↑ (%) | TPR ↑ (%) |
|---|---|---|---|
| W-Bench | Sto | 99.95 | 100.00 |
| W-Bench | Det | 99.96 | 100.00 |
| MS COCO | Sto | 99.74 | 99.00 |
| MS COCO | Det | 99.61 | 99.50 |

94.85% / 96.80%, respectively, and the representative flow-matching model FLUX retains 98.99% / 100.00%. This suggests that watermark persistence is affected by more specific properties of the target generative process than model generation alone.

Further analysis in Appendix F.2 helps explain these differences. Although PrivateSeal accounts for VAE reconstruction during watermark optimization, different generative models use different VAEs, which may affect the embedded watermark differently. Our analysis suggests that SDXL's VAE reconstruction weakens the watermark more than FLUX's, even without denoising. This may reduce its tolerance to subsequent changes and partly explain why the watermark is less robust under SDXL-based regeneration.

Overall, these results suggest that the optimized perturbations are not merely overfitted to the SD1.5 proxy model, but transfer to a broader class of generative transformations.

**Cross-Dataset Robustness.** To evaluate whether PrivateSeal generalizes beyond the W-Bench data distribution, we conduct an additional experiment on MS COCO dataset [30]. In this experiment, the proxy model, optimization procedure, payload length, attack protocol, and verification metric are kept the same as in the W-Bench evaluation, with only the image dataset changed.

Since MS COCO does not provide the editing instructions or spatial masks required by the editing attacks in W-Bench, we restrict this evaluation to regeneration attacks, including both stochastic and deterministic variants. This experiment is therefore not intended as a comprehensive benchmark comparison against prior methods. Instead, it serves as a controlled test of robustness under data distribution shift. As shown in Table 4, PrivateSeal maintains high decoding accuracy on the MS COCO dataset, with only a small drop compared with W-Bench. This suggests that the watermark is not overly specialized to the W-Bench image distribution, and that the proposed optimization objective transfers across datasets.

### 4.5 Robustness under Conventional Image-Processing Attacks

We further evaluate PrivateSeal against conventional image-processing attacks, including JPEG compression, random black occlusion, crop-and-resize, rotation, and Gaussian blur. These experiments are conducted separately from the W-Bench attack pipeline using independently implemented image-processing transformations. We evaluate 100 watermarked images and report the mean bit accuracy and true positive rate (TPR) at 1% and 0.1% false positive rates (FPR).

Table 5: Robustness under conventional image-processing attacks. Results are evaluated on 100 watermarked images with a 30-bit payload. Bit Accuracy is averaged over all message bits, while TPR is computed at the image level.

| Attack | Severity | Bit Accuracy ↑ (%) | TPR@1% FPR ↑ (%) | TPR@0.1% FPR ↑ (%) |
|---|---|---|---|---|
| JPEG | $Q = 90$ | 99.87 | 100 | 100 |
| JPEG | $Q = 70$ | 99.87 | 100 | 100 |
| JPEG | $Q = 50$ | 99.77 | 100 | 100 |
| JPEG | $Q = 30$ | 98.37 | 99 | 99 |
| JPEG | $Q = 10$ | 85.60 | 87 | 83 |
| Black occlusion | 10% area | 99.93 | 100 | 100 |
| Black occlusion | 25% area | 97.93 | 100 | 100 |
| Black occlusion | 50% area | 87.30 | 99 | 97 |
| Crop + resize | 90% side retained | 51.47 | 2 | 1 |
| Crop + resize | 75% side retained | 51.33 | 4 | 1 |
| Crop + resize | 50% side retained | 49.30 | 1 | 1 |
| Rotation | $5^\circ$ | 53.67 | 3 | 0 |
| Rotation | $15^\circ$ | 49.10 | 2 | 2 |
| Rotation | $30^\circ$ | 49.50 | 1 | 1 |
| Rotation | $45^\circ$ | 51.30 | 4 | 1 |
| Gaussian blur | radius $= 1$ px | 100.00 | 100 | 100 |
| Gaussian blur | radius $= 2$ px | 98.47 | 100 | 100 |
| Gaussian blur | radius $= 3$ px | 91.20 | 100 | 98 |

As shown in Table 5, PrivateSeal tolerates substantial image degradation under attacks that preserve spatial alignment. JPEG compression ($Q \geq 30$) retains over 98% bit accuracy, while Gaussian blur with a 3-pixel radius yields 98% TPR at 0.1% FPR. Even under 50% black occlusion, bit accuracy remains at 87.30% and TPR reaches 97%. In contrast, crop-and-resize retaining 90% of the original side length reduces bit accuracy to 51.47% and TPR to only 1%. Rotation similarly reduces bit accuracy to approximately 50% and detection TPR to nearly zero. This contrast suggests that spatial desynchronization, rather than image degradation alone, is a major factor affecting watermark robustness. Geometric transformations remap image content to different latent coordinates, potentially disrupting alignment between the embedded watermark and fixed projection directions. Robustness to geometric desynchronization thus remains a limitation.

## 5 Conclusion

We present PrivateSeal, a user-side watermarking framework for existing images under diffusion-based editing and regeneration. PrivateSeal embeds messages by optimizing image-specific latent perturbations along low-sensitivity directions, while using a user-specific projection key for non-iterative verification. These results suggest that diffusion-resilient watermarking can be achieved without relying on a learned, user-specific verifier. By separating image-dependent watermark optimization from lightweight key-based verification, PrivateSeal shifts the computational burden toward embedding while keeping verification efficient.

**Limitations.** PrivateSeal shows strong but non-uniform transfer across target generative models. Although our analysis in Appendix F.2 suggests that differences in VAE margin preservation may partly explain this variation, the complete interaction between model-specific denoising dynamics and watermark decoding remains insufficiently understood. A more comprehensive characterization of these factors is left for future work. In addition, PrivateSeal relies on spatial alignment for reliable verification. Geometric transformations such as cropping and rotation can disrupt the correspondence between latent coordinates and fixed projection directions, substantially degrading watermark detection.

## Acknowledgments and Disclosure of Funding

Y. Chen and Z. Ji were supported by the National Natural Science Foundation of China (NSFC) under Grants 62441235 and 62676274.

## References

[1] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-First International Conference on Machine Learning*, 2024.

[2] 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. *arXiv preprint arXiv:2307.01952*, 2023.

[3] 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*, pages 10684–10695, 2022.

[4] Tim Brooks, Aleksander Holynski, and Alexei A Efros. InstructPix2Pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023.

[5] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

[6] Haozhe Zhao, Xiaojian Ma, Liang Chen, Shuzheng Si, Rujie Wu, Kaikai An, Peiyu Yu, Minjia Zhang, Qing Li, and Baobao Chang. UltraEdit: Instruction-based fine-grained image editing at scale. *Advances in Neural Information Processing Systems*, 37:3058–3093, 2024.

[7] Wenda Li, Huijie Zhang, and Qing Qu. Shallow Diffuse: Robust and invisible watermarking through low-dim subspaces in diffusion models. In *The Thirty-Ninth Annual Conference on Neural Information Processing Systems*, 2025.

[8] Shilin Lu, Zihan Zhou, Jiayou Lu, Yuanzhi Zhu, and Adams Wai-Kin Kong. Robust watermarking using generative priors against image editing: From benchmarking to advances. In *International Conference on Learning Representations*, 2025.

[9] 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.

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[11] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[12] Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-Ring watermarks: Fingerprints for diffusion images that are invisible and robust. *arXiv preprint arXiv:2305.20030*, 2023.

[13] Zijin Yang, Kai Zeng, Kejiang Chen, Han Fang, Weiming Zhang, and Nenghai Yu. Gaussian Shading: Provable performance-lossless image watermarking for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12162–12171, 2024.

[14] Nir Goren, Oren Katzir, Abhinav Nakarmi, Eyal Ronen, Mahmood Sharif, and Or Patashnik. Noiseprints: distortion-free watermarks for authorship in private diffusion models. In *International Conference on Learning Representations*, volume 2026, pages 99698–99727, 2026.

[15] Zhihao Li, Jiale Cai, Gezheng Xu, Hao Zheng, Qiuyue Li, Fan Zhou, Shichun Yang, Charles Ling, and Boyu Wang. Versatile transferable unlearnable example generator. *Advances in Neural Information Processing Systems*, 38:17495–17522, 2026.

[16] Zhihao Li, Gezheng Xu, Jiale Cai, Ruiyi Fang, Di Wu, Qicheng Lao, Charles Ling, and Boyu Wang. When priors backfire: On the vulnerability of unlearnable examples to pretraining. In *The Fourteenth International Conference on Learning Representations*, 2026.

[17] Jiale Cai, Gezheng Xu, Zhihao Li, Ruiyi Fang, Ruizhi Pu, Di Wu, Qicheng Lao, Charles Ling, and Boyu Wang. Fuse: Full-spectrum unlearnable examples via spectral equalization. In *Forty-third International Conference on Machine Learning*, 2026.

[18] Ali Al-Haj. Combined DWT-DCT digital image watermarking. *Journal of Computer Science*, 3(9):740–746, 2007.

[19] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In *Proceedings of the European Conference on Computer Vision*, pages 657–672, 2018.

[20] Matthew Tancik, Ben Mildenhall, and Ren Ng. StegaStamp: Invisible hyperlinks in physical photographs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2117–2126, 2020.

[21] Zhaoyang Jia, Han Fang, and Weiming Zhang. Mbrs: Enhancing robustness of dnn-based watermarking by mini-batch of real and simulated jpeg compression. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 41–49, 2021.

[22] Kevin Alex Zhang, Lei Xu, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Robust invisible video watermarking with attention. *arXiv preprint arXiv:1909.01285*, 2019.

[23] Fan Guo, Jiyu Kang, Qi Ming, Emily Davis, and Finn Carter. Vanishing watermarks: Diffusion-based image editing undermines robust invisible watermarking. *arXiv preprint arXiv:2602.20680*, 2026.

[24] 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*, pages 22466–22477, 2023.

[25] Tu Bui, Shruti Agarwal, and John Collomosse. TrustMark: Robust watermarking and watermark removal for arbitrary resolution images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18629–18639, 2025.

[26] Xuanyu Zhang, Runyi Li, Jiwen Yu, Youmin Xu, Weiqi Li, and Jian Zhang. EditGuard: Versatile image watermarking for tamper localization and copyright protection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11964–11974, 2024.

[27] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 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.

[28] Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. MagicBrush: A manually annotated dataset for instruction-guided image editing. *Advances in Neural Information Processing Systems*, 36:31428–31449, 2023.

[29] 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*, pages 3836–3847, 2023.

[30] 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*, pages 740–755. Springer, 2014.

[31] Rui Ma, Mengxi Guo, Yi Hou, Fan Yang, Yuan Li, Huizhu Jia, and Xiaodong Xie. Towards blind watermarking: Combining invertible and non-invertible mechanisms. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 1532–1542, 2022.

[32] Xiaoshuai Wu, Xin Liao, and Bo Ou. SepMark: Deep separable watermarking for unified source tracing and deepfake detection. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 1190–1201, 2023.

[33] Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, et al. Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025.

[34] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[35] Han Fang, Zhaoyang Jia, Zehua Ma, Ee-Chien Chang, and Weiming Zhang. PIMoG: An effective screen-shooting noise-layer simulation for deep-learning-based watermarking network. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 2267–2275, 2022.

[36] KA Navas, Mathews Cheriyan Ajay, M Lekshmi, Tampy S Archana, and M Sasikumar. DWT-DCT-SVD based watermarking. In *3rd International Conference on Communication Systems Software and Middleware and Workshops*, pages 271–274. IEEE, 2008.

# A Preliminaries

Latent Diffusion Models (LDMs) [3] perform diffusion in a compressed latent space rather than directly in the pixel space. Given an image $x$, a pretrained VAE encoder $\mathcal{E}$ [34] maps it to a latent representation

$$z_0 = \mathcal{E}(x), \tag{15}$$

and a corresponding decoder $\mathcal{D}$ maps latent variables back to the image space.

Following Denoising Diffusion Probabilistic Models (DDPMs) [10], the forward diffusion process gradually corrupts the clean latent $z_0$ with Gaussian noise. At timestep $t$, the noisy latent can be written in closed form as

$$z_t = \alpha_t z_0 + \sigma_t \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \mathbf{I}), \tag{16}$$

where $\alpha_t$ and $\sigma_t$ are determined by a predefined noise schedule.

The denoising network $\epsilon_\theta(z_t, t)$ is trained to predict the injected noise:

$$\mathcal{L}_{\text{LDM}} = \mathbb{E}_{z_0, \epsilon, t}\left[\|\epsilon - \epsilon_\theta(z_t, t)\|_2^2\right]. \tag{17}$$

During sampling, the reverse process iteratively updates the noisy latent using the denoising prediction. A generic discrete reverse update can be written as

$$z_{t-1} = \mu_t z_t - \nu_t \epsilon_\theta(z_t, t) + \tilde{\sigma}_t \epsilon', \qquad \epsilon' \sim \mathcal{N}(0, \mathbf{I}), \tag{18}$$

where $\mu_t$, $\nu_t$, and $\tilde{\sigma}_t$ are sampler-dependent coefficients derived from the noise schedule. For deterministic samplers, the stochastic term can be omitted by setting $\tilde{\sigma}_t = 0$ [11].

In our method, the pretrained denoising network is used only during watermark embedding as a proxy model for regularizing latent perturbations, while verification is non-iterative.

# B Derivation of the Stability Objective

In this section, we formally derive the stability objective $\mathcal{L}_{\text{sta}}$ from the dynamics of a watermark perturbation under the reverse diffusion process.

**Watermarked forward process.** Let $z_0 = \mathcal{E}(x)$ be the clean latent and $z_w = z_0 + \delta$ the watermarked latent. Following the standard forward diffusion parameterization, we have

$$z_t = \alpha_t z_0 + \sigma_t \epsilon, \qquad z_t^w = \alpha_t z_w + \sigma_t \epsilon = z_t + \alpha_t \delta, \quad \epsilon \sim \mathcal{N}(0, \mathbf{I}). \tag{19}$$

Define the time-dependent perturbation $\delta_t \triangleq \alpha_t \delta$, so that $z_t^w = z_t + \delta_t$.

**Reverse process.** We consider a generic reverse step of the form

$$z_{t-1} = \mu_t z_t - \nu_t \epsilon_\theta(z_t, t) + \sigma_t \epsilon', \quad \epsilon' \sim \mathcal{N}(0, \mathbf{I}), \tag{20}$$

where $\mu_t, \nu_t, \sigma_t$ are determined by the noise schedule (e.g., DDPM or DDIM). The watermarked counterpart evolves as

$$z_{t-1}^w = \mu_t(z_t + \delta_t) - \nu_t \epsilon_\theta(z_t + \delta_t, t) + \sigma_t \epsilon_w'. \tag{21}$$

**Perturbation dynamics.** Subtracting the clean trajectory yields

$$\delta_{t-1} = \mu_t \delta_t - \nu_t\big[\epsilon_\theta(z_t + \delta_t, t) - \epsilon_\theta(z_t, t)\big] + \sigma_t(\epsilon_w' - \epsilon'). \tag{22}$$

Taking the expectation over the independent stochastic terms $\epsilon', \epsilon_w'$, which are zero-mean, gives

$$\mathbb{E}[\delta_{t-1}] = \mu_t \delta_t - \nu_t\, \mathbb{E}\big[\epsilon_\theta(z_t + \delta_t, t) - \epsilon_\theta(z_t, t)\big]. \tag{23}$$

**Stability condition.** To prevent the watermark from being actively removed by the denoising network, we wish to make the score-prediction difference as small as possible:

$$\epsilon_\theta(z_t + \delta_t, t) \approx \epsilon_\theta(z_t, t). \tag{24}$$

Under this condition, the expected perturbation dynamics reduce to $\mathbb{E}[\delta_{t-1}] \approx \mu_t \delta_t$. For common schedules where $\mu_t = \alpha_{t-1}/\alpha_t$, this further simplifies to $\mathbb{E}[\delta_{t-1}] \approx \alpha_{t-1}\delta$, i.e., the perturbation is merely rescaled by the deterministic schedule and not suppressed by the score function.

**Resulting objective.** Minimizing the squared discrepancy between the score predictions directly leads to the stability loss used in the main text:

$$\mathcal{L}_{\mathrm{sta}} = \mathbb{E}_{t,\epsilon}\Big[\big\|\epsilon_\theta(z_t + \alpha_t\delta, t) - \epsilon_\theta(z_t, t)\big\|_2^2\Big], \tag{25}$$

where we have substituted back $\delta_t = \alpha_t\delta$. This objective encourages $\delta$ to lie in directions where the denoising network is locally insensitive, hence improving the persistence of the watermark across the reverse diffusion process.

# C Embedding and Verification Procedures

We provide pseudocode for the two core procedures of PrivateSeal: key-conditioned watermark embedding and projection-based verification. Algorithm 1 summarizes the embedding stage, where an image-specific latent perturbation is optimized under a fixed-energy constraint using the message, robustness, and fidelity objectives defined in the main text. Algorithm 2 summarizes the verification stage, which requires only one VAE encoding pass followed by projection with the corresponding user-specific key. The verification procedure is non-iterative and does not use reverse diffusion, inversion, or a key-specific neural detector.

**Algorithm 1** PrivateSeal Embedding

**Input:** Image $x$, message $w$, projection key $P$, models $\mathcal{E}, \mathcal{D}, \epsilon_\theta$, augmentation $\mathcal{A}$, hyperparameters
**Output:** Watermarked image $x_w$

$z_0 \leftarrow \mathcal{E}(x)$
Initialize $\delta_{\mathrm{raw}} \sim \mathcal{N}(0, \mathbf{I})$
**for** $i = 1$ **to** $N$ **do**
  $\delta \leftarrow \xi \cdot \delta_{\mathrm{raw}} / \|\delta_{\mathrm{raw}}\|_2$
  $z_w \leftarrow z_0 + \delta$
  Sample $t \sim \mathcal{U}(1, T)$ and $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  $z_t \leftarrow \alpha_t z_0 + \sigma_t \epsilon$
  Compute $\mathcal{L}_{\mathrm{message}}$, $\mathcal{L}_{\mathrm{robust}}$, and $\mathcal{L}_{\mathrm{fidelity}}$
  Compute $\mathcal{L}$ according to Eq. 13
  Update $\delta_{\mathrm{raw}}$ by gradient descent on $\mathcal{L}$
**end for**
$\delta \leftarrow \xi \cdot \delta_{\mathrm{raw}} / \|\delta_{\mathrm{raw}}\|_2$
**return** $x_w \leftarrow \mathcal{D}(z_0 + \delta)$

**Algorithm 2** PrivateSeal Verification

**Input:** Suspicious image $x'$, projection key $P$, public encoder $\mathcal{E}$
**Output:** Estimated message $\hat{w} \in \{-1, 1\}^L$

$\hat{z} \leftarrow \mathcal{E}(x')$
$\hat{w} \leftarrow \mathrm{sign}\big(P \cdot \mathrm{vec}(\hat{z})\big)$
**return** $\hat{w}$

# D Detailed Experimental Settings

## D.1 Datasets and preprocessing

All images are evaluated at $512 \times 512$ resolution. For W-Bench experiments, we follow the official W-Bench evaluation protocol. The benchmark samples images from the UltraEdit dataset, where each instance contains a source image, an editing prompt, and a region mask. For the evaluated image-to-image W-Bench subset, each regeneration and global-editing setting is evaluated on 1,000 images, while local editing is evaluated over five mask-size buckets, each containing 1,000 images. Images are converted to RGB, resized to $512 \times 512$, normalized to $[-1, 1]$, and encoded by the Stable Diffusion 1.5 VAE into a latent tensor of size $4 \times 64 \times 64$.

For cross-dataset evaluation, we use a held-out subset of MS COCO. The proxy model, embedding objective, payload length, projection-key construction, attack protocol, and verification metric are kept unchanged, with the image source being the only changed factor. Since MS COCO does not provide the editing instructions or spatial masks required by W-Bench editing attacks, we restrict the cross-dataset evaluation to stochastic and deterministic regeneration.

### D.2 Attack configurations

For the main W-Bench image-to-image comparison, we follow the official W-Bench configurations for the evaluated attack categories without modification. No attack-specific watermark optimization, attack-specific training, or per-attack threshold tuning is performed.

We evaluate three image-to-image generative transformation categories from W-Bench: image regeneration, global text-guided editing, and local editing/inpainting. For stochastic regeneration, W-Bench uses a VP noise scheduler and varies the noise timestep $t_s$ from 60 to 240 with an interval of 20. For deterministic regeneration, it uses DPM-Solver and evaluates the number of sampling steps $n_d \in \{15, 25, 35, 45\}$. For global editing, W-Bench evaluates Instruct-Pix2Pix, MagicBrush, and UltraEdit. Each model uses its default sampler with 50 sampling steps. The difficulty is controlled by the classifier-free guidance scale of the text prompt, which ranges from 5 to 9, while the image guidance scale is fixed to 1.5. For local editing, W-Bench evaluates ControlNet-Inpainting and UltraEdit with spatial masks. The difficulty is controlled by the edited-region ratio, with mask-size intervals of 10–20%, 20–30%, 30–40%, 40–50%, and 50–60%. For all local editing settings, the image guidance and text guidance are fixed to 1.5 and 7.5, respectively.

### D.3 PrivateSeal implementation

PrivateSeal uses the pretrained Stable Diffusion 1.5 VAE as the public encoder and decoder, and the pretrained Stable Diffusion 1.5 U-Net as the proxy denoising model during embedding. All pretrained model parameters are frozen. We use the `runwayml/stable-diffusion-v1-5` checkpoint from Diffusers. We use a DDIM scheduler for timestep sampling in the proxy denoising objective.

The payload length is fixed to $L = 30$ bits. Each projection key $P \in \mathbb{R}^{L \times D}$ has $D = 4 \cdot 64 \cdot 64 = 16384$ columns. We generate a projection key by sampling a random Gaussian matrix and orthonormalizing its rows. Unless otherwise specified, the same projection key is used across all experiments, including evaluated W-Bench attacks, cross-model evaluations, and cross-dataset evaluations. For each image, we independently sample a target message from $\{-1, +1\}^L$ and store the assigned message for evaluation.

For each image, message, and projection key, we optimize an image-specific latent perturbation. The perturbation energy is fixed to $\xi = 25$ by normalizing the unconstrained optimization variable at every iteration. We use Adam with learning rate $0.1$ for 500 optimization steps. At each step, the timestep is sampled uniformly from $[100, 600)$ and Gaussian noise is sampled independently for each image. Unless otherwise specified, the loss weights are $\lambda_{\text{robust}} = 20$, $\lambda_{\text{fidelity}} = \lambda_{\text{message}} = 1$.

The message target amplitude is set to $\xi/\sqrt{L}$ so that the fixed latent energy budget is evenly distributed across message dimensions. During optimization, we save intermediate perturbations. For each image, we select the checkpoint with the lowest proxy SD1.5 robustness loss among them.

### D.4 Verification and detection threshold

Verification is non-iterative. Given a suspicious image, we encode it once using the public VAE encoder and apply the corresponding projection key to the latent vector. The recovered message is obtained by taking the sign of the projection outputs.

For each attacked watermarked image, we compare the recovered message with the stored target message assigned to that image. Bit accuracy is computed as the fraction of matched bits over the 30-bit payload. For detection, we use this per-image bit accuracy as the detection score. For the held-out unwatermarked images used to estimate FPR, no watermark is embedded. We apply the same verification procedure under the same target-message hypotheses and compute the bit agreement between the decoded bits and the corresponding target messages. These bit-agreement scores form the negative score distribution for threshold selection. The decision threshold is selected so that the

empirical FPR on these 1,000 unwatermarked images does not exceed 0.1%, and the same threshold is then used for all evaluated attacks and target models.

### D.5 Visual quality metrics

We report PSNR, SSIM, and LPIPS between the original image and the watermarked image before attack. Metrics are computed at $512 \times 512$ resolution in RGB space. PSNR and SSIM are computed with data range $[0, 1]$, and LPIPS is computed using the AlexNet backbone. Results are averaged over the evaluated image set.

### D.6 Runtime and hardware

Runtime and memory are measured on a single NVIDIA GeForce RTX 4090 GPU with 24GB memory. Embedding time includes latent optimization and the final VAE decoding step. Verification time includes image preprocessing, one VAE encoding pass, and projection-based message decoding. Disk I/O and model loading are excluded. Runtime is measured with batch size 1 using synchronized wall-clock time, and peak GPU memory is measured using PyTorch peak allocated memory.

All experiments are implemented in PyTorch 2.10.0 with CUDA 12.8 and cuDNN 9.10.2, using HuggingFace Diffusers 0.31.0 and Transformers 4.44.2.

## E Detailed Results on W-Bench

This section provides the full W-Bench comparison over all evaluated baselines. The main paper reports a representative subset of methods for readability, while Table 6 includes the complete set of baseline results used in our evaluation. The evaluation protocol, attack configurations, visual-quality metrics, payload length, and detection thresholding procedure are the same as those described in Sec. 4.2 and Appendix D. Each attack cell reports Bit Accuracy and TPR@0.1% FPR, allowing both average bit-level recovery and low-false-positive detection performance to be compared across methods.

## F Further Mechanistic Analysis

### F.1 Revisiting the Stability Objective

The stability objective in PrivateSeal is designed to encourage watermark perturbations along latent directions with reduced denoiser sensitivity. While this provides an explicit optimization criterion, the resulting end-to-end robustness reflects the interplay of multiple factors throughout watermark embedding, generative transformation and verification. A more detailed analysis is therefore needed to disentangle their respective contributions and better understand the observed robustness.

During the author rebuttal, we conducted a preliminary component-wise ablation in response to the reviewers' requests under a limited time budget. Subsequent, more systematic experiments revealed that some of the initial trends were not consistently observed. More importantly, these investigations revealed a trade-off between denoiser sensitivity and decoding margin, providing a unified explanation for the ablation findings and cross-model variations.

We first examine whether the stability objective effectively reduces the denoiser's response to the optimized perturbations. To quantify this effect, we define the normalized denoiser sensitivity as

$$S(\delta) = \frac{1}{K} \sum_{k=1}^{K} \frac{\|\epsilon_\theta(z_{t_k} + \alpha_{t_k}\delta, t_k) - \epsilon_\theta(z_{t_k}, t_k)\|_2^2}{\|\delta\|_2^2}, \tag{26}$$

where $K = 8$ independent timestep–noise realizations are sampled separately from those used during optimization. Lower values indicate a smaller denoiser response per unit perturbation energy.

We evaluate this metric by varying $\lambda_{\text{sta}} \in \{0, 1, 3, 10\}$ across three payload lengths $L \in \{30, 50, 70\}$, while keeping the other optimization settings fixed.

As shown in Table 7, increasing the stability weight effectively reduces the normalized denoiser response. These results confirm that the stability objective achieves its intended effect of encouraging

Table 6: Detailed W-Bench comparison over all evaluated baselines. Each cell reports Bit Accuracy (%) / TPR@0.1% FPR (%). The best value is in **bold**, and the second-best is underlined.

| Method | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Bit Accuracy ↑ (%) TPR@0.1%FPR ↑ (%) | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Regeneration | | Global Editing | | | Local Editing | |
| | | | | Sto | Det | Pix2Pix | Ultra | Magic | Ultra | CtrlN |
| MBRS [21] | 27.37 | 0.8940 | 0.1877 | 98.72<br>99.53 | 99.46<br>99.35 | 95.56<br>83.50 | 65.97<br>7.50 | 95.12<br>88.54 | 95.40<br>99.60 | 98.97<br>89.16 |
| CIN [31] | **43.19** | 0.9847 | 0.0270 | 76.34<br>44.85 | 77.83<br>51.65 | 79.81<br>51.40 | 62.79<br>17.00 | 82.91<br>68.38 | 85.13<br>51.28 | 78.09<br>66.04 |
| PiMoG [35] | 37.72 | 0.9863 | 0.0289 | 85.48<br>82.85 | 85.54<br>71.18 | 88.10<br>72.78 | 73.57<br>40.14 | 91.31<br>81.88 | 85.07<br>74.30 | 87.77<br>64.22 |
| RivaGAN [22] | 40.43 | 0.9702 | 0.0488 | 61.63<br>10.12 | 65.89<br>12.50 | 62.99<br>6.22 | 51.54<br>4.14 | 71.24<br>33.96 | 77.67<br>34.28 | 70.44<br>56.92 |
| SepMark [32] | 35.48 | 0.9814 | 0.0150 | 81.81<br>61.21 | 90.20<br>73.85 | 93.53<br>87.74 | 78.77<br>51.84 | 91.31<br>82.58 | 95.90<br>92.94 | 93.10<br>97.14 |
| DWTDCT [18] | 40.46 | 0.9705 | 0.0136 | 48.98<br>0.09 | 48.82<br>0.00 | 49.49<br>0.04 | 49.12<br>0.06 | 48.92<br>0.04 | 49.16<br>0.32 | 48.43<br>0.56 |
| DWTDCTSVD [36] | 40.40 | 0.9799 | 0.0265 | 53.27<br>3.12 | 47.68<br>1.43 | 50.41<br>3.82 | 51.33<br>4.02 | 65.19<br>30.84 | 74.42<br>24.56 | 64.23<br>50.04 |
| SSL [24] | 41.77 | 0.9796 | 0.0350 | 57.55<br>1.76 | 65.65<br>9.70 | 64.17<br>25.06 | 56.06<br>10.58 | 73.19<br>50.10 | 68.44<br>25.28 | 69.34<br>31.46 |
| StegaStamp [20] | 29.65 | 0.9107 | 0.0645 | 79.04<br>91.09 | 88.05<br>92.13 | 90.36<br>93.72 | 70.08<br>51.24 | 91.28<br>91.18 | 92.54<br>98.84 | 88.74<br>99.06 |
| TrustMark [25] | 41.27 | 0.9910 | **0.0026** | 57.00<br>9.22 | 67.26<br>34.20 | 80.23<br>77.72 | 66.21<br>43.48 | 84.52<br>85.90 | 73.43<br>76.62 | 78.59<br>59.78 |
| EditGuard [26] | 37.58 | 0.9406 | 0.0171 | 49.58<br>0.09 | 50.49<br>6.00 | 49.45<br>0.06 | 52.17<br>1.16 | 48.74<br>0.24 | 53.79<br>0.18 | 49.23<br>2.66 |
| VINE-Robust [8] | 37.34 | **0.9934** | 0.0063 | 88.59<br>99.66 | 95.43<br>99.98 | 92.04<br>97.46 | **84.02**<br>**86.86** | 91.83<br>**94.58** | 94.42<br>**99.96** | 88.78<br>93.04 |
| **PrivateSeal (Ours)** | 36.57 | 0.9329 | 0.0582 | **99.95**<br>**100.00** | **99.96**<br>**100.00** | **97.29**<br>**97.59** | 79.18<br>56.98 | **95.77**<br>92.34 | **97.81**<br>99.16 | **99.14**<br>**99.12** |

Table 7: Normalized denoiser response under different stability weights. Lower values indicate reduced denoiser sensitivity.

| Payload | $\lambda_{\text{sta}} = 0$ | $\lambda_{\text{sta}} = 1$ | $\lambda_{\text{sta}} = 3$ | $\lambda_{\text{sta}} = 10$ |
|---|---|---|---|---|
| 30 bits | 0.9174 | 0.9162 | 0.9076 | 0.8646 |
| 50 bits | 0.7563 | 0.7489 | 0.7360 | 0.6888 |
| 70 bits | 0.6666 | 0.6626 | 0.6471 | 0.5964 |

low-sensitivity latent perturbations. However, as shown in Table 8, the reduction in denoiser response does not consistently translate into higher post-attack bit accuracy.

These follow-up experiments provide a more systematic assessment than the preliminary ablation conducted during the author rebuttal, revealing that reduced denoiser sensitivity does not necessarily translate into improved robustness and suggesting that an additional factor influences the final decoding outcome.

To understand this behavior, we examine how much decoding margin the watermark retains before and after an attack. Since PrivateSeal decodes each message bit by the sign of its latent projection, the zero-projection boundary determines whether that bit is correctly recovered. For the $i$-th bit, we define the clean signed margin as

$$m_i = w_i p_i^\top \operatorname{vec}(\mathcal{E}(x_w)), \tag{27}$$

where $w_i \in \{-1, 1\}$ is the target bit and $p_i^\top$ is the corresponding row of the projection key. A larger positive margin means that a larger change is needed to flip the decoded bit.

Table 8: Post-attack bit accuracy (%) under different stability weights. Results are averaged across the evaluated attack strengths.

| Payload | Attack | $\lambda_{\text{sta}} = 0$ | $\lambda_{\text{sta}} = 1$ | $\lambda_{\text{sta}} = 3$ | $\lambda_{\text{sta}} = 10$ |
|---|---|---|---|---|---|
| 30 bits | Deterministic | 99.27 | 99.27 | 99.38 | **99.48** |
| | Stochastic | 99.00 | **99.08** | 99.00 | **99.08** |
| 50 bits | Deterministic | 95.94 | **96.19** | 96.13 | 95.94 |
| | Stochastic | **96.23** | 95.70 | 96.20 | 95.98 |
| 70 bits | Deterministic | 89.38 | **90.13** | 89.06 | 88.35 |
| | Stochastic | 91.05 | **91.29** | 90.61 | 90.13 |

An attack $T$ changes this margin by

$$d_i = w_i p_i^\top \operatorname{vec}\left(\mathcal{E}(T(x_w)) - \mathcal{E}(x_w)\right). \tag{28}$$

The bit remains correctly decoded as long as $m_i + d_i > 0$. Thus, robustness depends on two complementary quantities: the initial margin available before the attack and the harmful displacement introduced by the attack.

We next examine how stability regularization affects the clean decoding margin. Table 9 reports the average margin under four stability weights, with all other optimization settings held fixed.

Table 9: Effect of stability regularization on clean decoding margin. Mean signed decoding margins measured after VAE decoding and re-encoding, averaged over all message bits.

| Payload | $\lambda_{\text{sta}} = 0$ | $\lambda_{\text{sta}} = 1$ | $\lambda_{\text{sta}} = 3$ | $\lambda_{\text{sta}} = 10$ |
|---|---|---|---|---|
| 30 bits | 1.6349 | 1.6339 | 1.6281 | 1.6174 |
| 50 bits | 1.0934 | 1.0879 | 1.0747 | 1.0573 |
| 70 bits | 0.7720 | 0.7554 | 0.7536 | 0.7363 |

As shown in Table 9, the average clean decoding margin consistently decreases as the stability weight increases, with a more pronounced relative reduction at higher payloads. This observation offers a possible explanation for why lower denoiser sensitivity does not necessarily lead to higher post-attack bit accuracy. Although the stability objective effectively reduces the denoiser response, stronger regularization can also reduce the margin available to tolerate attack-induced changes.

Overall, these experiments confirm the intended effect of the stability objective on denoiser sensitivity, while showing that its impact on final decoding accuracy is more complex. Successful watermark recovery ultimately depends on the retained decoding margin after generative transformations, rather than denoiser sensitivity alone. This perspective also provides a basis for understanding the different robustness levels observed across generative models, which we investigate next.

### F.2 Understanding Cross-Model Variations

As shown in Table 3, PrivateSeal exhibits different levels of robustness across generative models. In particular, SDXL causes a noticeable degradation in watermark recovery, whereas FLUX retains nearly perfect bit accuracy. To better understand this difference, we examine two complementary factors: the denoiser's response to the embedded perturbation and the decoding margin retained after the target model's VAE round trip.

We first examine whether the low-sensitivity property learned using the SD1.5 proxy transfers to other generative models. Using 100 images, we compare the directional response of the watermark perturbation with that of 128 equal-energy random directions within each model's latent space.

As shown in Table 10, the watermark perturbations exhibit lower directional responses than equal-energy random directions across all three models, suggesting that the low-response property also extends to the held-out generative models.

However, SDXL exhibits a lower relative response than FLUX, despite having substantially worse regeneration robustness (Table 3). This suggests that the model-relative denoiser response alone is insufficient to explain the observed performance differences.

Table 10: Relative directional response across generative models. The response along the watermark perturbation is expressed as a percentage of the mean response over 128 equal-energy random directions within each model, evaluated on 100 images.

| Model | Relative response (%) |
|---|---|
| SD1.5 | 56.68 |
| SDXL | 43.57 |
| FLUX | 77.14 |

We therefore examine how the target VAE affects the watermark before denoising. Specifically, we perform a deterministic VAE-only round trip using each model's encoder and decoder, and measure the signed decoding margin using the public PrivateSeal verifier. The margin-retention rate is defined as the ratio of the mean signed margin after the round trip to that before the round trip.

Table 11: VAE margin retention and cross-model regeneration robustness. Margin retention is measured after a deterministic VAE-only round trip on 100 watermarked images. Regeneration bit accuracy is taken from Table 3.

| Model | Margin retention (%) | Regeneration Bit Accuracy (%) |
|---|---|---|
| SD1.5 | 97.2 | 99.98 |
| SDXL | 78.6 | 87.23 |
| FLUX | 98.3 | 98.99 |

As shown in Table 11, the decoding-margin retention differs substantially across three models. SDXL preserves only 78.6% of the original margin, compared with 97.2% for SD1.5 and 98.3% for FLUX. This indicates that the SDXL VAE causes substantially greater degradation of the watermark's decoding margin, leaving less tolerance for subsequent denoising-induced perturbations.

These observations offer a possible explanation for the weaker robustness under SDXL despite its favorable relative directional response. Together with the preceding ablation analysis, they suggest that denoiser sensitivity alone cannot fully characterize watermark robustness: the preservation of decoding margin throughout the generative transformation pipeline is also important.

## G Key Selectivity Analysis

We analyze the behavior of PrivateSeal under an independent wrong key. Let $z$ be any fixed latent representation and let $P' \in \mathbb{R}^{m \times D}$ be an independently sampled random projection key with orthonormal rows. Since $P'$ is independent of $z$, the sign vector $\mathrm{sign}(P' \cdot \mathrm{vec}(z))$ behaves as a random binary string over $\{-1, 1\}^m$.

Therefore, for any target message $w \in \{-1, 1\}^m$, the probability of exact message agreement under an independent wrong key is

$$\Pr[\mathrm{sign}(P' \cdot \mathrm{vec}(z)) = w] = 2^{-m}.$$

More generally, if verification accepts when at least $r$ out of $m$ bits match, the random acceptance probability is

$$\Pr[\mathrm{match} \geq r] = \sum_{k=r}^{m} \binom{m}{k} 2^{-m}.$$

For the 30-bit payload used in our experiments, exact random agreement is $2^{-30}$, and accepting at least 24 matching bits gives a random acceptance probability of approximately $0.0715\%$. This analysis supports the use of independent projection keys for scalable multi-user deployment, while not constituting a cryptographic security proof.

# H Security Analysis

## H.1 Partial Key Leakage

We consider a partial key leakage scenario where an attacker knows $q$ out of the $L$ projection rows of a user-specific key $P$. We assume that the attacker can correctly match the $q$ corresponding message bits, while each of the remaining $L - q$ bits is independently guessed with probability $1/2$.

Under this model, the number of matched bits follows

$$M_q = q + \text{Binomial}\left(L - q, \frac{1}{2}\right), \tag{29}$$

and the expected bit accuracy is

$$\mathbb{E}[\text{BA} \mid q] = \frac{1}{2} + \frac{q}{2L}. \tag{30}$$

If verification accepts an image when at least $r$ bits match, the acceptance probability is

$$P_{\text{accept}}(q) = \Pr[M_q \geq r]. \tag{31}$$

Table 12: Theoretical effect of partial key leakage for $L = 30$, with an acceptance threshold of 24 matching bits. The probabilities are analytically calculated under the independent-guessing assumption.

| Leaked rows ($q$) | Expected BA (%) | Acceptance probability (%) |
|---|---|---|
| 0 | 50.00 | 0.0715 |
| 5 | 58.33 | 0.7317 |
| 10 | 66.67 | 5.766 |
| 15 | 75.00 | 30.36 |
| 20 | 83.33 | 82.81 |
| 24 | 90.00 | 100.00 |

As shown in Table 12, partial key leakage progressively increases the probability of passing verification. This analysis characterizes a simplified leakage scenario rather than providing a cryptographic security guarantee against adaptive attacks.

If a projection key is fully compromised, its corresponding verification rule can no longer be considered secure. However, because PrivateSeal decouples the projection key from the public VAE and verifier architecture, a compromised key can be revoked and replaced with a newly sampled orthonormal key without retraining or modifying any neural-network component. Such key rotation applies to future watermark embedding and does not retroactively protect images embedded under the compromised key.

## H.2 Copy Attacks and Exploratory Content-Binding Extension

We further investigate donor-based copy attacks, in which an attacker transfers watermark signals from a watermarked image to unrelated content without knowing the private projection key.

We first evaluate the original latent-only verifier under this attack on 100 images. The resulting copied forgeries achieve an average watermark bit accuracy of 95.67%. We then treat legitimately diffusion-regenerated watermarked images as positive samples and copy-forged images as negative samples, using watermark bit accuracy as the confidence score. Under this evaluation protocol, the original verifier achieves 0% TPR at 1% FPR. This indicates that the latent-only verifier cannot reliably distinguish legitimate diffusion regeneration from watermark transplantation.

This limitation reflects a fundamental distinction between watermark robustness and forgery resistance. Consider a blind verifier $V_K(y) \in \{0, 1\}$ that observes only the final image $y$ and a fixed secret key $K$. For a legitimate diffusion transformation,

$$A^w \xrightarrow{T} B^w, \tag{32}$$

the verifier is expected to accept the transformed image, $V_K(B^w) = 1$, even when its visual content has changed substantially from $A$ to $B$.

An attacker, however, may construct

$$(A^w, B) \xrightarrow{\text{copy}} B^{\text{forge}}, \tag{33}$$

which preserves the content of $B$ while transplanting the watermark response from $A^w$. If the legitimate transformation and the forgery produce similar representations from the verifier's perspective, the verifier may be unable to distinguish their different origins based on the final image alone.

To explore a possible mitigation, we additionally investigate an optional content-binding extension. This extension does not modify the original 30-bit embedding procedure, rerun watermark optimization, or introduce additional perturbations into the released watermarked images. Instead, it introduces a registration-dependent auxiliary verification key.

The extension associates the latent-domain watermark response with a keyed, content-dependent fingerprint extracted from the current image in the pixel domain. Specifically, we construct an $m$-bit check vector

$$h_m(x) \in \{-1, +1\}^m \tag{34}$$

using keyed multi-scale comparisons of local luminance statistics. The secret key determines the compared patch locations, spatial scales, pairings, and comparison signs.

We then construct an auxiliary projection matrix $Q \in \mathbb{R}^{m \times D}$ such that, for each registered watermarked image $x_w$,

$$\text{sign}\left(Q \cdot \text{vec}(\mathcal{E}(x_w))\right) = h_m(x_w), \tag{35}$$

where $\mathcal{E}$ denotes the public VAE encoder. The original projection matrix $P$ and the auxiliary matrix $Q$ jointly serve as the secret verification parameters. At verification time, the auxiliary latent readout is checked against the content-dependent fingerprint of the inspected image.

Since the latent dimensionality is high, a single auxiliary matrix can satisfy the consistency constraints of multiple registered images when $m$ is moderate. In our experiment, we use a 70-bit consistency fingerprint for 100 registered images sharing the same auxiliary matrix $Q$. Test-time verification remains blind, requiring neither the original images nor per-image fingerprint records.

With this extension, the average verification score of copied forgeries decreases from 95.67% to 66.09%. Under the same binary evaluation protocol, with legitimate diffusion transformations as positive samples and copied forgeries as negative samples, the extended verifier achieves 99% TPR at 1% FPR.

We emphasize that this content-binding mechanism is an exploratory defense extension rather than a property of the original latent-only verifier. Its effectiveness against broader and adaptive forgery attacks remains to be investigated.

# I Additional Discussions

## I.1 Broader Impacts

PrivateSeal is intended to support ownership attribution, provenance tracking, and accountability for user-side images that may later undergo diffusion-based editing or regeneration. By enabling non-iterative verification and user-specific key assignment, such systems may help creators, platforms, and institutions preserve ownership signals after generative transformations.

At the same time, robust user-specific watermarking also introduces potential risks. If deployed without appropriate safeguards, per-user watermarks could be used for user tracking, surveillance, or unauthorized monitoring. Incorrect detections, false positives, or compromised keys may also lead to disputed or overconfident ownership claims. In addition, watermarking systems should not be treated as sole evidence of authorship or ownership. Practical deployments should therefore include calibrated detection thresholds, uncertainty reporting, secure key management, access control, audit mechanisms, and appeal procedures for contested detections.

### I.2 Existing Assets and Licenses

Our experiments use existing datasets, benchmarks, pretrained generative models, baseline implementations, and standard open-source software packages. We cite the corresponding papers and repositories in the main text and summarize the main assets below.

**Datasets and benchmarks.** We evaluate on W-Bench [8] and MS COCO [30]. W-Bench contains images and editing metadata sourced from existing datasets such as COCO, Flickr, and ShareGPT4V, and we use it only for research evaluation under its stated terms of use. For MS COCO, we follow the dataset terms, including the Flickr terms for images and the CC BY 4.0 license for annotations.

**Pretrained generative models.** We use Stable Diffusion 1.5 as the proxy denoising model for watermark optimization. For cross-model evaluation and generative transformations, we use publicly available checkpoints from the Stable Diffusion family, including SD2.1, SDXL, SD3, and SD3.5, as well as FLUX. We use these models only for research evaluation and follow their respective licenses and model cards, including CreativeML OpenRAIL/OpenRAIL++ licenses, Stability AI research or community licenses, and the Apache-2.0 license for FLUX.1-schnell where applicable.

**Editing models and baselines.** For W-Bench attacks and baseline comparisons, we follow the official W-Bench evaluation protocol and cite the corresponding methods, including InstructPix2Pix, UltraEdit, MagicBrush, ControlNet-Inpainting, VINE, Shallow Diffuse, and other baseline watermarking methods reported in the benchmark. When using released checkpoints, code, or evaluation outputs, we follow the corresponding repository licenses and terms of use.

**Software packages.** Our implementation builds on standard open-source Python and deep-learning packages, including PyTorch, torchvision, diffusers, OpenCV, Kornia, LPIPS, TorchMetrics, NumPy, pandas, and Matplotlib. We use these packages under their respective open-source licenses, such as BSD-style, Apache-2.0, and BSD-2-Clause licenses.

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the paper's contributions and scope: user-side watermarking for existing images under diffusion-based editing and regeneration, with low-cost verification and low-cost user-specific key assignment. The main limitation is discussed in Section 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the main limitation of PrivateSeal in Section 5: although PrivateSeal shows strong transfer across different target generative models, the transfer is not uniform, and the current work does not yet fully explain which properties of the target generative process govern watermark persistence.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A].

   Justification: The paper does not present formal theoretical results in the form of theorems or lemmas with complete proofs. However, we provide intuitive mathematical derivations to motivate the proposed method.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes].

   Justification: We disclose the information needed to reproduce the main experimental results to the extent that it affects the paper's claims and conclusions. Specifically, the main paper and Appendix D describe the embedding and verification algorithms, datasets and preprocessing, evaluated W-Bench image-to-image attack configurations, cross-dataset protocol, pretrained checkpoints, projection-key construction, payload length, optimization hyperparameters, detection-threshold calibration, visual-quality metrics, and hardware/software environment.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No].

   Justification: At the time of anonymous submission, we do not provide a complete public code package. However, the paper and appendix describe the proposed algorithm, model checkpoints, data sources, attack protocol, hyperparameters, evaluation metrics, threshold calibration, and compute environment needed to reproduce the main results.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes].

   Justification: We specify the experimental settings necessary to understand the results, including image resolution, preprocessing, dataset usage, attack parameters, payload length, projection-key generation, optimizer, learning rate, number of optimization steps, timestep sampling range, perturbation budget, loss weights, message target amplitude, checkpoint selection rule, verification procedure, detection score, FPR threshold selection, and visual-quality metrics. Full details are provided in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No].

   Justification: We do not report formal error bars or confidence intervals. However, the main W-Bench results are averaged over 1,000 watermarked images per attack, and the evaluation protocol, threshold calibration, and sample sizes are fully specified.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes].

   Justification: We report the compute environment and resource usage in the paper, including GPU type, image resolution, batch size, embedding time, verification time, and peak GPU memory. We also specify the number of images and attacks used in each evaluation, which allows estimating the total compute required to reproduce the experiments. Full details are provided in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

Justification: We have reviewed the NeurIPS Code of Ethics and the research conforms to it. The work does not involve human-subject studies or crowdsourcing. We use existing datasets, models, and baselines according to their terms, and we discuss potential societal impacts and limitations of watermark-based ownership verification.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes].

Justification: The paper discusses potential positive impacts of robust user-specific watermarking for ownership attribution, content provenance, and accountability under diffusion-based editing and regeneration. It also discusses potential negative impacts, including privacy and tracking concerns in per-user watermarking systems, possible over-reliance on watermark detection for ownership claims, harms from false positives or compromised keys, and misuse for unauthorized attribution or monitoring. We further mention mitigation strategies such as calibrated thresholds, reporting uncertainty, secure key management, access control, and avoiding the use of watermark detection as sole evidence.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A].

Justification: The paper does not release pretrained generative models, high-risk datasets, scraped datasets, or other assets that require controlled release safeguards. The work evaluates a watermarking method using existing public models and benchmarks.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes].

    Justification: We cite the datasets, benchmarks, pretrained models, baseline implementations, and software packages used in our experiments. We also summarize the corresponding licenses and terms of use in Appendix I.2, including W-Bench, MS COCO, Stable Diffusion-family checkpoints, FLUX, editing models, watermarking baselines, and standard open-source dependencies. All assets are used for research evaluation only and according to their respective licenses and terms.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A].

    Justification: The paper does not introduce or release a new dataset, pretrained model, or other standalone asset.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A].

    Justification: The paper does not involve crowdsourcing, user studies, human-subject experiments, or newly collected human annotations. We do not recruit participants, collect responses, provide instructions to human annotators, or compensate workers. The experiments are algorithmic evaluations on existing datasets and benchmarks, with performance measured using automatic metrics such as bit accuracy and TPR@0.1% FPR.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A].

    Justification: The research does not involve interaction with human participants, intervention on human subjects, collection of private personal data, or newly collected human-subject annotations. The experiments use existing public datasets, pretrained models, and benchmark protocols for algorithmic evaluation. Therefore, IRB approval or an equivalent institutional review is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A].

    Justification: LLMs are not used as an important, original, or non-standard component of the proposed method, experimental pipeline, or evaluation protocol. The core method consists of latent-space perturbation optimization, projection-key-based verification, and diffusion-model-based robustness regularization. Any use of language or coding assistance, if any, was limited to writing, editing, formatting, or routine implementation support and did not affect the core methodology, scientific claims, or originality of the research.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.