SEAL: Semantic Aware Image Watermarking

Supplementary Material

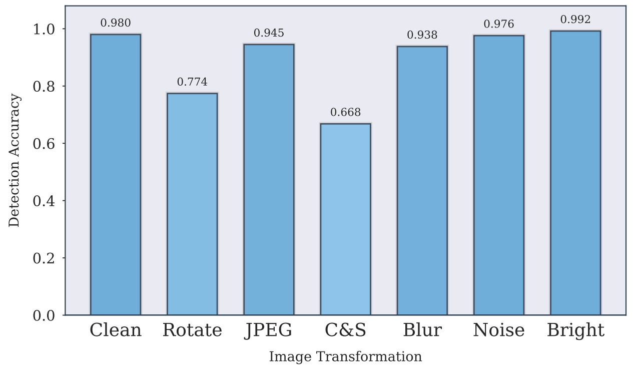

Figure 6. ***Robustness of Watermark Detection Against Image Transformations.*** *Comparison of correct watermark detection accuracy of* SEAL *under various image transformations.*

7. Additional Related Works

Post-Processing Methods. Post-processing techniques embed watermarks after the image generation stage, providing model-agnostic flexibility at the cost of potential quality degradation. Frequency-domain methods, such as methods using the Discrete Wavelet Transform (DWT) and Discrete Cosine Transform (DCT) [1, 23], embed watermarks in the transformed domains and offer robustness against operations like resizing and translation. Complementing these, deep encoder-decoder frameworks such as HiDDeN [37] and StegaStamp [30] utilize end-to-end neural training for watermark embedding and extraction. Despite these advancements, however, these methods are vulnerable to regeneration attacks [36]. Alternative strategies operating in latent spaces have also been proposed [11], though they also remain susceptible to sophisticated removal attacks.

8. Proof of Lemma 3.2

Proof of Lemma 3.2. The angle between the original semantic vector $\mathbf{v}$ used to generate the watermark and extracted semantic vector $\tilde{\mathbf{v}}$ of the suspect image is

$$\theta(\mathbf{v},\tilde{\mathbf{v}}) = \cos^{-1}\left(\frac{\langle \mathbf{v},\tilde{\mathbf{v}}\rangle}{\|\mathbf{v}\|_2\|\tilde{\mathbf{v}}\|_2}\right) \in [-90^\circ, 90^\circ].$$

By the property of SimHash[1] and Assumption 3.1, the probability[2] that the ith patch aligns is

$$\rho(\theta) := \Pr(\|\mathbf{z}_i - \tilde{\mathbf{z}}_i\|_2 \leq \tau) = \left(1 - \frac{\theta}{180^\circ}\right)^b.$$

Since each SimHash instance is independent, the number of matches m is distributed like a Binomial with n trials and success probability $\rho(\theta)$. During watermark detection, we count the number of patches that match, declaring an image watermarked if the number of matches exceeds the threshold m^{match}. Setting $m^{\text{match}} = \lfloor n\rho(\theta^{\text{mid}})\rfloor$ yields the lemma statement. $\square$

9. Implementation Details

9.1. Key Parameters

Unless otherwise stated, the results are reported with the following parameters: number of patch matching threshold $n_{match} = 12$; patch-wise matching threshold $\tau = 2.3$; number of projection per noise patch: $b = 7$; number of noise patches $k = 1024$. All parameters were chosen to optimize the overall performance.

9.2. Spatial Test

To better analyze potential image tampering, we examine the structural organization of high-intensity regions in the patch correspondence heatmaps (see Section 3, Tampering Detection). Specifically, we threshold the heatmap data at the 80th percentile and identify connected components. The extracted parameter, the number of distinct clusters detected at this threshold, provides insight into the fragmentation of high-intensity regions. A higher number of clusters indicates a more dispersed distribution, while a lower number suggests more contiguous structures, which may be indicative of image tampering.

9.3. Transformations for the Removal Attack

We use a standard suit of transformations, including a 75° rotation, 25% JPEG compression, 75% random cropping and scaling (C & S), Gaussian blur using an 8×8 filter, Gaussian noise with $\sigma = 0.1$, and color jitter with a brightness factor uniformly sampled between 0 and 6.

9.4. Embedding Model Fine Tuning Process

Source Prompts and Caption Pairing. Prompts were sampled from MS-COCO and the Stable Diffusion Prompt

[1]See Section 3 of [7] for details on why

$$\Pr_{\mathbf{r}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}(\text{sign}(\langle \mathbf{v},\mathbf{r}\rangle) = \text{sign}(\langle \tilde{\mathbf{v}},\mathbf{r}\rangle)) = 1 - \frac{\theta}{180^\circ}. \quad (3)$$

[2]Technically, there is an additional chance of a random collision but, given the size of modern cryptographic hash functions like SHA2, we assume this probability is negligible.

Table 3. ***Robustness of Steganalysis-Based Removal.*** *Comparison of performance metrics (ROC-AUC) under various levels of averaging.*

Method	5	10	20	50	100	200	500	1000	2000	5000
Tree-Ring (AUC)	0.293	0.267	0.314	0.275	0.214	0.228	0.211	0.224	0.224	0.241
WIND (AUC)	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
SEAL (AUC)	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**

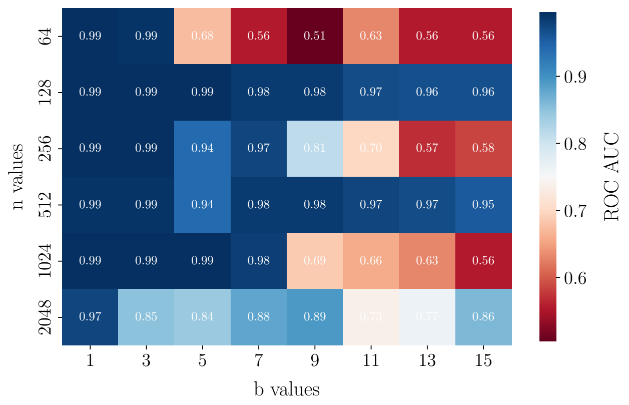

Figure 7. ***Ablation Study of the Number of Patches (n) and Bits (b) on Watermark Detection Performance.***

Dataset (`Gustavosta/Stable-Diffusion-Prompts`). Images were generated using Stable Diffusion v2.1 (`stabilityai/stable-diffusion-2-1-base`), then captioned with BLIP-2 (`Salesforce/blip2-flan-t5-xl`). A regeneration loop used each caption as a prompt to generate a second image, which was again captioned, yielding semantically aligned $(\text{caption}_1, \text{caption}_2)$ pairs. Unrelated pairs were constructed by randomly mismatching captions. This procedure yielded 10,000 caption pairs.

Fine-Tuning. A SentenceTransformer model (`paraphrase-Mpnet-base-v2`) was fine-tuned using `MultipleNegativesRankingLoss`. Training was performed for 140 epochs with a batch size of 64 and 10% warmup steps, using the AdamW optimizer with default settings.

We provide the full implementation details of the fine-tuning process to support reproducibility and enable further research.[3].

[3]Our fine-tuned caption embedding model is publicly available at https://huggingface.co/kasraarabi/finetuned-caption-embedding

10. Ablation of Number of Patches and Bits

To investigate the impact of the number of patches (n) and the number of bits (b) used to generate the initial noise, we conducted an exhaustive ablation study across various parameter combinations. The results are presented in Figure 7.

11. Resilience to Latent Forgery Attacks

We evaluate SEAL under the Latent Forgery Attack [17]. This attack aims to adversarially perturb non-watermarked images such that they appear watermarked by mimicking the latent representation of an originally watermarked image. This type of attacks assumes access to at least one watermarked image and attempts to shift unrelated images into the watermarked image latent region [22].
Our experiments, conducted on 100 images, demonstrated complete robustness against such attacks. Due to its semantic binding, our watermark is closely entangled with the high-level content of the original image (used by the attacker). Therefore, it is unlikely that the watermark can be transferred to images featuring unrelated content. Beyond this, the task of forging the latent patches is itself highly nontrivial. Yet, aiming to forge noise patches that better align semantically with the original watermarked image (used during the attack) might yield greater success. We leave this direction for future research.

12. Additional Limitations and Discussion

Distortion-Free Property for Sets of Images Our watermarking scheme securely generates the noise for each patch from a normal distribution, ensuring that each individual noise is distributed from a normal distribution. However, multiple watermarked images corresponding to related prompts may leak information about the noise i.e., the noise in some patches will match while the noise in other patches does not. This leakage arises from our design choice to make similar prompts produce similar watermarks, a feature that enhances consistency but comes at the cost of some information exposure.
In contrast, some prior works do not exhibit this property and instead maintain a stronger sense of distribution-free randomness. Ignoring cases where the exact same noise is reused, such as when multiple images are generated by the

same user in [34], these methods ensure that each image is independently sampled from a normal distribution. This fundamental difference highlights a trade-off between ensuring independent randomness and enabling structured watermark consistency across related prompts. A user concerned about the distortion-free property for sets may vary the secret salt for different generations. This will allow the user to enjoy the best of both worlds, At the cost of searching through possible salts that may have been used during detection time.

Further Possible Improvement. We made an initial attempt to find a semantic vector that is both known before generation and recoverable from the generated image. Yet, we believe this is a promising direction for future research. Improved semantic embedding methods, as well as approaches that jointly optimize image generation and semantic descriptor generation, could enhance the correspondence between the embedded watermark and the image's semantics. Such advancements may enable much stricter bounds on detecting when a watermarked image has been tampered with and how.

Watermarking Without a Proxy Image. To watermark an image directly with SimHash, we may embed the noise via post hoc diffusion inpainting (see Section 4.3 of [4]), at the expense of a modest quality decrease; SSIM = 0.768. Alternatively, one may optimize an embedding of the prompt to correlate well between a givan prompt and the caption of the resulting image (similarly to Figure 5).

Reliance on DDIM inversion. While diffusion models are not accurately mapped back to the initial noise used to generate them, our method is based on an empirical observation: patches with small enough ℓ_2 differences are almost always generated from the same seed, suggesting consistent behavior under approximate inversion. The ℓ_2 distance, as a metric used to determine whether a reconstructed noise patch matches the original, yields over 99.9% ROC-AUC.

Prompt Inversion Attack. An attacker may choose to use prompt inversion methods such as [21]. When an attacker approximates the prompt, our semantic safeguard may be less effective. Yet, forgery attacks often aim to harm the owner's reputation. Our method ensures that even if a forged image is produced, its semantics remain somewhat close to the original watermarked image.

Attacking the Confusion Band. As shown in Figure 5, at certain levels of semantic similarity, our method may confuse images that are related or unrelated to the embedded caption. However, a forger's ability to exploit this ambiguity is inherently constrained by the semantics of the original watermarked images — They can only forge images that are sufficiently similar to those whose watermark they supposedly managed to replicate. This limitation could potentially be mitigated in the future through improved embedding models.

Watermarking Capacity. The ability to encode a very large number of distinct watermarks, or to identify one user out of many millions possible watermark owners, may raise concerns. As previous work has shown, noise-based watermarking methods can support millions of different users by using different hash salts, while still ensuring that the key patterns remain distinguishable [4, 14].

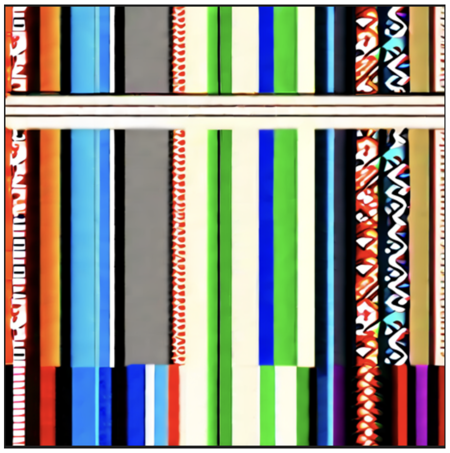

Figure 8. ***Impact of Repetitive Patches in the Initial Noise on Image Generation.***

13. Additional Experiments

13.1. CatAttack Performance vs. Object Scale

We varied the size of the pasted object in the CatAttack from 10% to 40% of the image area and evaluated detection performance at each scale. Table 4 reports the ROC-AUC (%) for each object scale, showing a gradual improvement from 95.4% at a 10% scale to 98.0% at a 40% scale.

Table 4. ***CatAttack Detection Performance vs. Object Scale.*** ROC-AUC (%) for different object sizes, as detected by SEAL + Spatial Test.

Scale (% of image)	10	20	30	40
ROC-AUC (%)	95.4	96.2	97.7	98.0

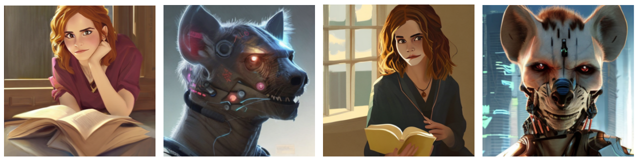

Figure 9. Comparison between the proxy image x_{pre} (left two) and the final generated image x (right two). Zoom in for clarity.

13.2. Effect of Insertions on Semantic Embeddings and LSH Binning

We evaluate how localized insertions (e.g., “cat”, “house”, “human”) impact the semantic embedding vector and its SimHash bin assignments. Table 5 reports the average angular shift $\Delta\theta$ between the original and edited embeddings, along with the proportion of hash bins that flip due to each insertion. The results show that even modest insertions can produce substantial rotations in semantic space (up to $71.2°$) and high bin-flip ratios (exceeding 90%), underscoring the sensitivity of LSH-based watermark detection to semantic changes. The experimental details are similar to those of *Cat Attack* in Section 4.

Table 5. ***Insertion Type Analysis.*** *The angle between the original and edited semantic embeddings, and the ratio of changed bins.*

Insert Type	Cat	House	Human
Angle (°)	71.2 ± 13.8	60.3 ± 19.7	68.8 ± 17.3
Flip Ratio	$96\% \pm 4\%$	$90\% \pm 8\%$	$94\% \pm 5\%$

13.3. Robustness under Regeneration Attack

We benchmark SEAL against a range of detection approaches, including both generation-time and post-hoc methods such as HiDDeN [37], Stable Signature [12], TrustMark [6], and WOUAF [18]. As shown in Table 6, SEAL maintains a 98% detection accuracy under a regeneration attack [36], outperforming prior methods.

Table 6. Robustness of watermarking methods under regeneration-based removal attacks [36]. We report the watermark detection accuracy on regenerated (attacked) images (%).

Method	HiDDeN	StableSig	TrustMark	WOUAF	SEAL
Acc. (%)	47	41	5	51	**98**

13.4. Proxy vs. Generated Image Comparison

In Figure 9, we present a visual comparison between the proxy image x_{pre}, which guides the watermark placement, and the final generated image x, illustrating SEAL’s ability to preserve semantic intent during watermarked generation.

Table 7. ***CLIP Score Evaluation.*** *Comparison of CLIP scores before and after watermarking for images generated using prompts from the Stable-Diffusion-Prompts [27] and COCO [20] dataset.*

Stable-Diffusion-Prompts		COCO	
CLIP (before)	CLIP (after)	CLIP (before)	CLIP (after)
32.378	32.401	31.365	31.499

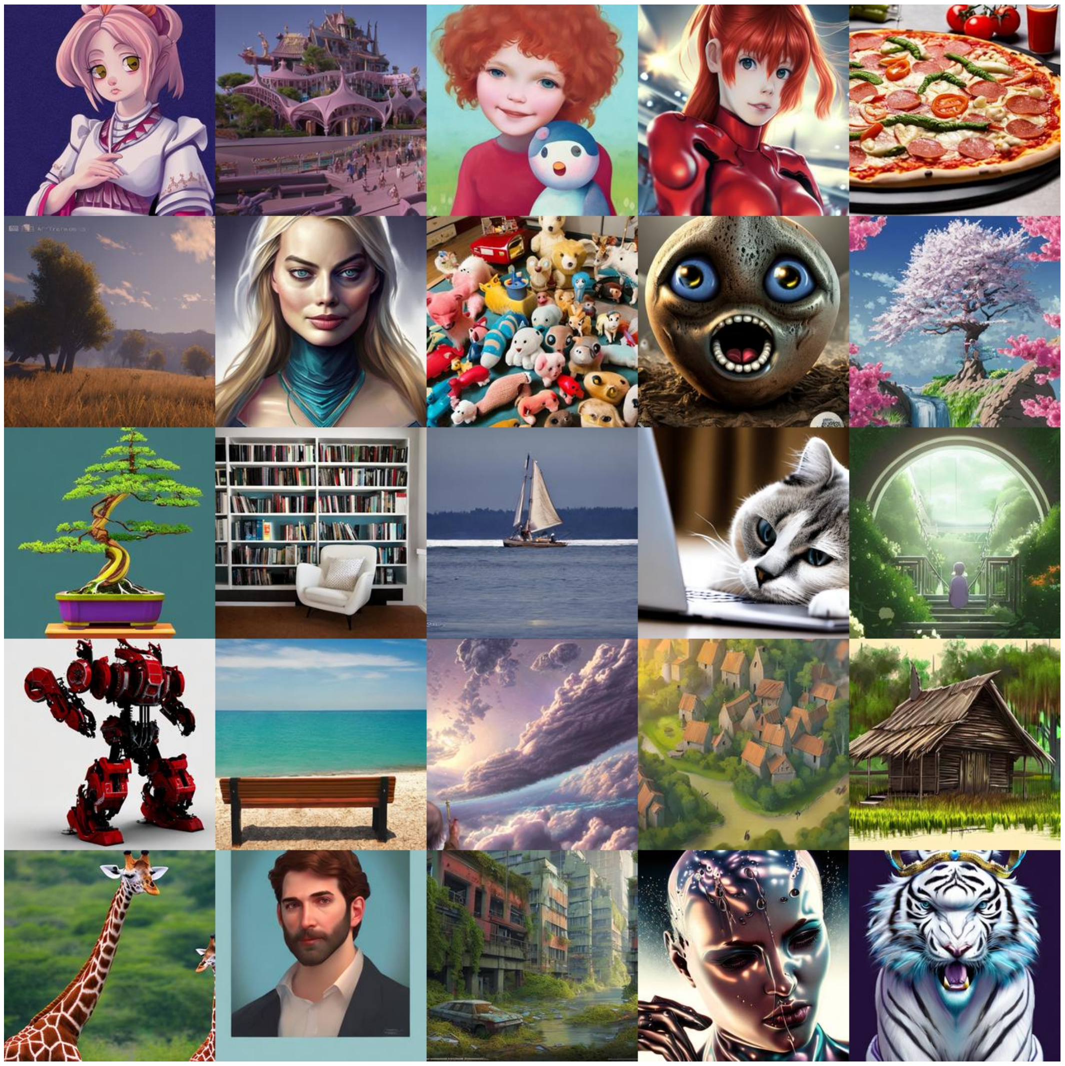

Figure 10. ***Watermarked images generated using* SEAL.**