DISTORTION-FREE SEMANTIC WATERMARKS FOR BLACK-BOX GENERATIVE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

We present a provably distortion-free semantic watermarking algorithm that requires only black-box sampling access to a generative model and black-box access to a semantic embedding. A watermark is called distortion-free if, averaged over its secret detection key, it leaves the generator's output distribution unchanged; this provides a formal guarantee that watermarking does not reduce output quality. Conventional such watermarks are often brittle to transformations. This is especially problematic for images, where benign operations such as compression, cropping, or resizing substantially change the entire pixel representation. To this end, semantic watermarks introduce the requirement that watermark detection should be robust to transformations that preserve content. Previous methods combining these properties rely heavily on specific white-box properties of the underlying model: either to autoregressive text generation or to diffusion-specific access, such as control of the initial noise or diffusion inversion. In contrast, our construction is fully generic: It only interacts with the generative model and the semantic embedding, a function that determines the notion of "preserving content" under which detection should be robust, via sample and query access. We then instantiate this framework for image generation. We characterize the two competing properties required of the semantic embedding: it should remain stable under transformations that preserve an image's content, but should distinguish independent generations from the same prompt. We train an image encoder directly for this tradeoff. The resulting encoder improves separation while maintaining stability over off-the-shelf alternatives and enables strong aggregate detection while remaining robust to various transformations. We demonstrate the effectiveness of our watermark scheme on both open-source and closed models, which is possible only due to restriction to black-box access.

1 INTRODUCTION

Generative models can produce large amounts of text and images for which the original source is difficult to determine. Watermarking addresses this problem by embedding a signal that can later be detected using a secret key. Such a signal can be used to reliably attribute content to a particular model or service.

A watermark should not noticeably degrade the model's output. Christ et al. (2024) introduced a theoretical framework to guarantee the preservation of quality via the notion of cryptographic undetectability: without a (secret) watermark detection key, no efficient observer should be able to distinguish watermarked outputs from ordinary model outputs; in particular, no benchmark may show any degradation. A weaker variant of the above notion is distortion-freeness, under which the marginal distribution of a (single) watermarked output, averaged over the secret key, is identical to the original output distribution. This approach was subsequently adopted in practical systems. DeepMind introduced a configurable distortion-free version of SynthID-Text and deployed it in Gemini (Dathathri et al., 2024). Anthropic later adopted a version of SynthID-Text for Claude (Anthropic, 2026). The resulting deployments illustrate an important advantage of distortion-free watermarking: the quality-preservation guarantee circumvents reevaluation for every benchmark or class of prompts.

Distribution preservation, however, does not guarantee another ideal property of watermarking: robustness to edits. A watermark may be detectable in the original output but disappear after the output

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

is slightly modified. The problem with editing is highly relevant for images. Even mundane operations such as JPEG compression, cropping, resizing, or applying a filter can change essentially the entire pixel representation of the image. A watermark tied directly to that pixel representation may therefore disappear even though the content of the image remains the same. This motivates semantic watermarking, in which detection should be robust to transformations that preserve the content represented by the image.

Existing semantic watermarking methods typically rely on specific "white-box" properties of the underlying generation process. For text, this is less restrictive because many models share an autoregressive token-by-token interface. Image watermarks, on the other hand, often require control over a particular diffusion latent space or noise distribution, as well as a model-specific inversion procedure. Such access is unavailable through closed image-generation APIs and any change in the generation architecture may render the watermark scheme inapplicable.

We introduce the first watermarking framework that is simultaneously distortion-free, semantically robust, and uses only black-box access with respect to both the generative model and the semantic embedding, making it applicable to general generative models. We access the generative model only as an oracle that returns independent samples and the semantic embedding only as an oracle that maps a completed output to a vector, without assuming how either oracle is implemented.

Our algorithm samples several candidate outputs, maps them to semantic codes depending on their semantic embeddings, and selects one using a secret keyed function. By accounting for the number of candidates that received each code, we ensure that the selected output has exactly the generator's original marginal distribution when averaged over the key and the algorithm's randomness. This holds for every generator and semantic embedding. If an edit preserves the semantic representation as formalized in Appendix A, then the watermark will still be detected. We prove these guarantees for any two oracles, without requiring a particular embedding architecture or model architecture. This generality comes at a cost: single-output detection has a non-negligible false-positive rate and limited true-positive rate. We therefore focus on settings where multiple outputs can be attributed to the same source, such as images uploaded by one social-media profile or assignments submitted by one student. Aggregating the watermark detection scores across this batch of outputs enables high-confidence detection.

To apply our framework to images, we need a semantic embedding that distinguishes different images generated from the same prompt, while remaining stable under edits that preserve an image's content. Standard semantic encoders may intentionally treat such independent generations from the same prompt as similar because they depict the same high-level concept. For watermarking, however, we need to observe the variation among the generated outputs that remains distinguishable after editing. Otherwise, selecting one output rather than another cannot introduce a detectable semantic watermark. This requires randomness in the meaning of the generated content itself, beyond randomness in its pixel representation; we call this *semantic entropy* and show that reliable detection inherently requires sufficient such entropy. We train an image encoder specifically to distinguish outputs generated by the same model from the same prompt using different random seeds, while keeping each image close to edited versions of it. We compare the resulting encoder with existing image embeddings, evaluating both its stability under content-preserving transformations and its ability to distinguish independent generations across several image-generation models.

We evaluate detection from individual images and batches, robustness to common transformations, image quality, and same-prompt diversity. With 32 candidates, we obtain an AUROC of 0.873 for one image and 0.990 for ten. Even when reusing a fixed key, our method achieves the lowest FID relative to unwatermarked generations among the evaluated methods, while retaining the same Vendi diversity. We also apply our framework to OpenAI's closed GPT Image API, using only sampling access, and obtain an AUROC of 0.949 for batches of five images with eight candidates per output.

Our main contributions are therefore: (1) We give a black-box, distortion-free semantic watermarking framework for general generative models, requiring only sampling access to the generator and a semantic-embedding oracle. (2) We characterize the properties required of the semantic encoder. In particular, we identify semantic entropy among independent generations as an inherent requirement for obtaining a detectable watermark through black-box output selection. (3) For diffusion-image generation, we train an encoder and evaluate image quality, same-prompt diversity, detection, and robustness to content-preserving transformations.

2 RELATED WORKS

Early language-model watermarks embedded statistical signals during autoregressive generation. Kirchenbauer et al. (2023) promoted a keyed subset of tokens during decoding. Later work strengthened the distributional guarantees: Christ et al. (2024) formalized cryptographic undetectability, Vaikuntanathan & Zamir (2026) introduced bundle sampling that exactly preserves the output distribution, and Bahri & Wieting (2026) obtained distortion-free watermarking using only sampled text. The latter also observed that shared meanings among candidates may weaken the watermark. Our framework uses this sampling perspective, but selects images using keyed semantic scores.

Preserving the output distribution does not ensure robustness to edits. Robust text schemes withstand substitutions and paraphrasing, including a construction based on pseudorandom error-correcting codes (Kuditipudi et al., 2024; Christ & Gunn, 2024). Semantic methods make the signal depend on meaning: SemStamp uses locality-sensitive hashing, k-SemStamp uses clustering, and PMark considers multiple channel constraints (Hou et al., 2024a;b; Huo et al., 2026). These methods exploit autoregressive decisions; ours scores outputs, aggregating evidence across independent generations.

Image watermarking has largely relied on diffusion internals. Tree-Ring inserts Fourier patterns into the initial noise, while Gaussian Shading maps watermark information to Gaussian latents (Wen et al., 2023; Yang et al., 2024). Gunn et al.'s image construction uses pseudorandom error-correcting codes for multi-query undetectability and distinguishes Gaussian Shading's single-image guarantee from fixed-key reuse, where they report reduced same-prompt diversity (Gunn et al., 2025). T2SMark adds per-generation randomness to relax fixed-codeword restrictions (Yang et al., 2025). These methods require access to the initial noise which is typically unavailable through image-generation APIs and diffusion inversion for detection. NoisePrints avoids inversion but requires the generation seed (Goren et al., 2026). We instead select among completed outputs.

Images also make the motivation for semantic watermarking clear: a watermark should survive pixel-level changes that preserve content, but not changes to the content itself. Reprompting attacks expose the latter risk by generating different content that retains the original watermark (Müller et al., 2025). SEAL encodes caption-derived information in the initial noise (Arabi et al., 2025), although captions may omit appearance or composition. SemBind learns prompt-consistent codes (Zhang et al., 2026), while SLICE separates semantic factors (Gao et al., 2026). We instead use direct image representations that remain stable under content-preserving edits yet distinguish independent same-prompt generations, without modifying or inverting generation.

3 PRELIMINARIES

In this section, we introduce the notation used by our method, with additional formal definitions and discussion provided in Appendix A. Let $G : \mathcal{P} \times \mathcal{S} \to \mathcal{X}$ be a conditional generator, where $p \in \mathcal{P}$ is a prompt, $s \in \mathcal{S} = \{0,1\}^k$ is a seed, and $\mathcal{X}$ is the output space. For a fixed prompt, we write $X = G(p, S)$ for $S \sim \mathrm{Unif}(\mathcal{S})$. A semantic encoder $E : \mathcal{X} \to \mathbb{R}^d$ maps an output to a semantic representation. We use an M-bit random-hyperplane locality-sensitive hash $H_{\mathcal{R}} : \mathbb{R}^d \to \mathcal{C}$, where $\mathcal{C} = \{0,1\}^M$, which partitions the representation space into regions identified by binary codes. The binary representation also provides a natural notion of proximity between hash regions: for $c, c' \in \mathcal{C}$, the Hamming distance $d_{\mathrm{H}}(c, c')$ is the number of bits on which they differ. The parameter r_{ham} denotes the maximum number of boundary-uncertain bits considered during detection (Section 5). For a key $\kappa \in \mathcal{K}$, where $\mathcal{K}$ is the secret-key space, a keyed pseudorandom function $F_\kappa : \mathcal{C} \to [0,1]$ assigns each hash code a secret score.

4 SEMANTIC ENCODERS FOR DISTORTION-FREE WATERMARKING

Distortion-freeness imposes an inherent entropy requirement. If a prompt deterministically produces one output, any distortion-free scheme must return it, making watermarked and nonwatermarked outputs indistinguishable. More generally, Christ, Gunn, and Zamir showed that the detection power of distortion-free watermarking is necessarily tied to generation entropy (Christ et al., 2024).

Semantic robustness creates a stronger limitation: a generator may have high output entropy while producing outputs equivalent under the chosen semantics. For example, the same scene viewed from slightly different viewpoints. A robust semantic representation should map them to one semantic region, leaving no variation from which to introduce a detectable signal. We therefore require sufficient variation in the semantic representations of outputs sampled from the same model and prompt.

We refer to this variation as *semantic entropy*. This requirement is inherent to distortion-free semantic watermarking rather than specific to our construction.

Both our detection-power analysis and our robustness guarantee depend on the chosen semantics, which we represent through a semantic embedding. Our framework is not specific to images or to a particular embedding architecture. For our image instantiation, however, the embedding must satisfy two complementary requirements: an image and its content-preserving transformations should remain close in the embedding space, while independent same-prompt generations should remain distinguishable. Most semantic encoders are designed to separate images associated with different concepts or prompts, rather than different generations of the same prompt. Yet changing the random seed can substantially alter content and composition even when the prompt remains fixed (Xu et al., 2025). We therefore train the encoder to capture these instance-level differences.

Figures 3 and 4 in Appendix D illustrate the representation requirements and training pairs: transformations of one image should preserve its representation, whereas an independent same-prompt generation may receive a different representation. Existing representations are not explicitly optimized for both requirements: CLIP emphasizes image–text alignment, DINOv2 invariance across augmented views, and TPIPS perceptual similarity. We therefore fine-tune DINOv2 with a contrastive objective that treats content-preserving transformations as positives and different-seed generations from the same prompt as negatives. Appendix D provides architecture and training details.

Unlike exact-output entropy, the Vendi Score (Friedman & Dieng, 2023) quantifies how diverse same-prompt generations appear to a particular encoder: as the von Neumann entropy of its normalized cosine-similarity kernel, it treats nearly identical representations as similar.

Definition 1 (Semantic entropy)**.** Given a semantic encoder E, a generative model G with finite seed set $\mathcal{S}$, and a prompt p, define

$$z_s = \frac{E(G(p,s))}{\|E(G(p,s))\|_2}, \qquad s \in \mathcal{S}.$$

Let $K_{E,p} \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ denote the cosine-similarity matrix with $(K_{E,p})_{s,t} = z_s^\top z_t$. We define the semantic entropy of G for prompt p, measured in the representation space induced by E, as

$$\mathrm{Ent}^E_{\mathrm{sem}}(G \mid p) = -\operatorname{Tr}\left(\frac{K_{E,p}}{|\mathcal{S}|}\log\frac{K_{E,p}}{|\mathcal{S}|}\right).$$

Its exponential is the corresponding Vendi score (Friedman & Dieng, 2023).

Because enumerating all seeds is impractical, we estimate semantic entropy from n independent samples and report $\widehat{H}$, the estimate normalized by its maximum value, $\log n$; the unnormalized estimate converges to full-seed entropy as n grows (Ospanov & Farnia, 2025, Corollary 2(a)).

Semantic entropy captures the representation-level variability available for watermarking. The next section shows how it yields a detectable signal while preserving the generator's output distribution.

5 METHOD

Figure 1 summarizes watermarked generation. Given a prompt, we sample N independent candidate images. We use the given semantic encoder E and fixed LSH function $H_{\mathcal{R}}$ to map each candidate to an M-bit code. Then, we use the secret key κ, fixed across generations, to assign each such code a pseudorandom score $F_\kappa(c)$. Inspired by the bundle-sampling idea of Vaikuntanathan & Zamir (2026), we take the collisions in semantic codes into consideration before using either the scores or true randomness to pick one of the samples. This guarantees that we select every candidate with probability $1/N$ when averaged over the secret key and sampler randomness, yielding distortion-free generation. Algorithm 1 in Appendix B gives the complete procedure.

Detection reuses the same encoder, hash function, and fixed key. It maps each image to its LSH region and reads the corresponding secret-key score. For B images evaluated under the same key κ, the detector averages their scores and compares the result with a threshold; $B = 1$ gives single-image detection. The black-box detector needs neither the original image nor access to the generator.

In realistic settings, content-preserving transformations such as reframing, cropping, rotation, resizing, compression, and filtering may slightly move the semantic embedding of an output. Its score

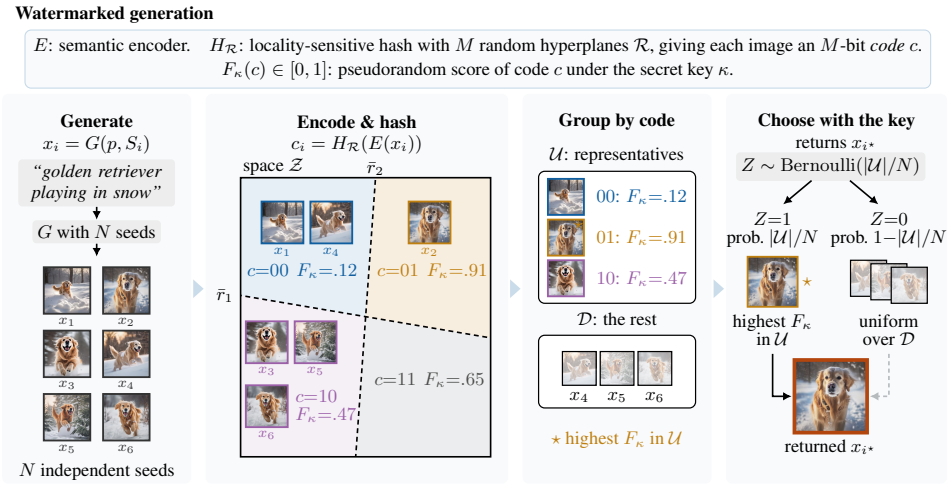

Figure 1: **Overview of watermarked generation.** Candidates are encoded, hashed, and grouped by semantic code. The sampler returns the highest-scoring representative in $\mathcal{U}$ with probability $|\mathcal{U}|/N$ and otherwise samples uniformly from $\mathcal{D}$, preserving selection probability $1/N$ for every candidate when averaged over the key and sampler randomness.

is unchanged while it remains in the same region of the space; after crossing an LSH boundary, it may receive a different score. For a representation near a boundary, the detector also considers neighboring-region scores, as detailed in Appendix B. In batch detection, such as identifying whether a social-media account repeatedly shares watermarked content, error decreases exponentially with the number of independently generated images when the expected watermarked and unwatermarked scores are separated by the threshold (Lemma 18 in Appendix C). Appendix B gives the complete per-image and batch procedures in Algorithms 2 and 3.

6 Theoretical Guarantees

We now establish the two guarantees underlying our framework: distortion-free generation and an expected keyed-score gap between watermarked and unwatermarked generations.

Theorem 3 (Distortion-free generation).
Fix a prompt p. Let K be sampled uniformly from the key space, and let X^{wm} be the output of Algorithm 1 using key K. Under the ideal-PRF assumption (Assumption 17),

$$\mathbb{E}_{K\sim\mathrm{Unif}(\mathcal{K})}\left[P_{G_K^{\mathrm{wm}}}(\cdot \mid p)\right] = P_G(\cdot \mid p).$$

Thus, after averaging over the secret key, a watermarked output has exactly the same distribution as a sample from the original generator.

Although the watermarked output follows the original generator distribution when averaged over the key, a fixed key still affects which output is selected. The detector measures that dependence through the keyed score. Its strength depends on the number $L = |U|$ of distinct semantic codes occupied by the N candidates.

Theorem 4 (Expected keyed-score gap).
Fix a batch of N candidates whose LSH codes contain L distinct values. Let $I^\star$ be the index selected by Algorithm 1, and let y be any fixed unwatermarked sample with code c_y. Under the ideal-PRF assumption,

$$\Delta_L = \mathbb{E}[F_K(c_{I^\star}) - F_K(c_y)] = \frac{L(L-1)}{2N(L+1)}, \tag{1}$$

where the expectation is over a uniformly sampled key and the sampler's internal randomness, with the candidate batch and its codes held fixed.

The formula shows that code occupancy governs the signal: the gap is zero when all candidates share one code and reaches $(N-1)/(2(N+1))$ when all N codes are distinct, so increasing N

helps only when it adds occupied codes. Across independent generations, averaging bounded scores makes both detection errors decrease exponentially with batch size.

The encoder must separate independent generations to reduce collisions and increase L, yet keep content-preserving transformations close enough to retain the code and keyed score. Empirically, our encoder achieves the highest mean detection AUROC across all three evaluated configurations (Appendix G.2), while Section 7.3 links higher encoder similarity to greater watermark retention. Appendix C proves distortion-freeness from uniform candidate selection, the score gap and batch concentration, and geometric conditions for code occupancy and clean-score preservation.

7 Experiments and Results

Experimental setup. We conduct the watermarking experiments using Stable Diffusion 2.1 (Stability AI, 2022; Rombach et al., 2022) and COCO-2017 prompts (Lin et al., 2014; Chen et al., 2015) held out from semantic encoder training. Our black-box method selects among complete candidate images. Because sampling-based watermarks primarily target language while image methods typically access the generation process, no directly comparable image-watermarking baseline shares the same sampling-only access model. We therefore compare with Tree-Ring (Wen et al., 2023), Gaussian Shading (Yang et al., 2024), and SEAL (Arabi et al., 2025), three closely related methods for watermarking diffusion-generated images. We evaluate all methods on a common generation backbone using each method's native generation and detection procedures. For representation analysis, we compare our encoder with DINOv2 (Oquab et al., 2024), CLIP (Radford et al., 2021), DINOv3 (Siméoni et al., 2025), and TPIPS (Wang et al., 2026) on independent generations from SD2.1 and two FLUX variants (Black Forest Labs, 2024) and content-preserving perturbations of real images. To demonstrate closed-API use, we additionally evaluate GPT Image through black-box sampling (Appendix G.3). The subsections and appendix provide experiment-specific details.

7.1 Generation Quality and Diversity

Our distortion-free guarantee matches the original generator distribution after averaging over secret keys; it does not require each fixed key to preserve that distribution. Because deployment typically reuses one key across many generations, we empirically evaluate fixed-key quality and diversity. Prior work shows that fixed-key reuse can reduce same-prompt diversity for some initial-noise watermarking methods (Gunn et al., 2025; Yang et al., 2025). We therefore measure fixed-key distributional deviation and same-prompt diversity; Appendix G.4 reports prompt alignment.

We evaluate Stable Diffusion 2.1 Base at 512×512. We measure distributional deviation by computing FID between each watermarked population and its matched unwatermarked population, using 10,000 generations from 1,000 COCO-2017 prompts at 10 generations per prompt. To assess same-prompt diversity, we generate 60 images for each of 60 prompts and measure the relative Vendi Score in a frozen DINOv2-base feature space, independent of our semantic encoder. Each configuration uses an unwatermarked reference set with the same model, prompts, and sampling settings.

Table 1: **Fixed-key generation quality and diversity.** FID compares 10,000 matched watermarked and clean generations from 1,000 COCO-2017 prompts (lower is better). Diversity change is $100(\mathrm{Vendi}_{\mathrm{wm}}/\mathrm{Vendi}_{\mathrm{clean}} - 1)$ over 60 prompts with 60 generations each; 0% denotes equal diversity.

Method	FID ↓	Diversity change
Ours $(N, M) = (8, 4)$	4.71	$+0.3\%$
Ours $(N, M) = (16, 8)$	5.01	-0.7%
Ours $(N, M) = (32, 12)$	5.19	-0.2%
Tree-Ring	5.61	$+0.4\%$
Gaussian Shading	5.42	-9.1%
SEAL	6.88	-13.2%

Table 1 reports both quantities. Across our three configurations, matched-population FID ranges from 4.71 to 5.19, averaged across 10 key selections, compared with 5.42–6.88 for the baselines. Same-prompt diversity remains within 0.8% of clean generation, whereas Gaussian Shading and SEAL reduce it by 9.1% and 13.2%, respectively; Tree-Ring is unchanged. Together, these re-

sults show that, among the evaluated methods, our method has the smallest measured distributional shift under fixed-key reuse while retaining near-clean same-prompt diversity. Appendix G.4 reports complementary FID against real COCO-2017 images and image-prompt alignment results.

7.2 Watermark Detection and Robustness

We evaluate detection on clean outputs and after common content-preserving transformations.

Clean images and image sets. Using 1,000 held-out COCO-2017 prompts, we evaluate matched watermarked and unwatermarked generations for $(N, M) \in \{(8, 4), (16, 8), (32, 12)\}$ and set sizes $B \in \{1, 5, 10\}$. We report AUROC and TPR at empirical FPRs of 1%, 5%, and 20%. Tree-Ring, Gaussian Shading, and SEAL provide native single-image references; Appendix F.1 reports the full curves, aggregation protocol, and available post-transformation and calibration FPRs.

Table 2 shows that aggregation improves detection: for $(N, M) = (32, 12)$, TPR@20% rises from 0.853 for one image to 0.954 for five and 0.987 for ten. While native baselines are stronger for single-image detection because they modify the generation process, our black-box method instead selects among N complete outputs and preserves its distortion-free guarantee. Consequently, it faces a theoretical single-image ceiling, $\mathrm{TPR} \leq N\,\mathrm{FPR}$, inherent to black-box sampling (Appendix H).

We also test the framework through OpenAI's closed GPT Image API, using eight candidates per output. With only sampling access to the generator, we obtain an AUROC of 0.769 for individual images and 0.949 for batches of five (Appendix G.3).

Table 2: Clean detection on 1,000 matched prompts. TPR@20% is the true-positive rate at empirical FPR $\leq 20\%$. A dash means that only single-image detection was evaluated for that native baseline.

	$B = 1$		$B = 5$		$B = 10$	
Method	AUROC	TPR@20%	AUROC	TPR@20%	AUROC	TPR@20%
Ours $(N = 8, M = 4)$	0.736	0.556	0.870	0.782	0.912	0.864
Ours $(N = 16, M = 8)$	0.820	0.740	0.949	0.926	0.980	0.975
Ours $(N = 32, M = 12)$	0.873	0.853	0.967	0.954	0.990	0.987
Tree-Ring	1.000	1.000	–	–	–	–
Gaussian Shading	1.000	1.000	–	–	–	–
SEAL	1.000	1.000	–	–	–	–

Robustness under common transformations. To approximate deployment conditions, we evaluate images after transformations commonly encountered when they are stored, shared, or edited: appearance changes, compression, geometric transformations, and localized edits produced by a generative image-editing model. For the configurations $(N, M) \in \{(8, 4), (16, 8), (32, 12)\}$, we compare with the baseline methods in the single-image setting. We report both detection after transformation and retention, defined as the fraction of images detected before transformation that remain detected afterward. This distinction separates robustness from the initial detection strength of each method. Appendix F.1 details transformations, detector configurations, calibration, and settings.

Figure 2 shows different patterns across transformation types. The baselines are stronger under appearance changes and compression, whereas all three of our configurations retain more initial detections under geometric transformations. For $(N, M) = (32, 12)$, detection after transformation reaches 45.4%, close to Tree-Ring (46.7%) and above Gaussian Shading and SEAL. Thus, detection after transformation and retention provide complementary views of robustness.

Across the 80 generative edits, $(N, M) = (32, 12)$ achieves the highest TPR after editing (65.0%), exceeding Tree-Ring (60.0%), Gaussian Shading (37.5%), and SEAL (10.0%). Retention instead peaks at 84.2% for $(16, 8)$, showing that stronger absolute detection does not imply better retention.

7.3 Semantic Representation Analysis

Our watermarking method relies on a semantic representation with two complementary properties: content-preserving transformations should produce similar representations, while independent gen-

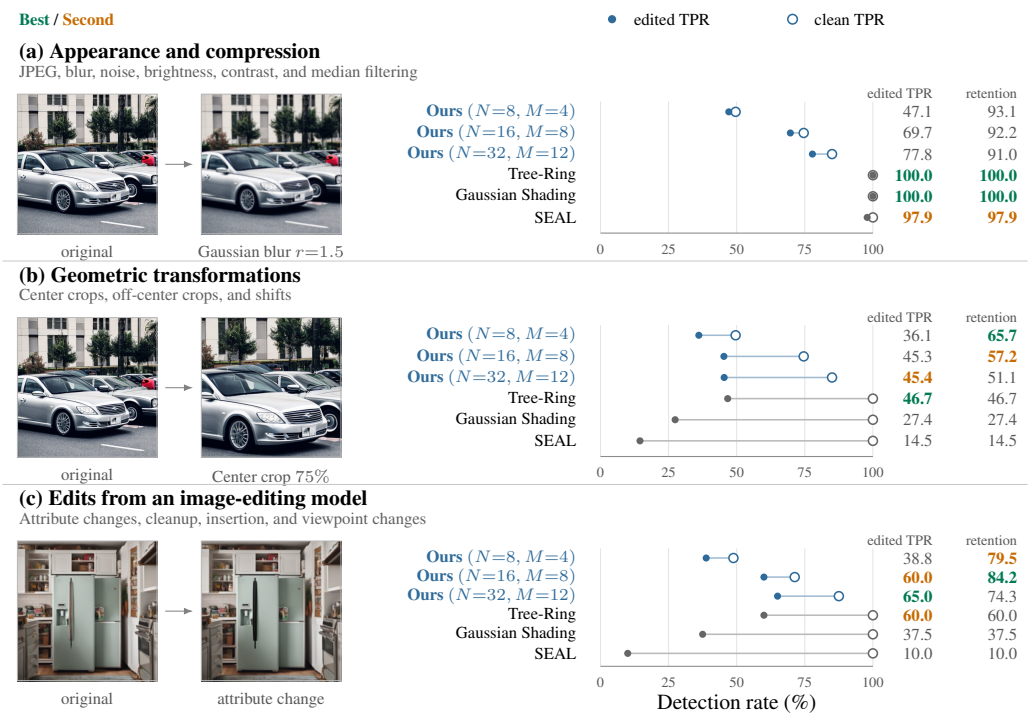

Figure 2: Robustness at 20% nominal clean FPR. Filled and open markers show edited and clean detection, respectively; retention is the fraction of initial detections preserved after editing. Panels (a,b) summarize appearance/compression and geometric transformations, while panel (c) reports 80 generative edits. Colors mark the best and second-best values. Appendix F.1 provides the detector configurations, individual settings, protocols, and available post-transformation and calibration FPRs.

erations should remain distinguishable. We analyze whether the learned encoder satisfies these properties, its transfer across generators, and how its geometry relates to watermark retention.

We compare our encoder with DINOv2-base, CLIP ViT-B/32, DINOv3 L/16, and TPIPS-Embed. To evaluate same-prompt diversity, for each of 200 COCO-2017 prompts per generator, all held out from encoder training, we generate 16 images using different random seeds with SD2.1 and two FLUX variants. Testing these generators evaluates transfer beyond the SDXL (Podell et al., 2024) data used for training. Appendix E reports the complementary evaluation of stability under content-preserving transformations and provides full protocol details.

We quantify these properties using mean pairwise cosine similarity and estimated semantic entropy. Cosine similarity measures how closely representations are clustered, while $\widehat{H}$ estimates their normalized semantic entropy as defined in Definition 1. For content-preserving transformations, higher cosine similarity and lower $\widehat{H}$ indicate greater invariance. For independent same-prompt generations, lower cosine similarity and higher $\widehat{H}$ indicate greater diversity in the encoder's representation.

Table 3 compares how the encoders represent independent generations from the same prompt. Our encoder achieves the lowest cosine similarity and highest estimated semantic entropy for every evaluated generator, including SD2.1 and both FLUX variants, which were not used for training. Appendix E reports stability under content-preserving transformations. Across these transformations, our encoder and DINOv3 are strongest overall: ours performs best under JPEG compression and resizing, while DINOv3 performs best under Gaussian blur and mild crop-and-color transformations.

Appendix E connects these representation properties to the discrete codes used by the watermark. Across independent generations, our encoder produces more occupied hash codes and fewer collisions. As established by Theorem 4, greater code occupancy yields a larger expected keyed-score

Table 3: **Same-prompt diversity in encoder space.** Mean pairwise cosine similarity (Cos.) and estimated normalized semantic entropy ($\widehat{H}$) across 16 independent generations from the same prompt. Lower cosine similarity and higher estimated semantic entropy indicate that the generations are more diverse in the encoder's representation. Appendix E provides the complete protocol and stability results under content-preserving transformations.

	DINOv2		CLIP		DINOv3		TPIPS		Ours	
	Cos. ↓	$\widehat{H}$ ↑	Cos. ↓	$\widehat{H}$ ↑	Cos. ↓	$\widehat{H}$ ↑	Cos. ↓	$\widehat{H}$ ↑	Cos. ↓	$\widehat{H}$ ↑
FLUX distilled	0.760	0.372	0.900	0.186	0.722	0.418	0.829	0.304	**0.365**	**0.573**
FLUX base	0.675	0.469	0.877	0.222	0.680	0.468	0.749	0.407	**0.259**	**0.642**
SD2.1	0.634	0.516	0.863	0.246	0.651	0.500	0.645	0.537	**0.222**	**0.674**

gap. Under content-preserving transformations, our encoder maintains relatively few distinct codes across all four evaluated transformation families, complementing the continuous invariance results in Appendix E. Appendix G.2 isolates the encoder's effect in the complete watermarking pipeline, where ours achieves the highest mean AUROC in all three evaluated configurations.

Encoder similarity closely tracks watermark retention. Across the 20 transformation settings, the Spearman correlation between encoder similarity and retention ranges from 0.996 to 0.999 for our configurations, compared with 0.747 to 0.779 for the baselines. This is consistent with our geometric analysis: higher cosine similarity makes the original and transformed images more likely to receive the same LSH code and clean keyed score (Lemma 20 in Appendix C). Figure 13 in Appendix G provides the complete visualization. Together, these results suggest that an encoder better aligned with human judgments of content preservation while still separating independent generations could directly improve detection and robustness.

8 Limitations and Future Work

Our current image implementation provides weaker clean single-image detection than methods that directly modify the diffusion process (Table 2). This tradeoff matters most when every image must be detected independently and internal model access is available. Our method instead prioritizes black-box applicability, semantic binding, and distribution preservation, while aggregation substantially strengthens detection (Table 2). More generally, any stateless method that draws N samples and returns one satisfies $\mathrm{TPR} \leq N\,\mathrm{FPR}$ for each released image, regardless of the scoring rule. This $O(N)$ ceiling belongs to black-box sampling, not our construction (Appendix H). A better semantic encoder could narrow this gap and strengthen robustness. It must keep content-preserving transformations close while separating independent generations; ours improves this balance over the evaluated alternatives (Appendix G.2).

Computational cost is another significant limitation. Our strongest configuration generates 32 complete candidates per released image, which may be impractical for large-scale or latency-sensitive applications. Early-selection methods may reduce this cost by using intermediate denoiser activations to predict quality or reusing low-resolution semantic features across denoising steps (Guo et al., 2026; Habibian et al., 2024; Parmar et al., 2026). If these signals predict the final semantic code, unpromising candidates could be discarded early.

In Appendix H we discuss these limitations at length as well as further open problems.

9 Conclusion

We introduced a black-box, distortion-free semantic watermarking framework requiring only sampling access to a generator and query access to a semantic encoder. Our analysis connects detectability to semantic entropy, while experiments demonstrate aggregate detection and robustness to common transformations without compromising generation quality or diversity. These results establish semantic rejection sampling as a general black-box foundation for generative watermarking; stronger encoders, more efficient sampling, and validation beyond images remain future work.

10 AI USE STATEMENT

Generative AI assistants were used to support software development and analysis workflows, literature discovery, and the editing and formatting of manuscript text. They were also used as a source of feedback while refining the presentation of the method, experimental protocol, and theoretical exposition. We independently designed the study, ran and checked the experiments, verified code and mathematical statements, and take full responsibility for the paper's content. The images produced by the generative models evaluated in this work are experimental outputs generated under the stated protocols, rather than content produced by these assistants.

11 ETHICS STATEMENT

This work studies watermarking for AI-generated images. Its intended use is to support provenance by detecting content associated with a particular secret key. However, a positive detection result does not establish that an image is truthful, safe, or human-authored, and a negative result does not rule out synthetic origin. Any deployment should protect watermark keys and communicate the detector's scope and limitations. Our experiments use existing models and publicly available benchmark prompts; they do not involve human-subject research or new collection of personal data.

12 REPRODUCIBILITY STATEMENT

The main paper specifies the sampling-based watermarking method, its theoretical assumptions, and the experimental setup. Appendix A provides formal definitions, Appendix B gives algorithmic details, and Appendix C contains complete proofs. The semantic-encoder training and evaluation procedures, detector calibration, baseline configurations, generation settings, and robustness protocols are documented in Appendices D–F.1. These materials identify the configurations and evaluation protocols used to produce the reported results.

REFERENCES

Anthropic. How Claude's text watermark works, August 2026. URL https://www.anthropic.com/news/claude-text-watermark.

Kasra Arabi, R. Teal Witter, Chinmay Hegde, and Niv Cohen. SEAL: Semantic aware image watermarking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16196–16205, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Arabi_SEAL_Semantic_Aware_Image_Watermarking_ICCV_2025_paper.html.

Dara Bahri and John Wieting. A watermark for black-box language models. *Transactions on Machine Learning Research*, 2026. URL https://arxiv.org/abs/2410.02099.

Shuai Bai et al. Qwen3-VL technical report, 2025. URL https://arxiv.org/abs/2511.21631.

Black Forest Labs. Announcing black forest labs, 2024. URL https://bfl.ai/blog/24-08-01-bfl.

Moses S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pp. 380–388, 2002. doi: 10.1145/509907.509965.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015. URL https://arxiv.org/abs/1504.00325.

Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2818–2829, 2023. URL https://openaccess.thecvf.

com/content/CVPR2023/html/Cherti_Reproducible_Scaling_Laws_for_Contrastive_Language-Image_Learning_CVPR_2023_paper.html.

Miranda Christ and Sam Gunn. Pseudorandom error-correcting codes. In *Advances in Cryptology – CRYPTO 2024*, pp. 325–347. Springer, 2024. doi: 10.1007/978-3-031-68391-6_10. URL https://doi.org/10.1007/978-3-031-68391-6_10.

Miranda Christ, Sam Gunn, and Or Zamir. Undetectable watermarks for language models. In *Proceedings of the Thirty-Seventh Conference on Learning Theory*, volume 247 of *Proceedings of Machine Learning Research*, pp. 1125–1139. PMLR, 2024. URL https://proceedings.mlr.press/v247/christ24a.html.

Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, Jamie Hayes, Nidhi Vyas, Majd Al Merey, Jonah Brown-Cohen, Rudy Bunel, Borja Balle, Taylan Cemgil, Zahra Ahmed, Kitty Stacpoole, Ilia Shumailov, Ciprian Baetu, Sven Gowal, Demis Hassabis, and Pushmeet Kohli. Scalable watermarking for identifying large language model outputs. *Nature*, 634: 818–823, 2024. doi: 10.1038/s41586-024-08025-4.

Dan Friedman and Adji Bousso Dieng. The Vendi score: A diversity evaluation metric for machine learning. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=g97OHbQyk1.

Zheng Gao, Yifan Yang, Xiaoyu Li, Xiaoyan Feng, Haoran Fan, Yang Song, and Jiaojiao Jiang. SLICE: Semantic latent injection via compartmentalized embedding for image watermarking. *arXiv preprint arXiv:2603.12749*, 2026. URL https://arxiv.org/abs/2603.12749.

Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, 1986. doi: 10.1145/6490.6503.

Nir Goren, Oren Katzir, Abhinav Nakarmi, Eyal Ronen, Mahmood Sharif, and Or Patashnik. Noiseprints: Distortion-free watermarks for authorship in private diffusion models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=gwHCXvPDBd.

Sam Gunn, Xuandong Zhao, and Dawn Song. An undetectable watermark for generative image models. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.07369.

Huanlei Guo, Hongxin Wei, and Bingyi Jing. Toward early quality assessment of text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 38410–38419, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Guo_Toward_Early_Quality_Assessment_of_Text-to-Image_Diffusion_Models_CVPR_2026_paper.html.

Amirhossein Habibian, Amir Ghodrati, Noor Fathima, Guillaume Sautiere, Risheek Garrepalli, Fatih Porikli, and Jens Petersen. Clockwork diffusion: Efficient generation with model-step distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8352–8361, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Habibian_Clockwork_Diffusion_Efficient_Generation_With_Model-Step_Distillation_CVPR_2024_paper.html.

R. W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 29(2): 147–160, 1950. doi: 10.1002/j.1538-7305.1950.tb00463.x.

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

Abe Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. SemStamp: A semantic watermark with paraphrastic robustness for text generation. In *Proceedings of the 2024 Conference*

of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pp. 4067–4082. Association for Computational Linguistics, 2024a. URL https://aclanthology.org/2024.naacl-long.226/.

Abe Hou, Jingyu Zhang, Yichen Wang, Daniel Khashabi, and Tianxing He. k-SemStamp: A clustering-based semantic watermark for detection of machine-generated text. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 1706–1715. Association for Computational Linguistics, 2024b. URL https://aclanthology.org/2024.findings-acl.98/.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Jiahao Huo, Shuliang Liu, Bin Wang, Junyan Zhang, Yibo Yan, Aiwei Liu, Xuming Hu, and Mingxun Zhou. PMark: Towards robust and distortion-free semantic-level watermarking with channel constraints. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.21057.

Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, pp. 604–613, 1998. doi: 10.1145/276698.276876.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 17061–17084, 2023. URL https://proceedings.mlr.press/v202/kirchenbauer23a.html.

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2307.15593.

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

Andreas Müller, Denis Lukovnikov, Jonas Thietke, Asja Fischer, and Erwin Quiring. Black-box forgery attacks on semantic watermarks for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20937–20946, 2025. URL https://arxiv.org/abs/2412.03283.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2304.07193.

Azim Ospanov and Farzan Farnia. Do Vendi scores converge with finite samples? truncated Vendi score for finite-sample convergence guarantees. In *Proceedings of the Forty-first Conference on Uncertainty in Artificial Intelligence*, volume 286 of *Proceedings of Machine Learning Research*, pp. 3272–3299. PMLR, 2025. URL https://proceedings.mlr.press/v286/ospanov25a.html.

Gaurav Parmar, Or Patashnik, Daniil Ostashev, Kuan-Chieh Wang, Kfir Aberman, Srinivasa Narasimhan, and Jun-Yan Zhu. Scaling group inference for diverse and high-quality generation. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/9132fb7713ccc4860399d41695be8622-Abstract-Conference.html.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/081b08068e4733ae3e7ad019fe8d172f-Abstract-Conference.html.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Rombach_High-Resolution_Image_Synthesis_With_Latent_Diffusion_Models_CVPR_2022_paper.html.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025. URL https://arxiv.org/abs/2508.10104.

Stability AI. Stable diffusion v2.1 and dreamstudio updates, 2022. URL https://stability.ai/news-updates/stablediffusion2-1-release7-dec-2022.

Vinod Vaikuntanathan and Or Zamir. Undetectable conversations between AI agents via pseudorandom noise-resilient key exchange, 2026. URL https://arxiv.org/abs/2604.04757.

Sheng-Yu Wang, Yotam Nitzan, Aaron Hertzmann, Jun-Yan Zhu, Eli Shechtman, Alexei A. Efros, and Richard Zhang. The many senses of visual similarity: A text-prompted image perceptual metric. *arXiv preprint arXiv:2607.18237*, 2026. URL https://arxiv.org/abs/2607.18237.

Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-rings watermarks: Invisible fingerprints for diffusion images. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Chenfei Wu et al. Qwen-Image technical report, 2025. URL https://arxiv.org/abs/2508.02324.

Katherine Xu, Lingzhi Zhang, and Jianbo Shi. Good seed makes a good crop: Discovering secret seeds in text-to-image diffusion models. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3024–3034, 2025.

Jindong Yang, Han Fang, Weiming Zhang, Nenghai Yu, and Kejiang Chen. T2SMark: Balancing robustness and diversity in noise-as-watermark for diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 118642–118668, 2025. URL https://papers.nips.cc/paper_files/paper/2025/hash/abf731c2993f9b1ee417cc3734787d7a-Abstract-Conference.html.

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*, 2024.

Xin Zhang, Zijin Yang, Kejiang Chen, Linfeng Ma, Weiming Zhang, and Nenghai Yu. SemBind: Binding diffusion watermarks to semantics against black-box forgery attacks. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2601.20310.

A Formal Preliminaries and Definitions

A.1 Generative Models and Sampling

Definition 5 (Conditional generative model).
Let $\mathcal{P}$ be the set of all possible prompts, let $\mathcal{S} = \{0,1\}^k$ be the set of all possible seeds, and let $\mathcal{X}$ be the output space. A conditional generative model is a mapping

$$G\colon \mathcal{P} \times \mathcal{S} \to \mathcal{X}.$$

We model $\mathcal{S}$ as a finite set of seeds, as is the case in practical implementations, whereas the prompt space $\mathcal{P}$ may be infinite. For any fixed prompt $p \in \mathcal{P}$ and seed $s \in \mathcal{S}$, the output $G(p,s)$ is deterministic.

Definition 6 (I.i.d. sample from a conditional generative model).
Fix a prompt $p \in \mathcal{P}$. Let

$$S_1, \ldots, S_n \overset{\text{i.i.d.}}{\sim} \mathrm{Unif}(\mathcal{S}), \qquad X_i = G(p, S_i).$$

Any realization $(x_1, \ldots, x_n)$ of $(X_1, \ldots, X_n)$ is called an i.i.d. sample from G conditioned on the prompt p.

Definition 7 (Encoder and semantic encoder).
Let $\mathcal{Z} = \mathbb{R}^d$ be a d-dimensional representation space. An encoder is a mapping

$$E\colon \mathcal{X} \to \mathcal{Z}.$$

For $x \in \mathcal{X}$, we call $E(x)$ the representation of x. We call E a *semantic encoder* if its representations capture the semantic content of the outputs.

The precise properties required of a semantic encoder in our framework are introduced in Section 4.

A.2 Watermarking Generative Models

Definition 8 (Keyed watermarking scheme).
Let G be a conditional generative model and let $\mathcal{K}$ be a finite key space. A keyed watermarking scheme for G is a pair

$$\mathsf{W} = \big(G^{\mathrm{wm}}, \mathsf{Detect}\big),$$

where, for every key $\kappa \in \mathcal{K}$:

1. The *watermarked generator* is a mapping
$$G^{\mathrm{wm}}_{\kappa}\colon \mathcal{P} \times \mathcal{S}^N \to \mathcal{X}.$$
Given a prompt and N seeds, it produces a watermarked output.

2. For every batch size $B \geq 1$, the *detector* is a mapping
$$\mathsf{Detect}^{(B)}_{\kappa,\tau}\colon \mathcal{X}^B \to \{0,1\}.$$
Given a batch of images $x_{1:B} = (x_1, \ldots, x_B)$ and a fixed detection threshold $\tau \in [0,1]$, it outputs 1 when the batch is detected as watermarked under the key κ, and 0 otherwise. The case $B = 1$ corresponds to single-image detection.

Definition 9 (Black-box watermarking scheme).
A watermarking scheme $\mathsf{W} = (G^{\mathrm{wm}}, \mathsf{Detect})$ is *black-box with respect to* G if G^{wm} and Detect may interact with G only by evaluating $G(p,s)$, and otherwise have no access to its internal implementation.

Definition 10 (Distortion-free watermarking).
Let $P_G(\cdot \mid p)$ denote the distribution of $G(p,S)$, where $S \sim \mathrm{Unif}(\mathcal{S})$. For each key $\kappa \in \mathcal{K}$, let $P_{G^{\mathrm{wm}}_{\kappa}}(\cdot \mid p)$ denote the output distribution of the watermarked generator for prompt p. The watermarking scheme is *distortion-free* if

$$\mathbb{E}_{K \sim \mathrm{Unif}(\mathcal{K})}\left[P_{G^{\mathrm{wm}}_{K}}(\cdot \mid p)\right] = P_G(\cdot \mid p)$$

for every prompt $p \in \mathcal{P}$.

This is a key-averaged, single-output guarantee: it requires equality after averaging over a uniformly sampled key K, but not for every fixed key.

A.3 Hashing and Pseudorandom Functions

Definition 11 (Hash function).
A hash function is a mapping $h\colon \mathcal{Z} \to \mathcal{C}$, where $\mathcal{C}$ is a finite code space. Two representations $z, z' \in \mathcal{Z}$ *collide* under h if $h(z) = h(z')$.

Definition 12 (Locality-sensitive hash family).
Let $(\mathcal{Z}, d)$ be a metric space. A distribution $\mathcal{H}$ over hash functions $h\colon \mathcal{Z} \to \mathcal{C}$ is (r, cr, p_1, p_2)-locality-sensitive if $c > 1$, $p_1 > p_2$, and for every $z, z' \in \mathcal{Z}$,

$$d(z, z') \le r \Longrightarrow \Pr_{h\sim\mathcal{H}}[h(z) = h(z')] \ge p_1,$$

whereas

$$d(z, z') \ge cr \Longrightarrow \Pr_{h\sim\mathcal{H}}[h(z) = h(z')] \le p_2.$$

We use the standard notion of locality-sensitive hashing (Indyk & Motwani, 1998). In our method, $\mathcal{H}$ is instantiated with the M-bit random-hyperplane LSH construction for angular distance, equivalently cosine similarity, following Charikar (2002).

Definition 13 (Random-hyperplane hash).
Let $\mathcal{R} = (\bar{r}_1, \ldots, \bar{r}_M)$ be M sampled unit hyperplane normals. For a normalized representation $z \in \mathbb{R}^d$, the random-hyperplane hash is the code

$$H_{\mathcal{R}}(z) = \big(\mathbf{1}\{\langle \bar{r}_1, z\rangle \ge 0\}, \ldots, \mathbf{1}\{\langle \bar{r}_M, z\rangle \ge 0\}\big) \in \{0,1\}^M.$$

Thus, each bit indicates on which side of its corresponding hyperplane the representation lies.

Definition 14 (Hamming distance and Hamming ball).
Following Hamming (1950), the Hamming distance between two codes $c, c' \in \{0,1\}^M$ is the number of bit positions on which they differ:

$$d_{\mathrm{H}}(c, c') = \sum_{j=1}^{M} \mathbf{1}\{c_j \ne c'_j\}.$$

For an integer $r \in \{0, \ldots, M\}$, the radius-r Hamming ball centered at c is

$$\mathcal{B}_{\mathrm{H}}(c, r) = \big\{c' \in \{0,1\}^M : d_{\mathrm{H}}(c, c') \le r\big\}.$$

Definition 15 (Boundary-restricted Hamming neighborhood).
For a normalized representation z and boundary threshold δ, let

$$J_\delta(z) = \big\{j \in [M] : |\langle \bar{r}_j, z\rangle| \le \delta\big\}$$

be the set of hash bits whose corresponding hyperplanes lie within the boundary margin. Let $\mathrm{Flip}(c, S)$ denote the code obtained by flipping the bits of c indexed by S. For an integer $r \in \{0, \ldots, M\}$, define

$$\mathcal{N}_{\delta,r}(z) = \{\mathrm{Flip}(H_{\mathcal{R}}(z), S) : S \subseteq J_\delta(z),\ |S| \le r\}.$$

Every code in $\mathcal{N}_{\delta,r}(z)$ lies within Hamming distance r of $H_{\mathcal{R}}(z)$. Unlike the complete Hamming ball, however, this neighborhood permits changes only to bits associated with nearby hyperplane boundaries.

We use a pseudorandom function (PRF) to assign a secret score to each hash code, following Goldreich et al. (1986).

Definition 16 (Keyed scoring function).
Let $\mathcal{K}$ be the key space and let $\mathcal{C} = \{0,1\}^M$ be the code space. A keyed scoring family is a collection of functions

$$\{F_\kappa : \mathcal{C} \to [0,1]\}_{\kappa\in\mathcal{K}}.$$

For a secret key κ, the value $F_\kappa(c)$ is the normalized secret score assigned to code c.

Assumption 17 (Ideal-PRF scoring model).
Let $K \sim \mathrm{Unif}(\mathcal{K})$. For every collection of distinct codes $c_1, \ldots, c_L$, the scores

$$F_K(c_1), \ldots, F_K(c_L)$$

are independent and uniformly distributed over $[0,1]$. Equivalently, in the theoretical analysis, F_K is modeled as a uniformly random function from $\mathcal{C}$ to $[0,1]$. In particular, distinct codes receive independent scores, and score ties occur with probability zero.

A practical implementation can obtain normalized scores from the finite bit-string output of a cryptographic PRF. The exact equalities in our theoretical results refer to the ideal model above; a secure PRF provides the corresponding computational interpretation.

B ALGORITHMIC SPECIFICATION

Algorithm 1 Distortion-free watermarked generation

Input: Prompt p and number of candidates N
Parameters: G, E, $H_{\mathcal{R}}$, and secret key κ

Sample independent seeds $S_1, \ldots, S_N$ — *Candidate seeds*
for $i = 1, \ldots, N$ **do**
  $x_i \leftarrow G(p, S_i)$ — *Generate candidate*
  $c_i \leftarrow H_{\mathcal{R}}(E(x_i))$ — *Assign its LSH code*
end for
$\mathcal{C}_N \leftarrow \{c_i : i \in [N]\}$ — *Occupied codes*
$U \leftarrow \emptyset$
for all $c \in \mathcal{C}_N$ **do**
  $I_c \leftarrow \{i \in [N] : c_i = c\}$ — *Candidates sharing code c*
  Sample $R_c \sim \mathrm{Uniform}(I_c)$ — *Choose one representative*
  $U \leftarrow U \cup \{R_c\}$
end for
$D \leftarrow [N] \setminus U$ — *Remaining candidates*
Sample $Z \sim \mathrm{Bernoulli}(|U|/N)$ — *Preserve uniform marginal selection*
if $Z = 1$ **then**
  $A \leftarrow \arg\max_{i \in U} F_\kappa(c_i)$ — *Highest keyed score*
  $i^\star \sim \mathrm{Uniform}(A)$ — *Break ties uniformly*
else
  $i^\star \sim \mathrm{Uniform}(D)$ — *Select a remaining candidate*
end if
return $x_{i^\star}$

Boundary-aware detection. A content-preserving transformation may move an image representation across a nearby hyperplane, changing one or more bits of its observed code. Because the detector does not know the original code, it treats bits near a hyperplane boundary as uncertain and also evaluates codes obtained by flipping up to r_{ham} of these bits. It then uses the largest resulting secret-key score. When $r_{\mathrm{ham}} = 0$, the detector scores only the observed region; this is the basic exact-code case. All experiments, including clean-image detection, use $r_{\mathrm{ham}} = 1$. Restricting the search to near-boundary bits avoids evaluating the complete Hamming ball. Appendix A formally defines this neighborhood.

Algorithm 2 Per-image score computation

Input: Image x
Parameters: E, unit hyperplane normals $\bar{r}_{1:M}$, boundary threshold δ, Hamming radius r_{ham}, and secret key κ
$\mathrm{Flip}(c, S)$ denotes code c with the bits in S flipped

$z \leftarrow E(x)/\|E(x)\|_2$ — *Normalized representation*
$c \leftarrow H_{\mathcal{R}}(z)$ — *Observed LSH code*
$J_\delta(z) \leftarrow \{j \in [M] : |\langle \bar{r}_j, z \rangle| \le \delta\}$ — *Bits close to a boundary*
$\mathcal{N}_{\delta, r_{\mathrm{ham}}}(z) \leftarrow \{\mathrm{Flip}(c, S) : S \subseteq J_\delta(z), |S| \le r_{\mathrm{ham}}\}$ — *Boundary-restricted neighborhood*
return $\mathsf{Score}_\kappa(x) \leftarrow \max_{c' \in \mathcal{N}_{\delta, r_{\mathrm{ham}}}(z)} F_\kappa(c')$ — *Largest keyed score*

Algorithm 3 Detection from multiple images

Input: Batch of images $x_{1:B}$
Parameters: Per-image score Score_κ and detection threshold τ
$q \leftarrow 0$
for $b = 1, \dots, B$ **do**
  $q \leftarrow q + \mathsf{Score}_\kappa(x_b)$ *Accumulate image scores*
end for
return $\mathbf{1}\{q/B \geq \tau\}$ *Threshold the average score*

C Proofs for the Theoretical Analysis

C.1 Conditional Uniform Candidate Selection

Lemma 2.

Let $X_1, \dots, X_N \overset{\text{i.i.d.}}{\sim} P_G(\cdot \mid p)$, and let $I^\star \in [N]$ be the index selected by Algorithm 1. Under the ideal-PRF assumption (Assumption 17), for every fixed candidate batch $x_{1:N}$ and every $i \in [N]$,

$$\mathbb{E}_{K \sim \mathrm{Unif}(\mathcal{K})} \left[\Pr(I^\star = i \mid X_{1:N} = x_{1:N}, K) \right] = \frac{1}{N}.$$

Proof. Fix a candidate batch $x_{1:N}$ and its codes $c_{1:N}$. Let $L = |U|$ be the number of occupied codes, and let $n_i = |I_{c_i}|$ be the number of candidates that share candidate i's code.

Candidate i is chosen as the representative of its code with probability $1/n_i$. The representative branch is taken with probability L/N. Under the ideal-PRF assumption, the L distinct occupied codes have independent, uniform scores, so each is the maximizer with probability $1/L$. The representative branch therefore selects candidate i with probability

$$\frac{1}{n_i} \frac{L}{N} \frac{1}{L} = \frac{1}{N n_i}.$$

If $L < N$, candidate i belongs to D with probability $(n_i - 1)/n_i$. The complementary branch is taken with probability $(N - L)/N$ and selects uniformly among the $N - L$ candidates in D. Its contribution is therefore

$$\frac{n_i - 1}{n_i} \frac{N - L}{N} \frac{1}{N - L} = \frac{n_i - 1}{N n_i}.$$

Adding the two contributions gives $1/N$. If $L = N$, every candidate has a distinct code, so the complementary branch has probability zero and the representative branch alone selects each candidate with probability $1/N$. □

C.2 Distortion-Free Generation

Theorem 3 (Distortion-free generation).

Fix a prompt p. Let K be sampled uniformly from the key space, and let X^{wm} be the output of Algorithm 1 using key K. Under the ideal-PRF assumption (Assumption 17),

$$\mathbb{E}_{K \sim \mathrm{Unif}(\mathcal{K})} \left[P_{G_K^{\mathrm{wm}}}(\cdot \mid p) \right] = P_G(\cdot \mid p).$$

Thus, after averaging over the secret key, a watermarked output has exactly the same distribution as a sample from the original generator.

Proof. Fix any possible output x. For every candidate i,

$$\Pr(X_i = x \mid p) = P_G(x \mid p).$$

By Lemma 2, after conditioning on the complete candidate batch and averaging over the key, candidate i is selected with probability $1/N$. Averaging over the remaining candidates therefore also gives

$$\Pr(I^\star = i \mid X_i = x, p) = \frac{1}{N}.$$

The output equals x exactly when some candidate X_i equals x and that candidate is selected. Summing over the possible selected index gives

$$\begin{aligned}\Pr(X^{\mathrm{wm}} = x \mid p) &= \sum_{i=1}^{N} \Pr(X_i = x, I^\star = i \mid p) \\ &= \sum_{i=1}^{N} \Pr(X_i = x \mid p) \Pr(I^\star = i \mid X_i = x, p) \\ &= \sum_{i=1}^{N} \frac{1}{N} P_G(x \mid p) = P_G(x \mid p).\end{aligned}$$

Thus, the key-averaged watermarked distribution is the original generator distribution. □

C.3 Expected Keyed-Score Gap

Theorem 4 (Expected keyed-score gap).

Fix a batch of N candidates whose LSH codes contain L distinct values. Let $I^\star$ be the index selected by Algorithm 1, and let y be any fixed unwatermarked sample with code c_y. Under the ideal-PRF assumption,

$$\Delta_L = \mathbb{E}[F_K(c_{I^\star}) - F_K(c_y)] = \frac{L(L-1)}{2N(L+1)}.$$

$*$ The expectation is over a uniformly sampled key and the sampler's internal randomness, with the candidate batch and its codes held fixed.

Proof. For the fixed code c_y, the ideal-PRF assumption gives $F_K(c_y) \sim \mathrm{Unif}([0,1])$ over the random key. Hence,

$$\mathbb{E}[F_K(c_y)] = \frac{1}{2}.$$

The representative branch is taken with probability L/N. It selects the largest of the L independent uniform scores assigned to the distinct occupied codes, whose expectation is $L/(L+1)$. The complementary branch is taken with probability $1 - L/N$. Its candidate is selected independently of the PRF scores, so its score has uniform marginal distribution and expectation $1/2$. Therefore,

$$\mathbb{E}[F_K(c_{I^\star})] = \frac{L}{N}\frac{L}{L+1} + \left(1 - \frac{L}{N}\right)\frac{1}{2}.$$

Subtracting the unwatermarked expectation gives

$$\Delta_L = \frac{L}{N}\left(\frac{L}{L+1} - \frac{1}{2}\right) = \frac{L(L-1)}{2N(L+1)}.$$

□

C.4 Batch-Detection Concentration

The following lemma formalizes why averaging detection scores across independent generations improves batch detection.

Lemma 18 (Batch-detection concentration).
Fix a key κ, and let μ_0 and μ_{wm} denote the expected per-image scores for unwatermarked and watermarked generations, respectively. Suppose that

$$\mu_0 < \tau < \mu_{\mathrm{wm}},$$

where τ is the threshold used by our batch detector in Algorithm 3. For a batch of B independently generated images, the false-positive and false-negative probabilities satisfy

$$\Pr(\mathrm{FP}) \le \exp\left[-2B(\tau - \mu_0)^2\right]$$

and

$$\Pr(\mathrm{FN}) \le \exp\left[-2B(\mu_{\mathrm{wm}} - \tau)^2\right].$$

Thus, both error probabilities decrease exponentially with B.

Proof. Let $Y_1, \ldots, Y_B$ be independently generated unwatermarked images and write

$$S_b^0 = \mathsf{Score}_\kappa(Y_b), \qquad \overline{S}_B^0 = \frac{1}{B}\sum_{b=1}^{B} S_b^0.$$

For the fixed key κ, the scores $S_1^0, \ldots, S_B^0$ are independent, lie in $[0,1]$, and have expectation μ_0. A false positive occurs when $\overline{S}_B^0 \geq \tau$. Hoeffding's inequality therefore gives

$$\Pr(\mathrm{FP}) = \Pr\Big(\overline{S}_B^0 - \mu_0 \geq \tau - \mu_0\Big) \leq \exp\big[-2B(\tau-\mu_0)^2\big].$$

Similarly, let $X_1^{\mathrm{wm}}, \ldots, X_B^{\mathrm{wm}}$ be independently generated watermarked images and write

$$S_b^{\mathrm{wm}} = \mathsf{Score}_\kappa(X_b^{\mathrm{wm}}), \qquad \overline{S}_B^{\mathrm{wm}} = \frac{1}{B}\sum_{b=1}^{B} S_b^{\mathrm{wm}}.$$

These scores are independent, lie in $[0,1]$, and have expectation μ_{wm}. A false negative occurs when $\overline{S}_B^{\mathrm{wm}} < \tau$, so Hoeffding's inequality gives

$$\Pr(\mathrm{FN}) = \Pr\Big(\mu_{\mathrm{wm}} - \overline{S}_B^{\mathrm{wm}} > \mu_{\mathrm{wm}} - \tau\Big) \leq \exp\big[-2B(\mu_{\mathrm{wm}}-\tau)^2\big].$$

Because $\mu_0 < \tau < \mu_{\mathrm{wm}}$, both distances from the threshold are positive, which proves the exponential decay in B. □

C.5 Code Occupancy and Representation Geometry

We next connect the number of occupied codes to the geometry of the candidate representations. For two normalized representations with cosine similarity ρ, the probability that one random hyperplane assigns them the same bit is $1 - \arccos(\rho)/\pi$. Since the M hyperplanes are sampled independently, the probability that they receive the same M-bit code is

$$\left(1 - \frac{\arccos(\rho)}{\pi}\right)^M.$$

Thus, when candidate representations are well separated, code collisions are less likely and more distinct codes are expected. The following lemma formalizes this relationship using the maximum pairwise cosine similarity within the candidate batch.

Lemma 19 (Expected code occupancy).
Fix $N \geq 2$ candidates with nonzero encoder representations, and let

$$z_i = \frac{E(x_i)}{\|E(x_i)\|}, \qquad \rho_{\max} = \max_{i \neq j} \langle z_i, z_j \rangle.$$

Let $H_{\mathcal{R}}$ be formed from M independently sampled random hyperplanes, and let L be the number of occupied codes among $H_{\mathcal{R}}(z_1), \ldots, H_{\mathcal{R}}(z_N)$. Then

$$\mathbb{E}_{\mathcal{R}}[L] \geq \max\left\{1, N\left[1 - (N-1)\left(1 - \frac{\arccos(\rho_{\max})}{\pi}\right)^M\right]\right\}.$$

Proof. For every pair $i \neq j$, random-hyperplane hashing gives

$$\Pr_{\mathcal{R}}[H_{\mathcal{R}}(z_i) = H_{\mathcal{R}}(z_j)] = \left(1 - \frac{\arccos(\langle z_i, z_j \rangle)}{\pi}\right)^M \leq q,$$

where

$$q = \left(1 - \frac{\arccos(\rho_{\max})}{\pi}\right)^M.$$

Candidate i has a nonunique code only if it collides with at least one of the other $N-1$ candidates. By the union bound,

$$\Pr_{\mathcal{R}}[\text{candidate } i \text{ has a nonunique code}] \leq (N-1)q.$$

Let S be the number of candidates whose codes are unique within the batch. By linearity of expectation,

$$\mathbb{E}_{\mathcal{R}}[S] \geq N\big(1-(N-1)q\big).$$

Every candidate counted by S occupies a distinct code, so $L \geq S$. Also, $L \geq 1$. Combining these two bounds yields

$$\mathbb{E}_{\mathcal{R}}[L] \geq \max\big\{1, N\big(1-(N-1)q\big)\big\},$$

which is the claimed result. □

The same pairwise collision probability also yields a clean-score robustness guarantee for semantically similar images.

Lemma 20 (Clean-score preservation under semantic similarity). *Fix two images x and x' with nonzero encoder representations, and let*

$$z = \frac{E(x)}{\|E(x)\|}, \qquad z' = \frac{E(x')}{\|E(x')\|}, \qquad \rho = \langle z, z' \rangle.$$

Let $H_{\mathcal{R}}$ be formed from M independently sampled random hyperplanes, and fix a key κ. Then

$$\Pr_{\mathcal{R}}[H_{\mathcal{R}}(z') = H_{\mathcal{R}}(z)] = \left(1 - \frac{\arccos(\rho)}{\pi}\right)^{M}.$$

Whenever the two codes coincide, x and x' receive the same clean keyed score. Consequently, if the clean keyed score of x exceeds the detection threshold, then our detector also classifies x' as watermarked on this event.

Proof. For one random hyperplane, the probability that z and z' receive the same bit is $1 - \arccos(\rho)/\pi$. Independence of the M hyperplanes gives the stated code-collision probability. On the collision event,

$$F_\kappa(H_{\mathcal{R}}(z')) = F_\kappa(H_{\mathcal{R}}(z)).$$

The observed code $H_{\mathcal{R}}(z')$ is included in the neighborhood scored by our detector. Therefore, if $F_\kappa(H_{\mathcal{R}}(z)) \geq \tau$, then $\mathsf{Score}_\kappa(x') \geq \tau$, and x' is detected as watermarked. □

D SEMANTIC ENCODER TRAINING

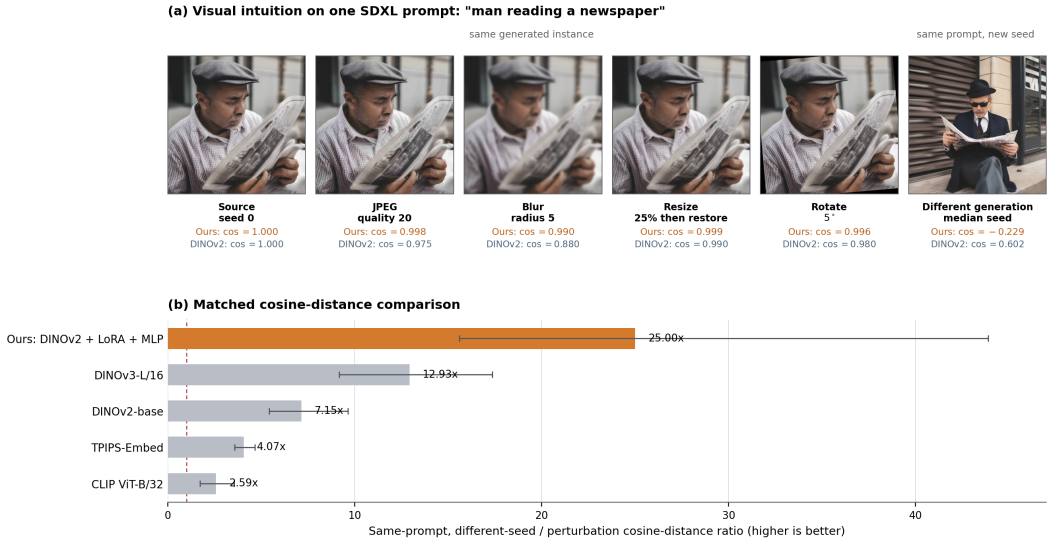

Figure 3: **The two requirements of a semantic encoder.** (a) An SDXL image, content-preserving transformations of that image, and an independent generation from the same prompt. (b) Mean cosine distance between different-seed generations divided by mean cosine distance under transformations. Higher values indicate stronger separation relative to transformation sensitivity. This ratio is distinct from the semantic entropy defined below.

We train the semantic encoder to satisfy two complementary requirements: semantics-preserving transformations of the same image should remain close, whereas independent generations from the same prompt should remain distinguishable. Figure 3 summarizes these requirements, and Figure 4 illustrates the corresponding training pairs.

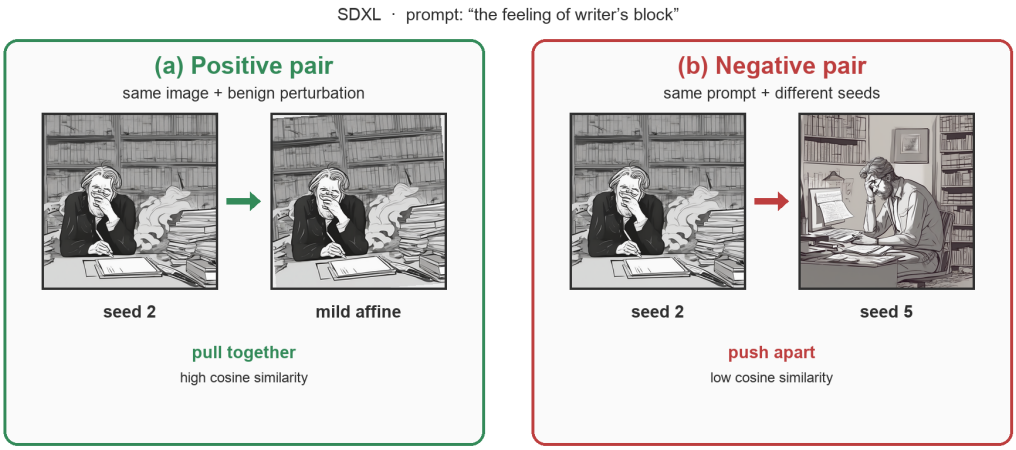

Figure 4: Training-pair construction using SDXL generations for the prompt *the feeling of writer’s block*. (a) An image and a semantics-preserving transformation form a positive pair and are pulled together. (b) Independent generations from the same prompt but different seeds 2 and 5 form a negative pair and are pushed apart.

Architecture and objective. We initialize the encoder from DINOv2-base and freeze its pre-trained parameters. Rank-8 LoRA adapters (Hu et al., 2022) are applied to the query and value projections of the last four transformer blocks. The CLS representation is passed through a $768 \to 512 \to 256$ MLP with GELU activation and dropout 0.1, followed by ℓ_2 normalization. For normalized embeddings z_i and z_j, we use

$$\mathcal{L}_+ = 1 - z_i^\top z_j, \qquad \mathcal{L}_- = \max\bigl(0, z_i^\top z_j\bigr) .$$

Positive pairs include Gaussian noise, Gaussian blur, JPEG compression, resampling, photometric changes, and affine transformations. Negative pairs contain generations obtained from different seeds of the same prompt.

Training details. We use 100 prompts evenly from DiffusionDB’s 2M-text-only split and the official COCO-2017 training captions. We divide them into 80 training and 20 validation prompts; neither set is used in the reported evaluation experiments. We use 64 seeds per prompt. Each optimizer update contains 16 pairs of one type, with pair types balanced across updates. We train using AdamW with learning rate 10^{-4}, mixed precision, gradient clipping at 1.0, and learning-rate decay of 0.95 per epoch. We select the rank-8 LoRA model with the MLP head on the validation split. The checkpoint used in Appendix E was selected after $19{,}840$ optimizer updates; the semantic-entropy results were not used for model selection.

E Encoder Evaluation: Details and Additional Results

This appendix expands the semantic representation analysis in Section 7.3. We provide the perturbation settings, analyze code counts under both transformations and independent generation, and conclude with a qualitative example. The main text gives the entropy definition and encoder comparison.

Stability evaluation. We evaluate stability using 200 real source images and 16 semantics-preserving transformations of each image and for every transformation family: JPEG compression, Gaussian blur, resizing and resampling, and mild cropping with color adjustment. Within each transformation family, severity increases gradually across the 16 variants: JPEG quality ranges from 35 to 95, blur radius from 0.25 to 1.75, resize scale from 0.65 to 0.95, and crop scale from 0.86 to 0.98, with color factors from 0.90 to 1.08. Every encoder is evaluated on the same source images and transformations. For each family, Table 4 summarizes all 16 severity levels together: we compute cosine similarity and normalized semantic entropy across the variants of each source image and then average the resulting per-image values over the evaluation set.

Table 4: **Encoder stability under content-preserving transformations.** Mean pairwise cosine similarity (Cos.) and estimated normalized semantic entropy ($\widehat{H}$). Higher cosine similarity and lower entropy indicate greater invariance.

	DINOv2		CLIP		DINOv3		TPIPS		Ours	
	Cos. ↑	$\widehat{H}$ ↓	Cos. ↑	$\widehat{H}$ ↓	Cos. ↑	$\widehat{H}$ ↓	Cos. ↑	$\widehat{H}$ ↓	Cos. ↑	$\widehat{H}$ ↓
JPEG compression	0.974	0.052	0.959	0.079	0.976	0.049	0.956	0.092	**0.985**	**0.032**
Gaussian blur	0.922	0.106	0.932	0.099	**0.974**	**0.049**	0.895	0.160	0.963	0.056
Resize–resample	0.992	0.020	0.994	0.016	0.995	0.015	0.978	0.051	**0.996**	**0.012**
Mild crop + color	0.954	0.094	0.964	0.079	**0.972**	**0.064**	0.919	0.166	0.937	0.105

Distinct codes under perturbations. We next examine whether semantics-preserving transformations retain the same discrete hash code. For each transformation type, the analysis uses 200 source images and 16 perturbed variants of each image. We apply 8-bit random-hyperplane hashing to the corresponding embeddings and average over 200 projections, using each projection consistently across all images and encoders. Figure 5 reports the resulting code counts. Fewer distinct codes indicate that transformed versions of the same image more often retain its original code. This discrete analysis complements the continuous results above.

Distinct codes under independent generations. We perform the corresponding code-count analysis on independent generations from the same prompt. For each of SDXL and the two FLUX variants, we evaluate 200 prompts with up to 16 independently generated images per prompt. All encoders are evaluated on the same images. For every encoder, results are averaged over 200 random-hyperplane projections, with each projection fixed across all images. Figure 6 reports the corresponding code counts. More distinct codes indicate fewer collisions and complement the higher semantic entropy reported in the main text. By Theorem 4, greater code occupancy yields a larger expected keyed-score gap.

Qualitative view of generation diversity. Figure 7 presents 16 SDXL images generated from one prompt. Although the images share the same description, they differ in layout and appearance. The cosine-similarity matrices show how DINOv2 and our encoder represent these differences.

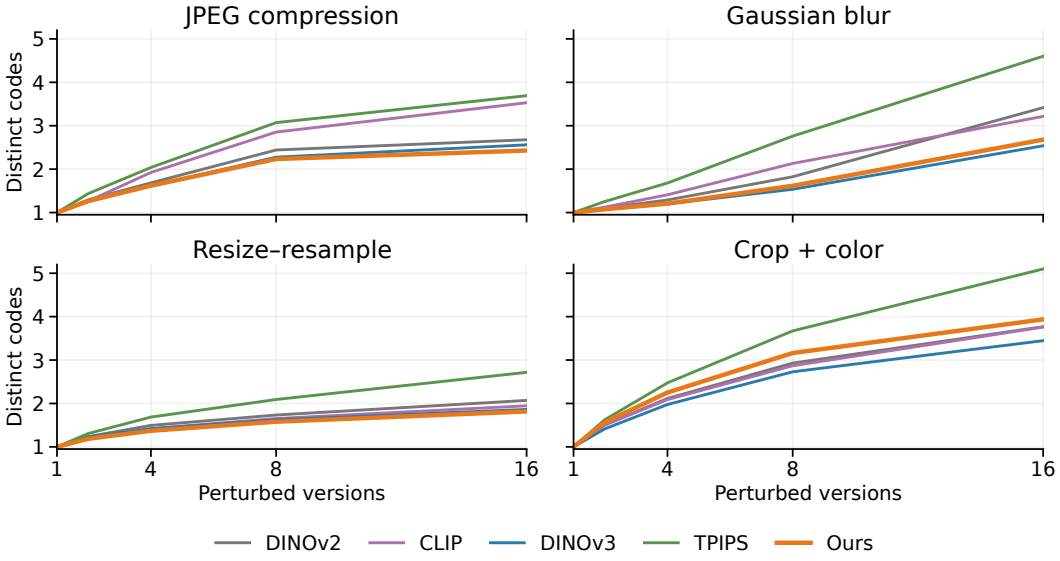

Figure 5: **Distinct codes under semantics-preserving perturbations.** Mean number of distinct 8-bit codes as additional perturbed versions are included. Each panel averages 200 source-image groups over 200 random hyperplane draws. Lower counts are desirable. Separate panels highlight the different behavior of the four perturbation types.

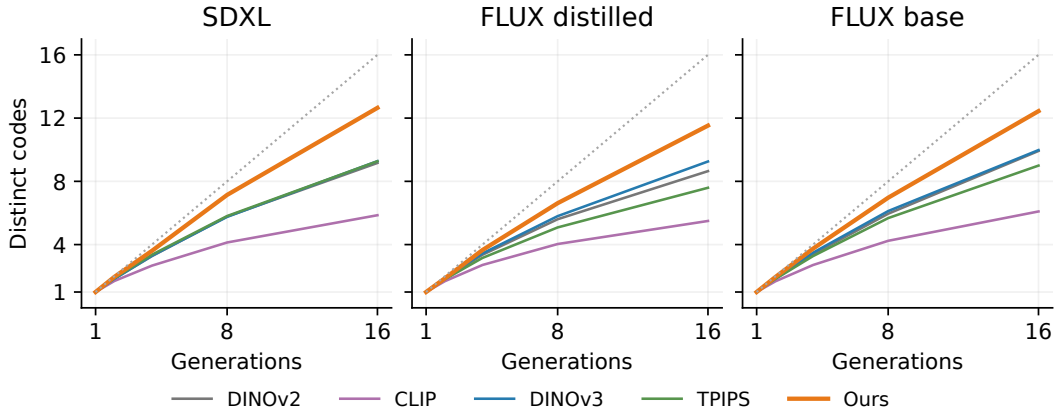

Figure 6: **Distinct codes under independent generations.** Mean number of distinct 8-bit codes as additional independent generations from the same prompt are included. Each panel averages results across 200 prompts and 200 random-hyperplane projections. Higher counts are desirable because they indicate fewer collisions. Separate panels show SDXL and the two FLUX variants, making transfer across generators visible.

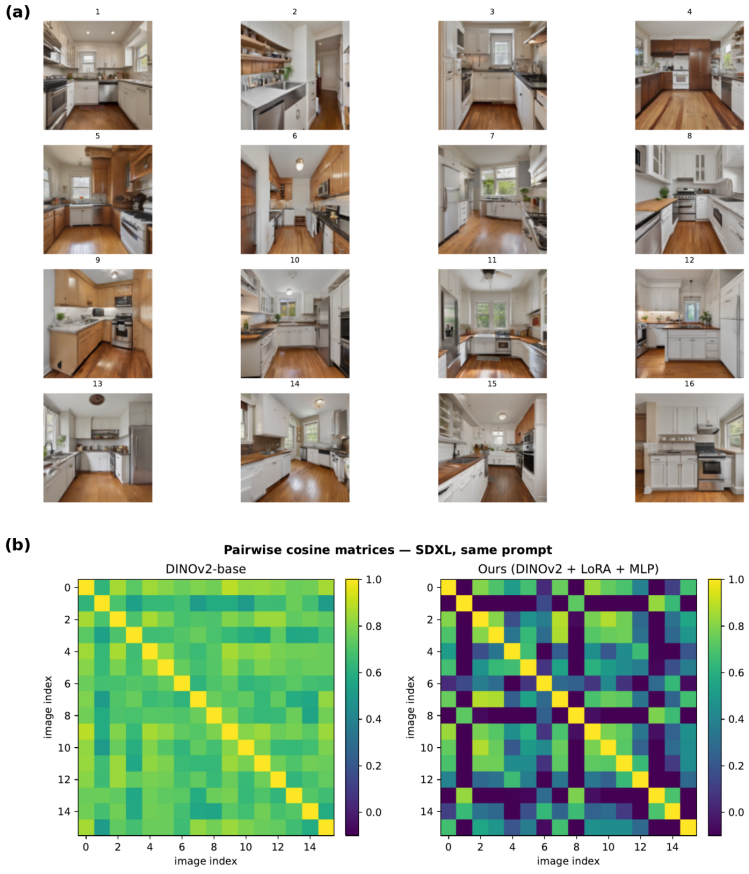

Figure 7: **Different images from one prompt.** (a) Sixteen independent SDXL generations. (b) Pairwise cosine similarities under DINOv2-base and our encoder. DINOv2 assigns high similarity across the group, whereas our encoder represents more of the differences between the images. SDXL is the encoder's training domain; the prompts are held out.

F Detection and Robustness Protocols

F.1 Evaluation Setup

Prompts and datasets. We evaluate clean detection using images generated from 1,000 captions from the COCO-2017 validation set. For each prompt, we generate both watermarked and unwatermarked images using the same model and generation settings. The watermarked images form the positive population, while the unwatermarked images form the negative population used to measure the false-positive rate. These prompts are held out from semantic-encoder training; further details about the encoder-training data are provided in Appendix D.

Detection settings. The detector has three parameters: the boundary threshold δ, the Hamming radius r_{ham}, and the score threshold τ. Boundary-aware detection is activated when an image representation lies within margin δ of at least one of the M random hyperplanes used to construct its hash code. The activation rate is the fraction of clean watermarked images for which this occurs. When activated, the detector evaluates neighboring codes that differ in at most r_{ham} bits. For each (N, M) configuration, we select δ and r_{ham} through a parameter sweep on a separate development set drawn from the COCO-2017 training set, considering clean detection, detection after editing, and the corresponding false-positive rate. We use $r_{\mathrm{ham}} = 1$ throughout and choose δ to obtain activation rates of 10%, 50%, and 75% for $(N, M) = (8, 4)$, $(16, 8)$, and $(32, 12)$, respectively. The sweep favors higher activation rates for larger M: longer codes introduce more hyperplane boundaries that an edit may cross, while the larger candidate budget provides a stronger watermark signal that can support more frequent boundary correction. With δ and r_{ham} fixed, we vary τ to obtain the required FPR. The selected settings are summarized in Table 5.

Table 5: Selected detector settings. Because δ is calibrated separately for each random-hyperplane frame, we report its target activation rate rather than a single numeric value.

(N, M)	δ calibration target	r_{ham}
$(8, 4)$	10% activation	1
$(16, 8)$	50% activation	1
$(32, 12)$	75% activation	1

Clean-detection experiment. We evaluate detection from either a single image or a set of $B \in \{5, 10\}$ images generated using the same watermark key. For image sets, we average the B per-image detection scores and compare the resulting score with τ. Watermarked images form the positive population, and unwatermarked images generated from the same prompts form the negative population. For each configuration and set size, the false-positive rate is the fraction of unwatermarked images or image sets whose score exceeds τ, while the true-positive rate is the corresponding fraction of watermarked images or sets. We compute ROC curves from these held-out positive and negative scores and report AUROC and TPR at empirical FPR targets, averaged over 25 combinations of five random-hyperplane frames and five watermark keys. Table 2 summarizes the AUROC and TPR@20% results, and Figures 8 and 9 show the corresponding TPR/FPR curves on linear and logarithmic FPR axes.

Table 6 reports the corresponding TPR at 1%, 5%, and 20% FPR for our configurations and the native single-image baselines.

F.2 Robustness Protocols and Results

Ordinary edits. We evaluate robustness on 200 prompts in the single-image setting ($B = 1$). Starting from clean watermarked and unwatermarked images, we apply the same transformation to both populations. The detector parameters δ, r_{ham}, and τ are fixed before applying the transformations and are not recalibrated on edited images. For each transformation, we report the edited TPR on watermarked images and the edited FPR on unwatermarked images. We also report paired retention: the fraction of watermarked images detected before transformation that remain detected afterward. This separates robustness to an edit from the detector's initial clean-detection rate. The evaluation

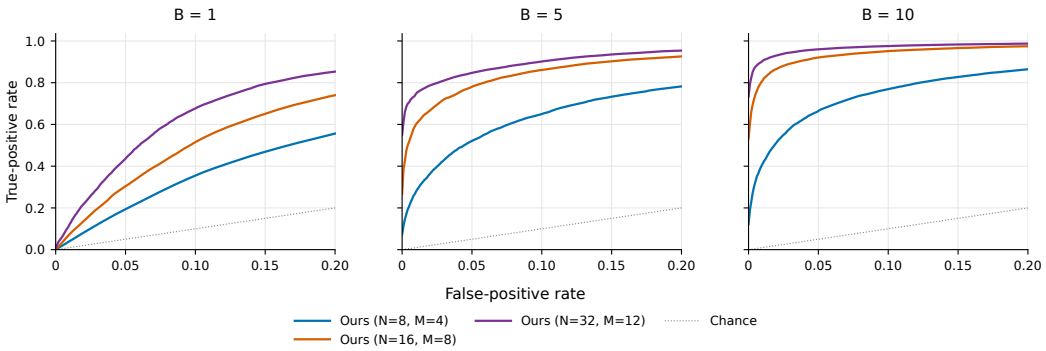

Figure 8: Clean detection for our method at $B \in \{1, 5, 10\}$ on a linear FPR axis. Curves are means over 25 hyperplane-frame and watermark-key combinations; Table 6 gives numerical operating points.

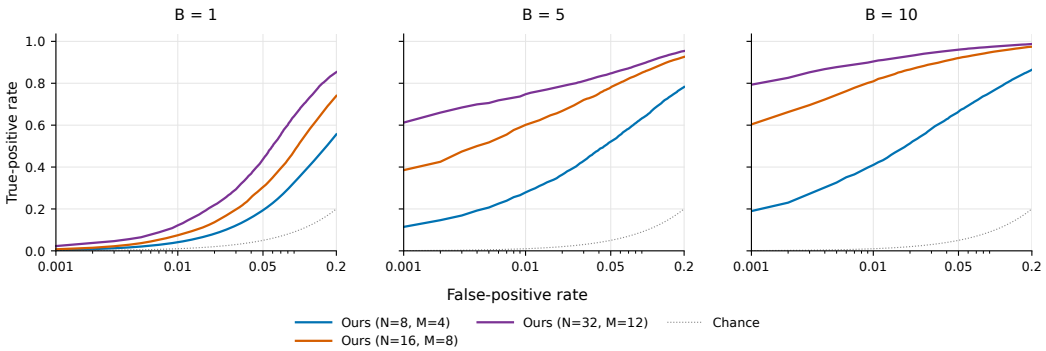

Figure 9: The same clean-detection curves on a logarithmic FPR axis, exposing aggregation gains at low FPR.

Table 6: Clean-detection TPR (%) at empirical FPR targets, averaged over 25 combinations of five hyperplane frames and five watermark keys. B is the number of images attributed to one source; baselines use their native single-image detectors.

Method	B	TPR at FPR		
		1%	5%	20%
Ours (N=8, M=4)	1	4.1	19.4	55.6
	5	27.9	52.1	78.2
	10	41.0	66.5	86.4
Ours (N=16, M=8)	1	7.4	30.4	74.0
	5	60.1	78.0	92.6
	10	80.9	92.0	97.5
Ours (N=32, M=12)	1	12.2	43.6	85.3
	5	74.8	84.7	95.4
	10	90.4	96.1	98.7
Tree-Ring	1	100.0	100.0	100.0
Gaussian Shading	1	100.0	100.0	100.0
SEAL	1	100.0	100.0	100.0

includes 20 settings spanning compression, appearance changes, noise and filtering, cropping, and translation.

DINOv2 is pretrained using augmentations including random resized crops, color changes, and Gaussian blur, and therefore already maps many such transformations to similar representations. Our fine-tuning preserves and adapts this behavior using positive pairs constructed with noise, blur, JPEG compression, resampling, photometric changes, and affine transformations, while negative pairs separate independent generations of the same prompt. The robustness evaluation includes transformations used during DINOv2 pretraining, transformations used during our fine-tuning, and transformations used during neither. Figure 11 reports the per-transformation results and marks this training exposure for each transformation. This allows us to examine generalization to held-out images and transformation settings, including transformation families not used during training.

More generally, robustness depends on the geometry of the semantic representation. For a target transformation, the method benefits from an encoder that maps the original and transformed images to nearby representations. Section 7.3 compares DINOv2 with our fine-tuned encoder and shows the relationship between encoder similarity and watermark retention. For applications involving different transformations, the semantic encoder can therefore be selected or trained to maintain high cosine similarity under those transformations.

For the non-semantic generative-edit evaluation, we use 80 watermarked images and 80 corresponding unwatermarked images, with 20 examples assigned to each of four edit categories: attribute change, background cleanup, object insertion, and camera-viewpoint change. For each source image, we provide Qwen3-VL-8B-Instruct (Bai et al., 2025) with the original image and the fixed category-specific prompt reported in Table 7. The vision model then produces an image-specific editing instruction that follows the assigned category while preserving the main subject and scene. Qwen-Image-Edit (Wu et al., 2025) receives the original image together with this instruction and generates the edited image. Figure 10 shows this pipeline on one attribute-change example. We manually review every edited image to verify that the requested change is visible while the main subject and overall scene remain preserved. Representative examples from all four edit categories are shown in Figure 12.

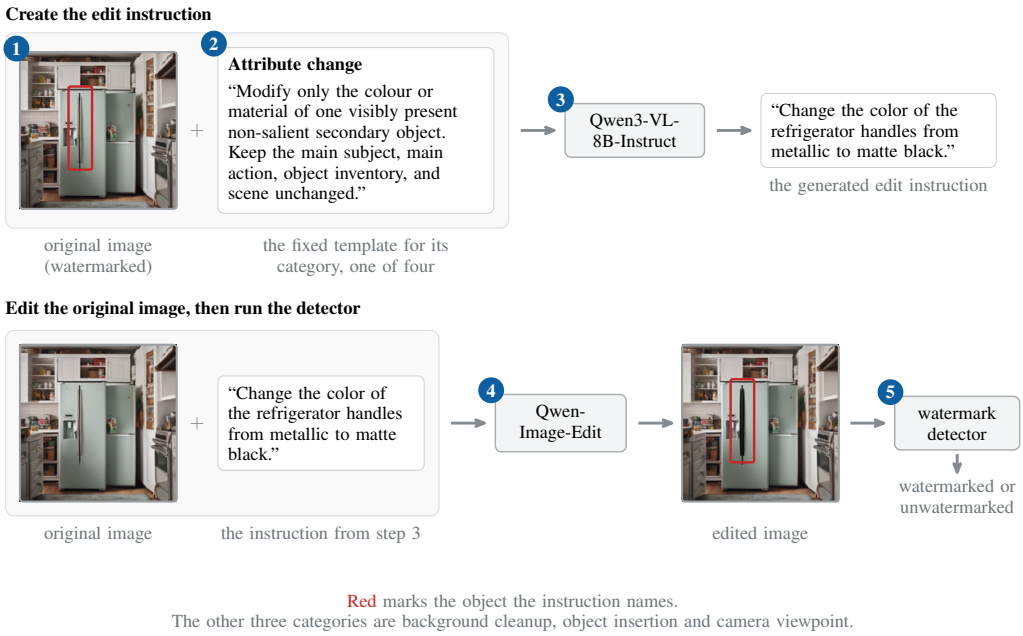

Figure 10: **Generative-edit evaluation protocol.** For each source image, the assigned category prompt and image are given to Qwen3-VL-8B-Instruct, which produces a validated image-specific instruction. The original image and this instruction are then passed to Qwen-Image-Edit, and the edited result is evaluated using detector parameters fixed before editing. The same pipeline is applied to watermarked and corresponding unwatermarked sources. The example changes the refrigerator handles from metallic to matte black; the boxes identify the object named by the instruction, not a guaranteed pixel-local change.

Table 7: Fixed category-specific constraints provided to Qwen3-VL-8B-Instruct when generating image-specific editing instructions.

Edit category	Fixed prompt constraint
Attribute change	Modify only the colour or material of one visibly present, non-salient secondary object. Preserve the main subject, main action, object inventory, and scene. Do not target faces, text, logos, or watermarks.
Background cleanup	Remove only one tiny, non-salient background clutter item. Preserve the main subject, main action, salient objects, and scene. Do not target faces, text, logos, or watermarks.
Object insertion	Add one unrelated small realistic cat in the background. The cat must be clearly visible but must not replace or occlude the main subject. Do not target faces, text, logos, or watermarks.
Camera-viewpoint change	Change only the camera viewpoint to a clearly different three-quarter or slightly elevated viewpoint. Preserve the same main subject, objects, actions, scene, colours, and lighting. Do not target faces, text, logos, or watermarks.

All results average five hyperplane frames and five PRF keys. Paired retention matches original and edited images within each method. For ordinary edits, achieved FPR is measured on transformed negatives and each reported category is an unweighted mean over its transformation settings. Table 8 summarizes the ordinary and generative-edit results. Detailed per-transformation TPR and retention results are reported in Figure 11.

The boundary-aware settings were selected on the separate development set described above. Table 9 evaluates their effect on the held-out ordinary-edit evaluation set by comparing them with exact-code detection on the same images within each (N, M) pair.

Table 8: Robustness at 20% nominal clean FPR (all values in %). Columns report post-edit TPR, paired retention conditioned on original detection, and post-edit FPR on unwatermarked images. Appearance/compression and geometric results are unweighted means over 14 and six settings, respectively; generative results summarize 80 edits. Detector settings are given in Table 5.

	Appearance/compression			Geometric			Generative edits		
Method	TPR	Ret.	FPR	TPR	Ret.	FPR	TPR	Ret.	FPR
Ours (N=8, M=4)	47.1	93.1	17.2	36.1	65.7	17.6	38.8	79.5	17.8
Ours (N=16, M=8)	69.7	92.2	19.6	45.3	57.2	20.3	60.0	84.2	19.2
Ours (N=32, M=12)	77.8	91.0	19.3	45.4	51.1	20.0	65.0	74.3	20.5
Tree-Ring	100.0	100.0	22.6	46.7	46.7	21.7	60.0	60.0	21.0
Gaussian Shading	100.0	100.0	21.6	27.4	27.4	23.7	37.5	37.5	22.0
SEAL	97.9	97.9	18.7	14.5	14.5	19.2	10.0	10.0	19.5

Table 9: Boundary-aware detector ablation on the held-out ordinary-edit evaluation set at 20% nominal clean FPR. Values are unweighted means over the 20 reported transformation settings (14 appearance/compression and six geometric). TPR and FPR are measured after transformation, and retention is conditioned on original detection. Exact-code detection uses no boundary expansion; boundary-aware detection uses the settings in Table 5. Each pair uses the same images.

(N, M)	Detector	TPR	Retention	FPR
$(8, 4)$	Exact code	43.49	82.64	16.75
	Boundary aware	43.81	84.86	17.33
$(16, 8)$	Exact code	59.50	72.49	19.18
	Boundary aware	62.41	81.72	19.81
$(32, 12)$	Exact code	59.70	65.80	17.89
	Boundary aware	**68.12**	79.01	19.53

Best / **Second**

Edited TPR (%)

Edit	DINOv2 pretrain	Ours fine-tune	Ours $(8, 4)$	Ours $(16, 8)$	Ours $(32, 12)$	Tree-Ring	Gaussian Shading	SEAL
Appearance and compression								
JPEG Q90	×	✓	49.1	74.3	84.0	**100.0**	**100.0**	**99.5**
JPEG Q75	×	✓	47.9	72.9	82.3	**100.0**	**100.0**	**99.5**
Gaussian blur $r = 0.5$	✓	✓	49.4	74.3	**84.5**	**100.0**	**100.0**	**100.0**
Gaussian blur $r = 1.5$	✓	✓	46.6	69.5	**77.1**	**100.0**	**100.0**	**100.0**
Gaussian noise $\sigma = 2$	×	✓	48.6	73.9	**83.5**	**100.0**	**100.0**	**100.0**
Gaussian noise $\sigma = 6$	×	✓	46.4	69.1	77.5	**100.0**	**100.0**	**99.5**
Brightness $\times 0.8$	✓	✓	49.1	74.1	**83.7**	**100.0**	**100.0**	**100.0**
Brightness $\times 1.2$	✓	✓	47.7	72.0	**80.3**	**100.0**	**100.0**	**100.0**
Contrast $\times 0.8$	✓	✓	49.0	73.9	**83.9**	**100.0**	**100.0**	**100.0**
Contrast $\times 1.2$	✓	✓	48.1	72.1	**80.9**	**100.0**	**100.0**	**100.0**
Median 3×3	×	×	48.2	72.8	**81.9**	**100.0**	**100.0**	**100.0**
Median 5×5	×	×	45.0	65.0	**70.8**	**100.0**	**100.0**	**100.0**
Salt-and-pepper 1%	×	×	43.7	59.5	64.1	**100.0**	**100.0**	**91.5**
Salt-and-pepper 3%	×	×	40.7	52.9	55.1	**100.0**	**100.0**	**80.5**
Geometric transformations								
Center crop 50%	✓	×	**29.5**	**30.7**	29.1	18.5	20.5	14.0
Center crop 75%	✓	×	36.9	**45.8**	45.6	**54.5**	32.5	9.0
Off-center crop 75%	✓	×	31.4	**33.0**	30.0	**38.5**	23.5	7.0
Off-center crop 90%	✓	×	38.5	**51.1**	**51.1**	**47.5**	41.0	18.5
Shift 32 px	×	✓	37.8	**50.3**	**51.4**	21.0	23.5	30.0
Shift 8 px	×	✓	42.8	61.3	**65.3**	**100.0**	23.5	8.5
Edits from an image-editing model								
Attribute change	×	×	50.0	55.0	**75.0**	**65.0**	25.0	10.0
Background cleanup	×	×	35.0	**75.0**	**90.0**	65.0	50.0	15.0
Cat insertion	×	×	35.0	**65.0**	60.0	**75.0**	60.0	10.0
Camera-viewpoint change	×	×	**35.0**	**45.0**	**35.0**	**35.0**	15.0	5.0

Paired retention (%)

Edit	DINOv2 pretrain	Ours fine-tune	Ours $(8, 4)$	Ours $(16, 8)$	Ours $(32, 12)$	Tree-Ring	Gaussian Shading	SEAL
Appearance and compression								
JPEG Q90	×	✓	98.1	99.0	98.6	**100.0**	**100.0**	**99.5**
JPEG Q75	×	✓	95.3	96.8	96.4	**100.0**	**100.0**	**99.5**
Gaussian blur $r = 0.5$	✓	✓	99.1	**99.3**	**99.3**	**100.0**	**100.0**	**100.0**
Gaussian blur $r = 1.5$	✓	✓	91.5	**91.9**	90.1	**100.0**	**100.0**	**100.0**
Gaussian noise $\sigma = 2$	×	✓	96.9	**98.2**	97.9	**100.0**	**100.0**	**100.0**
Gaussian noise $\sigma = 6$	×	✓	91.5	91.0	90.6	**100.0**	**100.0**	**99.5**
Brightness $\times 0.8$	✓	✓	98.0	**98.8**	98.2	**100.0**	**100.0**	**100.0**
Brightness $\times 1.2$	✓	✓	94.7	**95.5**	94.1	**100.0**	**100.0**	**100.0**
Contrast $\times 0.8$	✓	✓	97.9	**98.5**	98.3	**100.0**	**100.0**	**100.0**
Contrast $\times 1.2$	✓	✓	95.5	**95.7**	94.5	**100.0**	**100.0**	**100.0**
Median 3×3	×	×	96.0	**96.4**	96.0	**100.0**	**100.0**	**100.0**
Median 5×5	×	×	**87.5**	85.1	82.3	**100.0**	**100.0**	**100.0**
Salt-and-pepper 1%	×	×	83.9	77.3	74.1	**100.0**	**100.0**	**91.5**
Salt-and-pepper 3%	×	×	76.7	67.8	63.1	**100.0**	**100.0**	**80.5**
Geometric transformations								
Center crop 50%	✓	×	**48.0**	**35.8**	30.8	18.5	20.5	14.0
Center crop 75%	✓	×	**67.8**	**57.8**	51.3	54.5	32.5	9.0
Off-center crop 75%	✓	×	**53.1**	**39.9**	32.5	38.5	23.5	7.0
Off-center crop 90%	✓	×	**72.4**	**65.5**	58.2	47.5	41.0	18.5
Shift 32 px	×	✓	**70.4**	**64.2**	58.6	21.0	23.5	30.0
Shift 8 px	×	✓	**82.9**	79.9	75.3	**100.0**	23.5	8.5
Edits from an image-editing model								
Attribute change	×	×	**83.3**	**84.6**	78.9	65.0	25.0	10.0
Background cleanup	×	×	87.5	**93.8**	**94.7**	65.0	50.0	15.0
Cat insertion	×	×	**87.5**	**100.0**	85.7	75.0	60.0	10.0
Camera-viewpoint change	×	×	**63.6**	**60.0**	38.9	35.0	15.0	5.0

Figure 11: Per-edit detection and paired retention for the 20 selected appearance/compression and geometric settings, followed by four edit families produced by an image-editing model, at 20% nominal clean FPR. The upper section reports edited TPR and the lower section reports paired retention; all values are percentages. The two left columns in both sections mark transformation-family exposure during DINOv2 pretraining and our encoder fine-tuning. Bold green and orange indicate the best and second-best values within each row; ties at the displayed precision share a color. Detector configurations and evaluation protocols are given in Appendix F.1.

Ordinary-edit examples (1/3)

Family	Original	Setting 1	Setting 2
Appearance and compression			
JPEG		 Q90	 Q75
Gaussian blur		 $r = 0.5$	 $r = 1.5$
Gaussian noise		 $\sigma = 2$	 $\sigma = 6$
Brightness		 $\times 0.8$	 $\times 1.2$
Contrast		 $\times 0.8$	 $\times 1.2$

Figure 12: Examples of ordinary edits, grouped by family across three pages: appearance and compression, geometric transformations, and edits from an image-editing model. Each row pairs the original image with its edited version or versions; parameter values are shown below the images where applicable.

Ordinary-edit examples (2/3)

Family	Original	Setting 1	Setting 2
Appearance and compression			
Median filter		 3×3	 5×5
Salt-and-pepper		 1%	 3%
Geometric transformations			
Center crop		 50%	 75%
Off-center crop		 75%	 90%
Shift		 8 px per axis	 32 px per axis

Ordinary-edit examples (3/3)

Family	Original	Edited
Edits from an image-editing model		
Attribute change		
Background cleanup		
Object insertion		
Camera-viewpoint change		

G Additional Results

G.1 Encoder Similarity and Watermark Retention

We examine whether transformations that remain close to the original image in our learned representation are also more likely to preserve the watermark. We use the same 20 edit settings and configuration-specific detector settings as the robustness evaluation. For each method and edit setting, we pair the mean original–edited cosine similarity in our embedding with paired detection retention. Detection thresholds are calibrated on clean images and frozen across edits.

For our method, the Spearman correlations are 0.9955, 0.9985, and 0.9955 for $(N, M) = (8, 4)$, $(16, 8)$, and $(32, 12)$, respectively. The corresponding correlations are 0.7471 for Tree-Ring, 0.7572 for Gaussian Shading, and 0.7785 for SEAL. The stronger association for our method is consistent with its design: images that remain close in the encoder space are more likely to retain the same LSH code and clean keyed score. For the baselines, our embedding serves only as an external diagnostic and does not define their watermarking mechanisms.

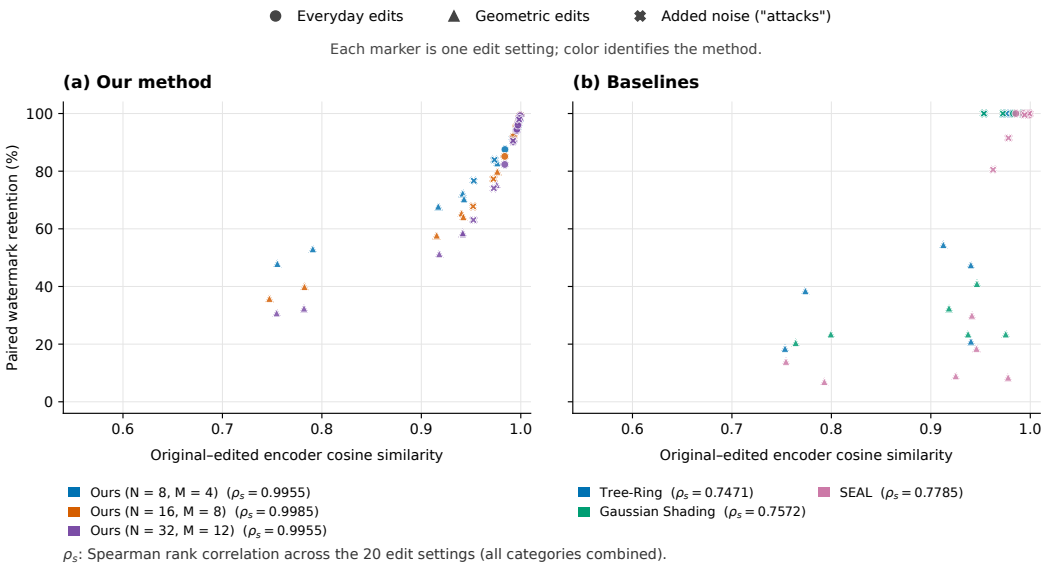

Figure 13: **Encoder similarity and watermark retention.** Each point pairs mean original–edited cosine similarity in our learned embedding with paired detection retention for one of 20 edit settings. Marker shape indicates the edit category, and color indicates the method. Spearman rank correlations are computed separately for each method. Detection thresholds are calibrated on clean images and frozen across edits.

G.2 Semantic Encoder Ablation for Clean Detection

To isolate the effect of the semantic encoder, we evaluate our learned encoder, DINOv2-base, CLIP ViT-B/32, DINOv3 L/16, and TPIPS-Embed (Qwen3VL-8B). Using the same 1,000 held-out COCO-2017 prompts and Stable Diffusion 2.1, we rerun our watermarking scheme and clean-image detection separately for each encoder. We evaluate $(N, M) \in \{(8, 4), (16, 8), (32, 12)\}$ with $B = 1$.

Each encoder uses normalized embeddings in its native dimension. We average over the Cartesian product of five random-hyperplane frames and five PRF keys, giving 25 replicates per encoder and configuration. We use the same configuration-specific detector settings as the main clean-detection evaluation: clean activation targets of 10%, 50%, and 75% for $(N, M) = (8, 4)$, $(16, 8)$, and $(32, 12)$, respectively, with Hamming radius one and five-fold clean-only calibration.

Figure 14 shows that our encoder obtains the highest mean AUROC in all three configurations: 0.736, 0.820, and 0.873. TPIPS-Embed is the strongest baseline, with mean AUROCs of 0.677,

0.765, and 0.840, respectively. At an empirical FPR of at most 0.2, our corresponding mean TPRs are 0.556, 0.740, and 0.853.

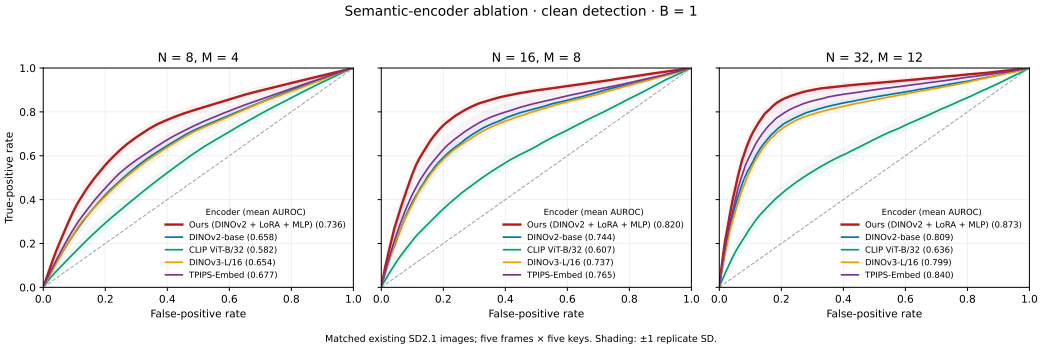

Figure 14: **Semantic-encoder ablation for clean watermark detection.** Mean ROC curves at $B = 1$ for $(N, M) = (8, 4)$, $(16, 8)$, and $(32, 12)$. Using the same 1,000 held-out COCO-2017 prompts and Stable Diffusion 2.1, we rerun our watermarking scheme and clean-image detection separately for each encoder. The dotted diagonal is random guessing (TPR=FPR). Exact detection settings are given in Appendix F.1.

G.3 Closed-API Black-Box Generation

To demonstrate that our sampling-only method can watermark outputs from a closed-API model without access to its internals, we use OpenAI's GPT Image API ("`gpt-image-2.5-sunburst`") strictly as a black-box generator. We use the first 50 prompts from the same held-out COCO-2017 registry. For each prompt, we independently request eight watermark candidates and one clean control at low quality and 1024×1024 resolution, for 450 API generations in total. We use $(N, M) = (8, 8)$, exact-code detection ($r_{ham} = 0$), five independently sampled choices of the hash function H_R, and five PRF keys. Reported curves and metrics average the resulting 25 key-hash repetitions.

GPT Image's candidates are more clustered under our learned encoder than the open-model generations in Table 3: their mean within-prompt cosine similarity is 0.537, compared with 0.222 for SD2.1. GPT Image's eight candidates occupy 5.624 distinct 8-bit codes on average across the 50 prompts and five choices of F, leaving substantial variation for keyed selection. Figure 15 illustrates this behavior for the prompt "Two women walk outside, both holding up umbrellas." Its eight candidates occupy a mean of $6.6/8$ distinct codes across the five choices of F.

For $B = 1$, mean AUROC is 0.769 and mean TPR@20% is 0.613. For $B = 5$, we partition the 50 prompts into ten disjoint groups and average the five scores in each group, using identical prompt membership for the clean and watermarked populations. Aggregation raises mean AUROC to 0.949 and mean TPR@20% to 0.936. Figure 16 shows the complete ROC curves. Because each $B = 5$ repetition contains only ten clean batches, these results demonstrate the benefit of aggregation but do not support a high-confidence low-FPR estimate.



Figure 15: **Eight independent GPT Image generations from one prompt.** All images were returned by separate API calls for "Two women walk outside, both holding up umbrellas." Although their layouts are more similar than in the open-model generation panels, our encoder maps them to a mean of 6.6 distinct 8-bit codes across five independently sampled hash functions.

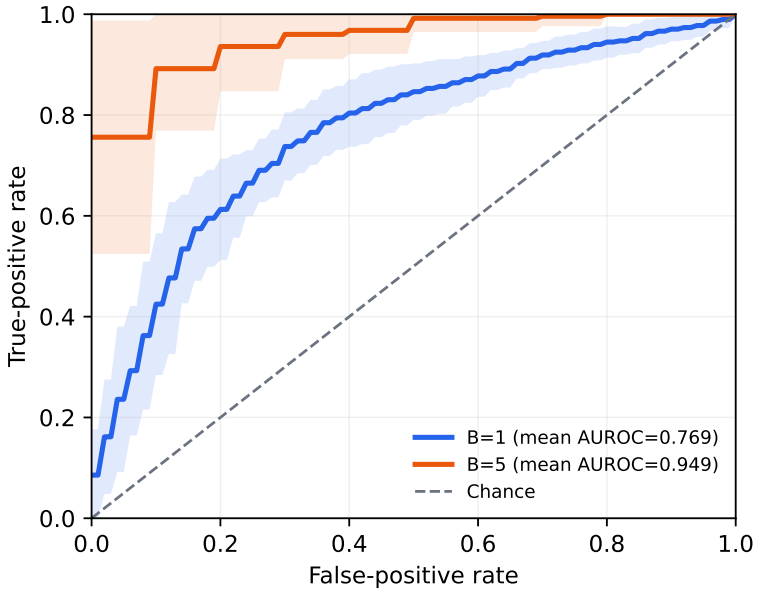

Figure 16: **Closed-API watermark detection.** Mean ROC curves for GPT Image with $(N, M) = (8, 8)$, averaged over five keys and five choices of F. Shading shows one standard deviation across the 25 repetitions. For $B = 5$, each observation averages five watermarked or clean scores with identical prompt membership.

G.4 Generation Quality and Prompt Alignment

To assess generation quality and prompt alignment, we evaluate Stable Diffusion 2.1 Base at a resolution of 512×512. We generate one image for each of 5,000 COCO-2017 prompts and measure FID (Heusel et al., 2017) against 5,000 COCO-2017 validation images. We measure OpenCLIP

similarity (Cherti et al., 2023) separately on 1,000 prompts. For every watermarking configuration, we generate an unwatermarked reference set using the same model, prompts, and sampling settings.

Table 10 reports generation quality and prompt alignment. Across our three configurations, FID improves by 1.28–2.20% relative to matched clean generation, while OpenCLIP similarity changes by at most 0.05%. Tree-Ring and Gaussian Shading change FID by -0.16% and $+0.05\%$, respectively, while SEAL changes it by $+1.78\%$. Relative to their matched clean reference sets, the watermarked populations have comparable FID to real COCO images and nearly unchanged image-prompt similarity. These complementary diagnostics are consistent with the direct matched-population FID results in the main text.

Table 10: Distributional quality on 5,000 unique COCO-2017 validation prompts and image–prompt alignment on 1,000 prompts. Each method is compared with its matched clean control. Relative change is $100(s_{\mathrm{wm}}/s_{\mathrm{clean}} - 1)$. This table supports the quality claim in Section 7.1.

Method	Watermarked	Matched clean	Relative change
FID (lower is better)			
Ours $(N, M) = (8, 4)$	26.598	26.943	-1.28%
Ours $(N, M) = (16, 8)$	26.480	26.943	-1.72%
Ours $(N, M) = (32, 12)$	26.350	26.943	-2.20%
Tree-Ring	24.853	24.894	-0.16%
Gaussian Shading	24.907	24.894	$+0.05\%$
SEAL	25.649	25.201	$+1.78\%$
OpenCLIP image prompt similarity (higher is better)			
Ours $(N, M) = (8, 4)$	0.3154	0.3155	-0.05%
Ours $(N, M) = (16, 8)$	0.3155	0.3155	0.00%
Ours $(N, M) = (32, 12)$	0.3158	0.3158	-0.03%
Tree-Ring	0.3144	0.3145	-0.03%
Gaussian Shading	0.3155	0.3145	$+0.31\%$
SEAL	0.3138	0.3140	-0.06%

H Extended Limitations and Future Directions

This appendix expands on the limitations summarized in the main paper. We discuss single-image detection under limited black-box sampling, computational cost, fixed-key reuse, adaptive attacks, and extension beyond images.

H.1 Single-Image Detection under Limited Black-Box Sampling

Our image implementation has weaker clean single-image detection than methods that insert a signal directly into the diffusion process (Table 2). Aggregating scores across independently generated outputs substantially improves detection, in agreement with Lemma 18. However, aggregation is unavailable when each image must be judged independently.

Consider a detector that accepts an α fraction of clean images, giving false-positive rate α. A stateless sampler drawing N candidates can return a detected image only if at least one candidate falls in the detector's acceptance region. Therefore,

$$\mathrm{TPR} \le 1 - (1 - \alpha)^N \le N\alpha = N\,\mathrm{FPR}\,.$$

Sampling can thus amplify the detection probability over the false-positive rate by at most a factor of N. Achieving constant true-positive rate requires a false-positive rate of order at least $1/N$; matching the exponentially small false-positive rates of methods that directly embed a watermark signal would require exponentially many generator queries. This limitation is independent of the scoring rule or semantic encoder and follows directly from selecting among only N unmodified generator outputs.

This does not contradict the exponential improvement from aggregating independently generated images in Lemma 18: the query limit concerns one sampled output, whereas batch detection accumulates evidence across multiple outputs.

H.2 Computational Cost and Early Candidate Selection

Our strongest configuration generates 32 complete candidates for every released image. This cost may be prohibitive in large-scale or latency-sensitive applications.

Recent work suggests that earlier candidate selection may be feasible for image diffusion models. Guo et al. (2026) show that intermediate denoiser activations can predict final-image quality and allow unpromising trajectories to be terminated early. Habibian et al. (2024) show that low-resolution diffusion features carry semantic layout and can be reused across denoising steps. These results do not establish that our final semantic code can be predicted before generation is complete, but they show that relevant structural and semantic information is available at intermediate stages.

A promising direction is to predict the final semantic code or keyed score from intermediate features, discard unlikely candidates, and complete only promising trajectories. Any such optimization must preserve the selection probabilities required by Theorem 3. Prediction errors may discard trajectories that would have received high final scores, weakening detection even when image quality is unchanged. A useful evaluation must therefore report expected completed denoising steps and wall-clock latency together with detection, image quality, and distribution shift. This would expose the full computation–detection tradeoff rather than treating early termination as a speedup in isolation.

H.3 Distribution Preservation under Fixed-Key Reuse

Theorem 3 establishes equality with the original generator distribution after averaging over the secret key and the sampler's internal randomness. It does not establish distribution preservation separately for every fixed key. A fixed key deliberately favors particular semantic codes, since this dependence creates the detection signal characterized by Theorem 4.

The matched direct-FID experiment shows that our method produces the smallest measured distribution shift among the evaluated methods, while the same-prompt diversity experiment shows that it retains the diversity of matched clean generations (Table 1). These experiments provide evidence that fixed-key reuse preserves quality and diversity in the evaluated setting, but they do not constitute a general per-key guarantee. Establishing fixed-key bounds in terms of the number of candidates, code occupancy, and encoder geometry remains an important theoretical direction.

H.4 Adaptive Removal and Forgery Attacks

Our robustness evaluation considers non-adaptive appearance, compression, geometric, and generative transformations. Figure 2 summarizes the resulting detection and retention rates, while Appendix F.1 reports the individual settings, detector configurations, and achieved false-positive rates.

These experiments do not establish security against an attacker who deliberately targets the encoder or detector. Because our score is bound to the semantic representation of the final image rather than to diffusion-specific latent noise, the method may be less susceptible to reprompting attacks that transfer a watermark to semantically different content, as studied by Müller et al. (2025) and addressed by SemBind (Zhang et al., 2026). This possibility has not been evaluated directly. An adaptive attacker may instead attempt to cross encoder or hash boundaries while preserving apparent content, remove a high-scoring code, or produce altered content that retains the original score. Lemma 20 characterizes non-adversarial behavior under representation similarity and should not be interpreted as a security guarantee against such attacks.

H.5 Extension beyond Images

The framework treats the generator as a sampling oracle and is not tied to a particular architecture or output modality. In principle, the same procedure could be applied to audio, video, 3D content, or other generative domains. Our empirical evaluation, however, is limited to images.

Each new domain requires a semantic encoder that separates independent outputs while remaining stable under the relevant content-preserving transformations. Future work should determine whether such encoders provide sufficient code occupancy for detection, whether their geometry supports robustness to realistic transformations, and whether sampling enough candidates remains computationally practical.