A. Notations

We summarize the key notations and definitions used in this work in Table 6.

Symbol	Description
x	Original image
x^w	Copyrighted or watermarked image
w	Value of a watermark (*e.g.*, bit stream, pattern)
x^*	Attack image
$\mathbf{x}_t$	Latent variable at timestep t
$\hat{\mathbf{x}}_t$	Anchor latent variable at timestep t
$\mathbf{x}_t^*$	Attack latent variable at timestep t
$\hat{\varepsilon}_t$	Predefined margin at timestep t
t	Arbitrary timestep
T	Total timesteps for diffusion process
$\mathcal{G}$	A generative model, *e.g.*, diffusion model
$\mathcal{V}$	Verifier/arbiter for copyright violations
$d(\cdot,\cdot)$	Distance metric
ϵ_θ	U-Net backbone for diffusion model
$\mathcal{E}$	encoder of VAE in stable diffusion
$\mathcal{D}$	decoder of VAE in stable diffusion
τ_θ	CLIP encoder for prompt
$e_\emptyset$	Text embedding for an empty string

Table 6. Notations

B. Challenges in Latent Optimization

From the main draft, we formalize our intuitive objective function as,

$$\min_{\mathbf{x}_T^*} d(\mathbf{x}_0^w, \mathbf{x}_0^*) - \gamma d(\mathbf{x}_T, \mathbf{x}_T^*) \tag{1}$$

Challenge 1. Huge memory consumption. To optimize Eq. 1, the partial derivative $\frac{\partial \mathbf{x}_{t-1}^*}{\partial \mathbf{x}_t^*}$ needs to be calculated for each timestep. For instances of DDPM and DDIM, the gradient is:

$$\frac{\partial \mathbf{x}_0^*}{\partial \mathbf{x}_T^*} = \prod_{t=0}^{T} b_t \left(\mathbf{I} - w_t \cdot \frac{\partial \boldsymbol{\epsilon}_\theta(\mathbf{x}_t^*, t)}{\partial \mathbf{x}_t^*} \right) \tag{2}$$

where $\mathbf{I}$ is the identity matrix and $\frac{\partial \boldsymbol{\epsilon}_\theta(\mathbf{x}_t^*, t)}{\partial \mathbf{x}_t^*}$ is the Jacobian matrix for U-Net model with respect to the input and timestep. The coefficients $(b_t, w_t) = \left(\sqrt{\alpha}_t, \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\right)$ for DDPM and $\left(\sqrt{\frac{\alpha_{t-1}}{\alpha_t}}, \sqrt{1-\alpha_t}\right)$ for DDIM.

Computing these Jacobian matrices is highly memory-intensive. In practice, processing a single timestep for one image requires roughly 10 GB of GPU memory. Even a DPM solver [8] that generates images in only 10 steps, demands over 100 GB, far exceeding the Hopper-100's maximum capacity of 80 GB.

Challenge 2. Gradient estimations would result in over-smoothing images. To reduce the memory consumption, some researchers skip the several sampling steps via estimating gradients [12]. Particularly, for a small timestep t (see Eq.15 in [6] for detailed derivation and empirical results),

$$\mathbf{x}_0 \approx \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) \tag{3}$$

With above "shortcut", the gradient computation between an arbitrary timestep and timestep 0 can be replaced with a single step. However, this shortcut results in significant quality degradation, producing over-smoothing images as illustrated in Figure 3.

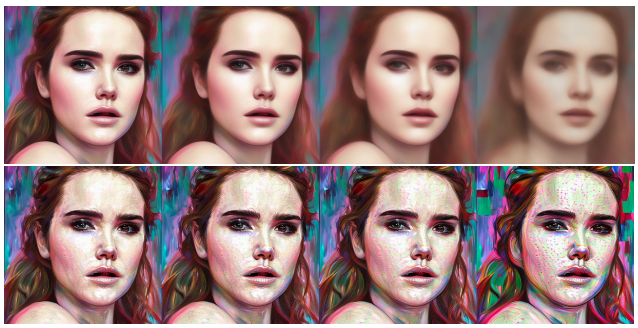

Figure 3. Preliminary: over-smoothing images and noisy images.

Challenge 3. Inducing Semantic Modifications. The core challenge in this optimization is to find correct direction that leads to meaningful semantic changes in images instead of noise. Conventional adversarial attacks [4, 9, 11] operate in image space, where imperceptible pixel-level changes are sufficient to manipulate predicted labels. But this approach will not work when it comes to copyrighted images, where the content of the images are being protected. Though perceptual distances [13] provide a means to measure semantic differences, they are not directly applicable to the optimization, particularly in latent space.

On the other hand, image interpolation and editing [1, 2, 10] using diffusion models can make smooth and controllable semantic modifications. But these methods require either additional reference images or explicit editing instructions, making them inapplicable to unsupervised search settings, not to mention that their ability to evade watermarking remains unexplored.

As illustrated in Fig. 3, our preliminary results show that performing adversarial search directly in latent space often leads to noisy images with little to no meaningful semantic modifications. This highlights the challenge of defining an effective semantic search space, as existing methods fail to navigate latent space in a way that preserves semantic integrity when forging replicas.

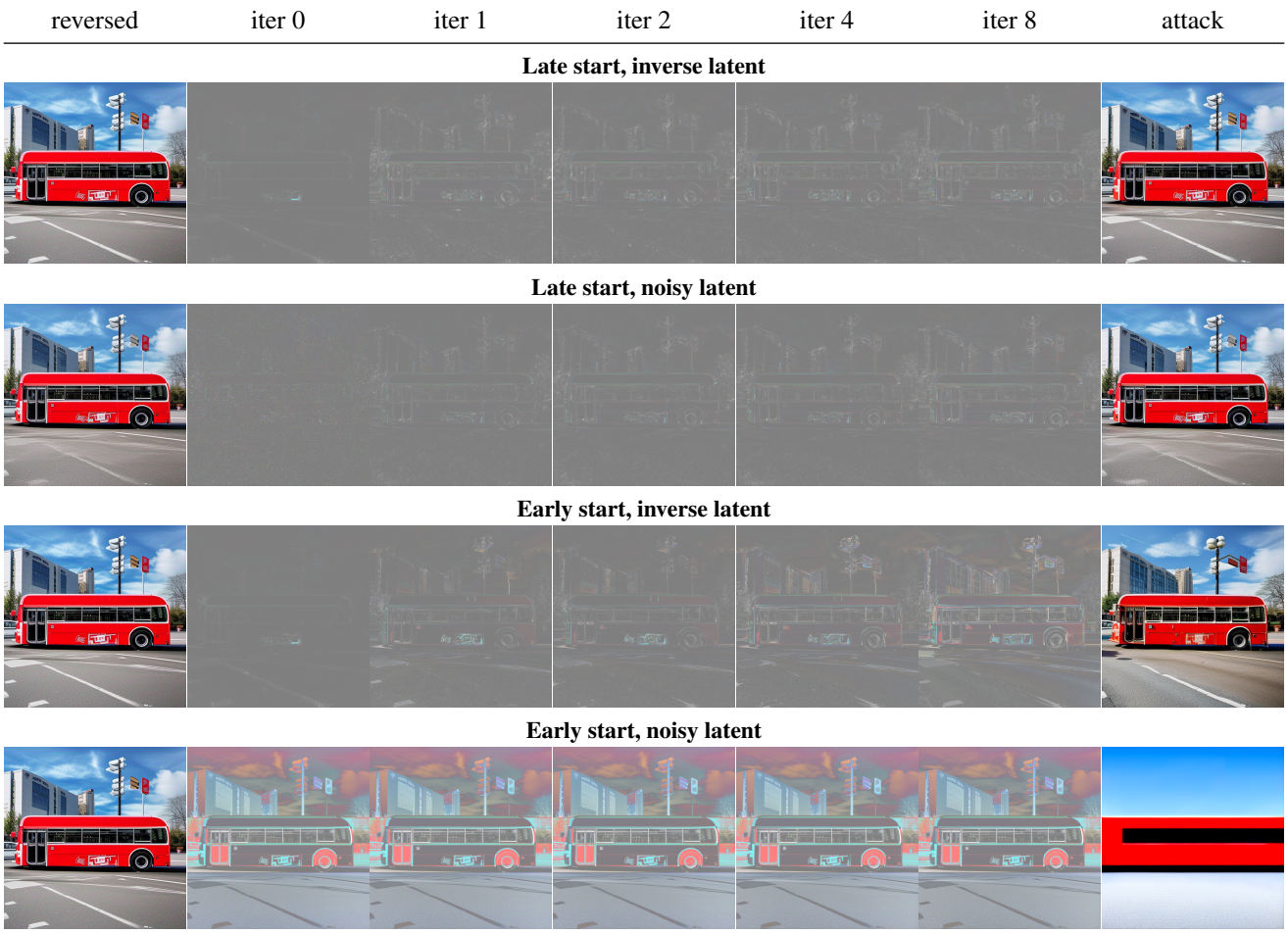

Table 7. The middle columns highlight the pixel differences between the target image (leftmost) and the attack image (rightmost) during optimization. These illustrate how the latent gradually diverges from the anchor latent in a seamless and semantically meaningful manner.

C. Inversion and Watermark Preservation

C.1. The Inversion Formula

We refer to the interpolation technique [3] (Section F) to inverse latent variable. The formula is,

$$x_{t+1} - x_t = \sqrt{\bar{\alpha}_{t+1}}\Big[\left(\sqrt{1/\bar{\alpha}_t} - \sqrt{1/\bar{\alpha}_{t+1}}\right) x_t + \left(\sqrt{1/\bar{\alpha}_{t+1} - 1} - \sqrt{1/\bar{\alpha}_t - 1}\right) \epsilon_\theta(x_t)\Big].$$

C.2. Exact Inversion is Not Necessary

We invert the latent variable from timestep 1 to T using the above formula and record the results as *anchors*. These anchors guide further optimization, so it is desirable for them to retain as much watermark information as possible. To evaluate this, we reconstructed images from inverse latents computed at different timesteps and measured the watermark detection rate. Our experiments show that images reconstructed from later timesteps (*i.e.*, when t is small) tend to have higher watermark detection rates, indicating better watermark preservation.

We then investigated whether exact inversion techniques for diffusion models [7], fine-tuning the VAE with an MSE loss, or applying regression to obtain a more accurate latent $\mathbf{x}_0$ could improve watermark preservation. However, these methods require significantly more processing time and do not enhance the watermark detection rate. In conclusion, exact inversion is not essential for watermark removal, and using anchors from the basic inversion process is sufficient.

D. Invisible Watermark Removal

D.1. Noisy vs. Inverse Latents, Early vs. Late Start

We evaluate our attack pipeline against invisible watermarks by testing all combinations of noisy versus inverse latents and early versus late start conditions. Table 7 visualize the semantic changes induced by the attack pipeline.

- **Late Start, Inverse Latent ✗:** Starting at a later timestep

(i.e., $t < 140$) results in fewer inversion and denoising steps. Directly denoising the inverse latent reduces global semantic changes, as these are already addressed in earlier timesteps. For example, in iterations 0 and 8, the semantic changes mainly emphasize the advertisement area on the bus.

- **Late Start, Noisy Latent ✓:** In contrast, beginning with a noisy latent proves beneficial for removing invisible watermarks. As shown in iteration 0, noise is uniformly distributed across the image, forcing semantic changes at all pixel locations. Moreover, because anchors record the inverse latent of the watermarked images, the attack latents diverge further from the anchors, thereby improving watermark removal performance.
- **Early Start, inverse latent ✓:** Starting at earlier timesteps allows the attacker to induce significant semantic changes, enabling the creation of plagiarized content even from renowned artworks and photographs (*e.g.*, Van Gogh's paintings).
- **Early Start, noisy latent ✗:** The noise addition process cannot apply to early start where timestep are pretty large ($t > 500$). This is because when the timestep is large, the noisy latent is almost Gaussian. Such that it is not useful to extract semantic information as query in attention units. As a result, the attack images will collapse to simple pattern, significantly deviate the original anchor and result in meaningless outputs.

E. Failed Attempts

Besides, we list a few other failed attempts for invisible watermark removal.

E.1. Artifacts from Inverse Images

As shown in Figure 4, we observed that attack images on DctDwtSvd-watermarked images [5] tend to exhibit vertical line artifacts when the inversion is performed at larger timesteps. The DctDwtSvd method embeds watermarks by first decomposing the image using discrete wavelet transform (DWT) and then applying the discrete cosine transform (DCT) on a selected sub-band. Singular value decomposition (SVD) is then used to modify the singular values according to the watermark bits. These modifications alter key frequency components, which are critical to preserving the image structure. When the inversion process uses larger timesteps (inverse from 1 to a large timestep), the impact of these frequency alterations is amplified, leading to visible artifacts such as vertical lines. This issue can be mitigated by choosing smaller timesteps, while the artifacts themselves may serve as an indicator of the underlying watermarking method.

Figure 4. Images reconstructed by inverse latent from DctDwtSvd watermarked images contain vertical line artifacts.

E.2. Add Shims on Latent Variable

In the main draft, we discuss finding alternative query, key and value (Q', K', V') that align with the original outputs. This suggests that shims can be added to latent variables only, text embeddings only, or both. However, our experiments show that adding shims to latent variables often introduces noise unless the hyper-parameters $\hat{\varepsilon}_t$ are very carefully tuned. Therefore, in this work, we add shims only to the text embeddings (*i.e.* (Q, K', V')).

E.3. Numerical Problem

Optimizing latent variables in our pipeline requires the use of single precision (`float32`). To ensure optimal generation quality, we consistently use `float32` for all our experiments.

F. LLM Arbiter

Since GPT4-o has updated their policy not to answer IP-related questions, we add *quantitative studies* with Qwen3-235B-A22B in Table 8. This result shows how our approach can increase success rate by introducing more semantic meaningful alterations at larger timestep. Please also note this success rate might be imprecise and subjective as: **(1)** vision language models were not trained to identify data plagiarism; **(2)** justification is difficult due to blurring line of data plagiarism in legal definition and the reasoning process with domain knowledge. For example, Qwen classified most 'white' replicas as Elon, even when the facial features, such as cheek, eyes, and ears, are so different that Elon himself would need surgery to resemble them, shown in Figure 5 and *Elon100*.

Target	t=500	t=700	t=900
Elon	16% (0.02)	32% (0.04)	59% (0.29)
Elsa	0% (0.06)	5% (0.12)	45% (0.21)
Monet	100% (0.15)	75% (0.26)	83% (0.34)

Table 8. Avg **Success rate(Lpips)** for 100 images. Disney IP is the most complex one. Prompts and Qwen responses will be provided.

Figure 5. Disagreement when Qwen: Yes and human: No.

G. More Attack Images

G.1. Portrait Replicas for Elon Musk

To demonstrate the potential negative impact of neural plagiarism on copyright protection, we generated 100 portrait replicas of Elon Musk using different random seeds. We then employed a GPT-based verifier to assess whether these replicas could be accurately identified as representations of Musk. As illustrated in Figure 6, all the generated portraits bear a strong resemblance to Musk, leading the GPT verifier to fail in recognizing them.

References

[1] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18208–18218, 2022. 1

[2] Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. Diffedit: Diffusion-based semantic image editing with mask guidance. *arXiv preprint arXiv:2210.11427*, 2022. 1

[3] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *Advances in Neural Information Processing Systems*, pages 8780–8794. Curran Associates, Inc., 2021. 2

[4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 1

[5] Mohamed Hamidi, Mohamed El Haziti, Hocine Cherifi, and Mohammed El Hassouni. A hybrid robust image watermarking method based on dwt-dct and sift for copyright protection. *Journal of Imaging*, 7(10):218, 2021. 3

[6] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 1

[7] Seongmin Hong, Kyeonghyun Lee, Suh Yoon Jeon, Hyewon Bae, and Se Young Chun. On exact inversion of dpm-solvers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7069–7078, 2024. 2

[8] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022. 1

[9] Aleksander Madry. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 1

[10] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021. 1

[11] C Szegedy. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1

[12] Ruochen Wang, Ting Liu, Cho-Jui Hsieh, and Boqing Gong. On discrete prompt optimization for diffusion models. In *International Conference on Machine Learning (ICML)*, 2024. 1

[13] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 1

Figure 6. Who is Elon musk? Diffusion models can easily produce 100+ replicates in a few minutes.