VTEGUARD:PROACTIVE DEFENSE AGAINST DIFFUSION-BASED VISUAL TEXT EDITING

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent diffusion models enable realistic visual text editing but also expose publicly shared text-containing images to unauthorized modification or malicious text tampering. To mitigate this risk, proactive defenses seek to protect images before release while preserving their appearance. However, existing defenses largely focus on overall visual degradation, often overlooking whether the target text is still successfully rendered. We propose VTEGuard, a proactive defense designed to disrupt unauthorized text editing while preserving the appearance of released images. Specifically, an Entropic-Smoothed Linear (ESL) objective combines global expansion of the variational autoencoder (VAE) posterior variance with adaptive emphasis on low-variance latent dimensions, aiming to destabilize the latent representations used for subsequent editing. To balance this disruption with perturbation imperceptibility, VTEGuard optimizes protective perturbations in the global YCbCr discrete cosine transform (DCT) coefficient space under a bounded perturbation budget. We construct VTEGuardBench to evaluate both image quality and text editing outcomes, including text editing success rate (TESR). VTEGuard outperforms the evaluated baselines across all nine metrics on VTEGuardBench, achieving a TESR of 2% while maintaining a protected-image PSNR of 36.38 dB. Further experiments demonstrate cross-model transferability, applicability to diverse editing and personalization tasks, and robustness under the tested input purification settings.

1 INTRODUCTION

Visual text is a pervasive carrier of information in natural images, posters, advertisements, and public notices. Recent progress (Tuo et al., 2024a; Chen et al., 2024) in diffusion-based visual text editing has made it possible to modify such text while preserving surrounding backgrounds, font styles and colors. In particular, diffusion-based inpainting and image editing models (Zhang et al., 2023; Rombach et al., 2022) have substantially lowered the technical barrier for high-fidelity text manipulation: with only a mask or a text prompt, non-expert users can alter words, slogans, or statements in an image while keeping the edited region visually consistent with the original scene. This capability enables useful applications such as poster redesign, document correction, and scene text translation. However, it also introduces serious security risks, including forged receipts, manipulated visual evidence, false announcements, malicious advertisement editing, and unauthorized modification of public or commercial messages.

Existing defenses against visual text tampering remain insufficient for this emerging editing paradigm. Most prior efforts focus on passive detection (Wang et al., 2022; Qu et al., 2025), which identifies manipulation after it occurs but cannot prevent the generation of tampered images. Its reliability may also decline as editing models produce fewer detectable artifacts. Text's Armor (Xiang et al., 2022) explores proactive protection for scene text editing but it is mainly designed around specific scene text editing pipelines (Wu et al., 2019) and is therefore less suited to the more open and flexible setting of diffusion-based visual text editing.

Proactive defenses for diffusion models use adversarial perturbations to disrupt image encoding or denoising (Liang et al., 2023; Li et al., 2024; Xue et al., 2024). However, these methods are primarily designed for face editing, deepfake prevention, artistic style protection, or general image editing.

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.

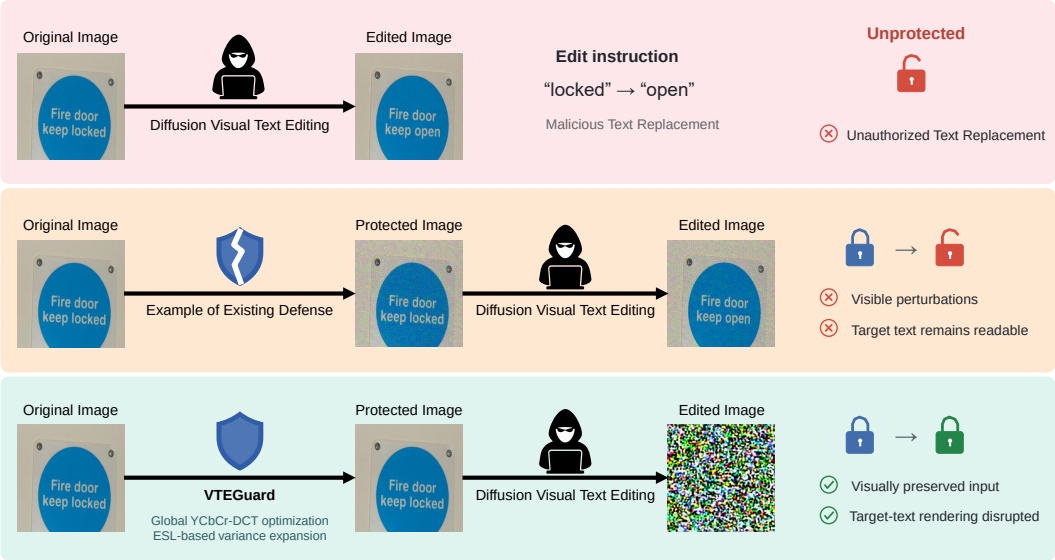


Figure 1: Overview of VTEGuard. Imperceptible perturbations protect images before release, disrupting unauthorized diffusion-based text edits while preserving image fidelity.

Visual text editing poses a task-specific security challenge: defenses must disrupt text correctness and legibility, as visually degraded outputs may still convey the intended unauthorized message.

Perturbation imperceptibility and robustness pose additional challenges. Pixel-space perturbations can introduce visible artifacts around text strokes or smooth backgrounds where even slight artifacts may affect perceptual quality. Moreover, most of these methods are not robust to purification techniques like JPEG compression. Although prior work (Bala et al., 2025) leverages block-wise discrete cosine transform (DCT) operations within the JPEG pipeline to improve perturbation robustness, this design remains inherently tied to JPEG compression and may introduce visible 8×8 blocking artifacts.

In this paper, we propose **VTEGuard**, a proactive defense framework for diffusion-based visual text editing. As illustrated in Figure 1, VTEGuard protects images before release by optimizing perturbations in the global YCbCr-DCT coefficient space. This parameterization enables frequency-domain optimization without partitioning the image into independent DCT blocks. We further introduce an **Entropic-Smoothed Linear** (ESL) loss based on VAE latent variance to disrupt latent representations and hinder subsequent text editing.

To facilitate evaluation, we construct **VTEGuardBench** to evaluate perturbation imperceptibility and protection effectiveness, including text editing success rate (TESR). Experiments show that VTEGuard achieves the best results across all reported metrics, with a TESR of 2%. Further evaluations demonstrate cross-model transferability, applicability to additional editing and personalization tasks, and robustness under the tested purification settings.

Our contributions are summarized as follows:

- We propose **VTEGuard**, a proactive defense framework against unauthorized diffusion-based visual text editing. By protecting images before release, VTEGuard disrupts the correctness and legibility of edited text while preserving the appearance of the original images.
- We develop a **global YCbCr-DCT perturbation optimization** strategy and an **Entropic-Smoothed Linear (ESL) loss** based on VAE latent variance. Together, they enable frequency-domain perturbation design and latent-space disruption, balancing protection effectiveness with perturbation imperceptibility.
- We construct **VTEGuardBench** for systematic evaluation of perturbation imperceptibility and protection effectiveness, including text editing success rate. Extensive experiments demonstrate VTEGuard's advantages over existing baselines, cross-model transferability,

applicability across editing and personalization tasks, and robustness under the tested input purification settings.

2 Related work

2.1 Visual Text Editing with Latent Diffusion Models

Diffusion models (Ho et al., 2020), particularly latent diffusion models (LDMs) (Rombach et al., 2022), provide strong generative priors for image synthesis and editing. Visual text editing builds on these capabilities to replace textual content while preserving typography and background appearance. Given a latent representation z_0, the forward diffusion process gradually perturbs it with Gaussian noise:

$$z_t = \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I),$$

where $\bar{\alpha}_t$ denotes the cumulative noise schedule. A denoising network ϵ_ω is trained to predict the added noise at each timestep, conditioned on editing information c:

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

For visual text editing, the condition c is typically composed of target text, glyph maps, text masks, bounding boxes, layout information, style references, or natural-language editing instructions. Existing methods explore complementary forms of text-specific guidance. DiffSTE (Ji et al., 2023) introduces a dual-encoder design that combines character encoding for text legibility with instruction encoding for style control. DiffUTE (Chen et al., 2023) uses a pretrained OCR model to encode glyph images, providing structural guidance for text editing. TextDiffuser-2 (Chen et al., 2024) incorporates LLM-based layout planning and line-level text-position encoding to control textual content and placement. AnyText and AnyText2 (Tuo et al., 2024b;a) support multilingual text generation and editing through glyph, position, and other text-specific conditioning, with AnyText2 further enabling text attribute control. Focusing on style preservation, TextCtrl (Zeng et al., 2024) introduces style-structure guidance and glyph-adaptive attention. These advances improve the accuracy and visual consistency of text editing, but also facilitate convincing unauthorized modifications. This motivates proactive protection that disrupts subsequent text editing while preserving the appearance of released images.

2.2 Adversarial Attacks for Latent Diffusion Models

Adversarial attacks on latent diffusion models (LDMs) have been widely explored for proactive image protection. By adding imperceptible perturbations to source images, these methods seek to disrupt unauthorized editing, imitation, or personalization. Early efforts mainly focus on disrupting the UNet-based denoising process. Some methods (Choi et al., 2025; Liang et al., 2023; Van Le et al., 2023; Xue et al., 2024) maximize the diffusion training objective or the noise prediction error, thereby interfering with the sampling trajectory. Beyond the denoising network, another line of work (Li et al., 2024; Liang & Wu, 2023; Salman et al., 2023; Yang et al., 2025; Bala et al., 2025) attacks the encoder of LDMs by manipulating latent representations through target alignment, feature displacement, or latent distribution objectives. These perturbations consequently degrade the downstream generation process. In parallel, several works (Lo et al., 2024; Jeon et al., 2025) shift attention to the text-conditioning or cross-attention modules, aiming to weaken the correspondence between prompt tokens and semantically relevant image regions and consequently undermine the model's text-image alignment capability. However, these methods primarily address general image manipulation or appearance imitation, without explicitly accounting for the correctness and legibility of edited text. Visually degraded outputs may still convey the intended unauthorized message, making image-level degradation alone insufficient to establish text protection.

3 Main method

3.1 Threat Model and Problem Formulation

We consider a malicious adversary who uses publicly available LDMs to manipulate text in online scene images while preserving their typography and background. Before release, we protect an image

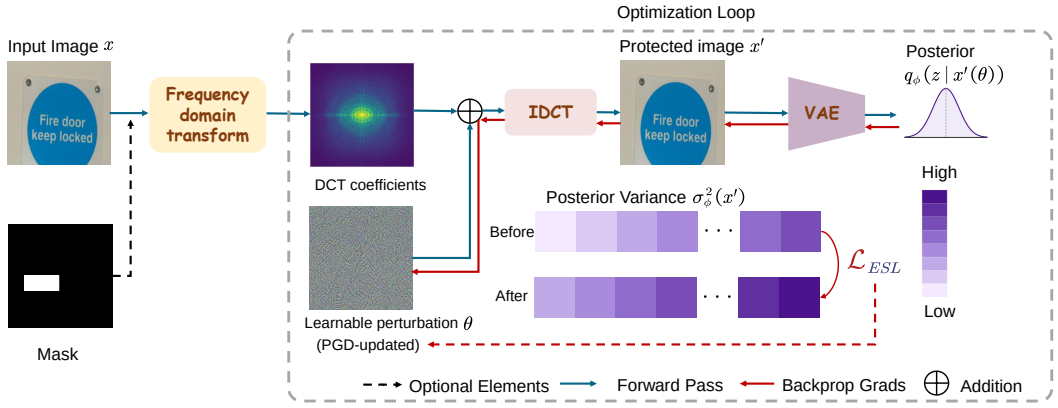


Figure 2: Overview of VTEGuard. A learnable perturbation is added to the input image's global YCbCr-DCT coefficients, followed by an inverse transform to reconstruct the protected image. The ESL loss promotes global VAE posterior-variance expansion with adaptive emphasis on low-variance dimensions.

$\mathbf{x}$ by adding an imperceptible perturbation $\boldsymbol{\delta}$:

$$\mathbf{x}_{\text{adv}} = \mathbf{x} + \boldsymbol{\delta}. \tag{1}$$

Our goal is to preserve input appearance while disrupting the semantic correctness and legibility of the edited text. Let $\mathcal{E}_\psi(\mathbf{x}, \mathbf{c})$ denote a visual text editing model conditioned on target text or an instruction $\mathbf{c}$. We formulate protection as maximizing the discrepancy between protected and clean editing outputs under a perturbation budget:

$$\begin{aligned} \boldsymbol{\delta}^* = \arg\max_{\boldsymbol{\delta}} \quad & \mathcal{D}_{\text{edit}}\left(\mathcal{E}_\psi(\mathbf{x}+\boldsymbol{\delta}, \mathbf{c}), \mathcal{E}_\psi(\mathbf{x}, \mathbf{c})\right) \\ \text{s.t.} \quad & \|\boldsymbol{\delta}\|_\infty \le \epsilon, \quad \mathbf{x}+\boldsymbol{\delta} \in [0,1]^d, \end{aligned} \tag{2}$$

where $\mathcal{D}_{\text{edit}}$ measures output discrepancy and ϵ is the perturbation budget.

3.2 Full-Image Frequency-Domain Parameterization

Most existing image immunization methods optimize perturbations directly in RGB pixel space. Such perturbations are often noticeable when zoomed in and can be attenuated by common purification operations, including JPEG compression, resizing, and denoising. We instead optimize the perturbation in the full-image YCbCr-DCT domain to obtain a globally structured frequency representation.

Given a clean RGB image $x \in \mathbb{R}^{H\times W\times 3}$, we first convert it to centered YCbCr space and apply a full-image 2D-DCT independently to each channel:

$$F = \mathcal{D}(\mathcal{C}(x) - \mathbf{b}_{\text{col}}), \tag{3}$$

where $\mathcal{C}$ denotes RGB-to-YCbCr conversion, $\mathbf{b}_{\text{col}}$ is the channel-wise centering offset, and $\mathcal{D}$ is the full-image DCT operator.

We introduce a learnable frequency-domain perturbation θ and construct the perturbed coefficients as

$$F' = F + M_f \odot \theta, \tag{4}$$

where M_f optionally restricts the active frequency components. The protected image is reconstructed as

$$x'(\theta) = \Pi_{\mathcal{B}_\epsilon(x)}\left[\mathcal{C}^{-1}\left(\mathcal{D}^{-1}(F') + \mathbf{b}_{\text{col}}\right)\right], \tag{5}$$

where $\mathcal{B}_\epsilon(x) = \{u \in [0,1]^{H\times W\times 3} : \|u - x\|_\infty \le \epsilon\}$. During optimization, θ is projected onto a bounded feasible set to control perceptual distortion. Unlike block-wise JPEG DCT (Bala et al., 2025), our transformation is applied over the entire spatial extent of each channel. It therefore represents the perturbation using global frequency bases and avoids discontinuities associated with independently modified 8×8 blocks. The resulting parameterization is differentiable almost everywhere, allowing the latent-space objective introduced below to directly guide the frequency coefficients.

3.3 Cross-Domain Gradient Propagation to VAE Latent Variance

LDM-based visual text editors do not directly manipulate input pixels. Instead, they first encode the input image into a latent distribution through a VAE, whose posterior representation provides the semantic and structural basis for subsequent editing. Prior studies (Li et al., 2024; Yang et al., 2025) have shown that perturbing the VAE posterior variance can induce substantial generation distortion and semantic erasure, indicating that posterior variance is a critical factor governing the stability of latent representations. This observation motivates us to target the model's latent encoding process rather than merely enlarging pixel-level discrepancies or introducing local texture corruption. Since perturbations are parameterized in the YCbCr-DCT domain, optimizing this latent-space objective requires backpropagation through the VAE encoder and inverse transform to the frequency coefficients. Given the protected image $x'(\theta)$, the surrogate VAE encoder produces the posterior distribution

$$\begin{aligned} q_\phi(z \mid x'(\theta)) &= \mathcal{N}\left(\mu_\phi(x'(\theta)), \mathrm{diag}\big(v_\phi(x'(\theta))\big)\right), \\ v_\phi(x') &= \sigma_\phi^2(x'), \end{aligned} \tag{6}$$

where v_ϕ denotes the vectorized posterior variance. Our objective $L(v_\phi)$ is defined directly over these variance components.

Let

$$P_\theta = \frac{\partial x'}{\partial \theta}, \qquad J_\phi(x') = \frac{\partial v_\phi}{\partial x'} \tag{7}$$

denote the Jacobians of the inverse frequency transformation and the VAE encoder, respectively. Applying the chain rule gives

$$\nabla_\theta L = P_\theta^\top J_\phi(x')^\top \nabla_{v_\phi} L = G(x')\, \nabla_{v_\phi} L, \tag{8}$$

where

$$G(x') = P_\theta^\top J_\phi(x')^\top \tag{9}$$

is the local cross-domain gradient transmission operator.

The transform Jacobian P_θ is determined by the inverse full-image DCT and the color-space conversion. It is constant in the unclipped regime. In contrast, $J_\phi(x')$ is input-dependent because the VAE encoder is nonlinear and is evaluated locally at the current protected image.

For the i-th frequency coefficient, equation 8 can be expressed as

$$\frac{\partial L}{\partial \theta_i} = \sum_j G_{ij}(x') \frac{\partial L}{\partial v_{\phi,j}}. \tag{10}$$

Thus, each DCT coefficient is updated according to a weighted combination of gradients from all latent variance components. The frequency-domain update is therefore governed not only by the transform and encoder Jacobians, but also by how the objective distributes gradients across the latent dimensions.

3.4 Entropic-Smoothed Linear Loss

As discussed in Sec. 3.3, the frequency-domain gradient is obtained by projecting the variance-level gradient through a local cross-domain transmission operator. A sparse or saturated variance gradient may therefore yield an imbalanced or weak update after this projection. To provide a persistent global signal while emphasizing insufficiently disrupted latent dimensions, we propose the Entropic-Smoothed Linear (ESL) loss.

Let $v_j = \sigma_j^2$ denote the j-th component of the vectorized VAE posterior variance. We maximize

$$L_{\mathrm{ESL}}(v) = a \sum_j v_j - b\tau \log \sum_j \exp\left(-\frac{v_j}{\tau}\right), \tag{11}$$

where $a, b > 0$ control the global and adaptive terms, respectively, and $\tau > 0$ is a temperature parameter. The linear term expands all variance components, whereas the second term is an entropy-smoothed approximation to the minimum variance. Maximizing the latter therefore preferentially increases variance components that remain relatively small.

The variance-level gradient has a particularly simple form:

$$\frac{\partial L_{\mathrm{ESL}}}{\partial v_j} = a + b w_j, \qquad w_j = \frac{\exp(-v_j/\tau)}{\sum_k \exp(-v_k/\tau)}. \tag{12}$$

Because w_j is larger for smaller v_j, ESL assigns additional weight to under-disrupted variance components. Meanwhile,

$$\frac{\partial L_{\mathrm{ESL}}}{\partial v_j} \geq a > 0, \tag{13}$$

so every component retains a non-saturating base gradient. The temperature τ controls the strength of bottleneck mining: a smaller τ concentrates the adaptive term on the lowest-variance components, while a larger τ distributes it more smoothly across the latent dimensions.

Combining equation 12 with the cross-domain propagation in Sec. 3.3 gives

$$\nabla_\theta L_{\mathrm{ESL}} = G(x') \left(a\mathbf{1} + b\mathbf{w}\right), \tag{14}$$

where $G(x')$ is the local transmission operator from the VAE variance space to the full-image DCT parameter space. The uniform component $a\mathbf{1}$ provides persistent global variance expansion, while $b\mathbf{w}$ adaptively amplifies latent dimensions that have not yet been sufficiently disrupted. ESL thus supplies a dense, non-saturating, and balanced signal before cross-domain projection, enabling the structured frequency perturbation to more effectively impair the readability, semantic correctness, and style consistency of edited text.

4 Experiments

4.1 Experimental Setup

Datasets and Models. We construct **VTEGuardBench**, comprising 100 text-rich images of signs, advertisements, and posters. We optimize perturbations using the VAE encoder of TextDiffuser2 and evaluate visual text editing on TextDiffuser2 and AnyText2. Additional evaluations cover SDEdit (Meng et al., 2022), DiffEdit (Couairon et al., 2023), Stable Diffusion inpainting (Rombach et al., 2022), and DreamBooth personalization (Ruiz et al., 2023). Dataset descriptions and model configurations are provided in Appendices B and D, respectively.

Baselines and Implementation. We compare VTEGuard with AdvDM (Liang et al., 2023), PhotoGuard (Salman et al., 2023), SDS (Xue et al., 2024), AdvPaint (Jeon et al., 2025), BlurGuard (Kim et al., 2025), DCT-Shield (Bala et al., 2025), PID (Li et al., 2024), and LA (Yang et al., 2025). We use an ℓ_∞ perturbation budget of $16/255$ and an image resolution of 512×512. Additional optimization and editing settings are provided in Appendix D.

Metrics. We evaluate protection methods from two perspectives: *perturbation imperceptibility* and *protection effectiveness*. For perturbation imperceptibility, we use PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) between original and protected images, with BRISQUE (Mittal et al., 2012) as a complementary no-reference quality metric. For protection effectiveness, we assess edited outputs using FID (Heusel et al., 2017), LPIPS, CLIP-S (Hessel et al., 2021), and CLIP-I, together with Text Editing Success Rate (TESR), which measures the proportion of outputs that successfully render the target text. The comparison references, feature extractors, and score scaling are specified in Appendix C.

4.2 Main Results on VTEGuardBench

Quantitative Results. VTEGuard achieves the best results across all nine metrics in Table 1. For perturbation imperceptibility, its PSNR reaches 36.380 dB, exceeding AdvPaint by 3.871 dB. Its input LPIPS is 0.117, half that of the runner-up, BlurGuard. For protection effectiveness, VTEGuard reduces TESR from 21% for the strongest baseline, LA, to 2%, a decrease of 19 percentage points. It also increases output LPIPS from 0.893 to 0.952 and reduces CLIP-I from 0.499 to 0.445 relative to LA. These results demonstrate that VTEGuard preserves input quality while disrupting both the visual quality of edited outputs and the successful rendering of target text.

Table 1: Results on VTEGuardBench. Perturbation imperceptibility is evaluated on protected inputs, and protection effectiveness is evaluated on edited outputs. Arrows indicate preferred directions. **Bold** and underlined values denote the best and second-best results, respectively.

Method	Perturbation Imperceptibility				Protection Effectiveness				
	PSNR↑	SSIM↑	LPIPS↓	BRISQUE↓	FID↑	LPIPS↑	CLIP-S↓	CLIP-I↓	TESR↓
AdvDM	26.764	0.560	0.392	35.198	177.660	0.536	0.236	0.796	0.56
PhotoGuard	26.022	0.543	0.384	45.258	221.527	0.559	0.223	0.727	0.35
SDS	26.977	0.540	0.352	51.703	169.588	0.603	0.227	0.767	0.32
AdvPaint	32.509	0.789	0.261	30.506	207.919	0.517	0.227	0.768	0.39
BlurGuard	26.738	0.879	0.234	31.333	93.066	0.228	0.237	0.910	0.69
DCT-Shield	26.459	0.558	0.244	46.774	121.184	0.460	0.233	0.845	0.47
PID	26.634	0.519	0.396	60.045	240.238	0.622	0.225	0.682	0.36
LA	26.722	0.527	0.406	64.624	400.736	0.893	0.190	0.499	0.21
VTEGuard	**36.380**	**0.897**	**0.117**	**27.144**	**481.814**	**0.952**	**0.184**	**0.445**	**0.02**

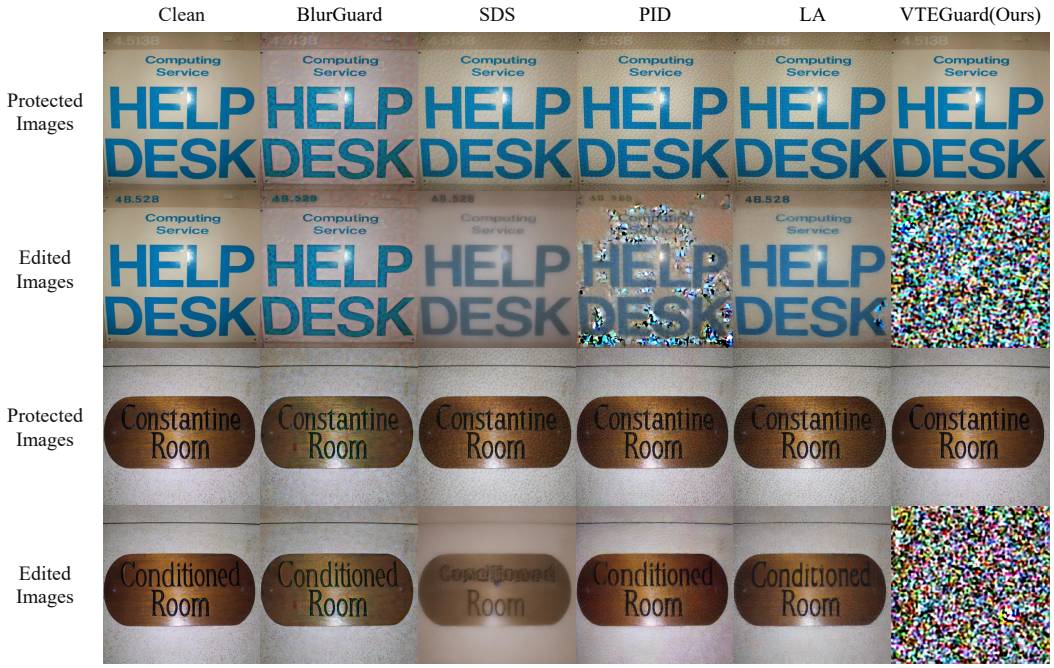


Figure 3: Qualitative comparison on VTEGuardBench, showing protected images and their corresponding text-editing outputs.

Qualitative Results. As shown in Figure 3, VTEGuard preserves the visual appearance of protected inputs while consistently rendering visual text editing outputs as semantically void noise, with no recognizable target text. In contrast, several baselines degrade output quality but leave the target text readable. Additional qualitative results are provided in Appendix E.

4.3 Ablation Studies

Perturbation Parameterization. Table 2 compares RGB pixel-space and YCbCr-DCT optimization under the ℓ_2 and ESL objectives. The ℓ_2 objective is defined in Appendix D. With ℓ_2 fixed, switching to YCbCr-DCT increases protected-image PSNR from 31.844 to 36.829 dB and reduces LPIPS from 0.244 to 0.112, while lowering TESR from 28% to 14%. The same trend holds under ESL, indicating that the YCbCr-DCT parameterization improves perturbation imperceptibility alongside protection effectiveness.

Table 2: Ablation of perturbation parameterization and optimization objective.

Configuration	Perturbation Imperceptibility				Protection Effectiveness				
	PSNR↑	SSIM↑	LPIPS↓	BRISQUE↓	FID↑	LPIPS↑	CLIP-S↓	CLIP-I↓	TESR↓
RGB + ℓ_2	31.844	0.823	0.244	36.774	390.521	0.867	0.197	0.585	0.28
RGB + ESL	31.801	0.815	0.215	33.327	407.921	0.882	0.195	0.543	0.20
YCbCr-DCT + ℓ_2	**36.829**	**0.899**	**0.112**	27.832	410.604	0.894	0.189	0.495	0.14
YCbCr-DCT + ESL	36.380	0.897	0.117	**27.144**	**481.814**	**0.952**	**0.184**	**0.445**	**0.02**

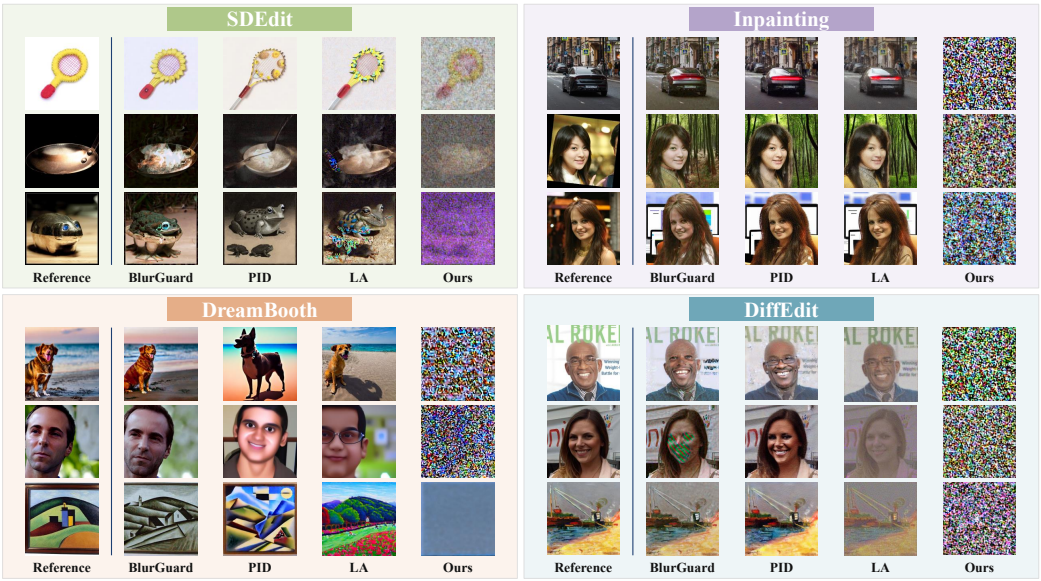


Figure 4: Qualitative comparison across SDEdit, DiffEdit, DreamBooth personalization, and inpainting.

ESL Objective. Within the YCbCr-DCT domain, replacing ℓ_2 with ESL reduces TESR from 14% to 2% and increases output LPIPS from 0.894 to 0.952, demonstrating stronger protection against successful text edits as well as greater visual disruption. These gains come with a modest fidelity trade-off: PSNR decreases by 0.449 dB and input LPIPS increases by 0.005. ESL therefore strengthens protection while largely preserving the imperceptibility benefits of YCbCr-DCT optimization.

4.4 Generalization across Models and Tasks

Cross-Model Transferability. We evaluate perturbations optimized on TextDiffuser2 using AnyText2, without re-optimization. Figure 5(a–b) summarizes protected-image fidelity and editing disruption in this transfer setting. VTEGuard achieves an output FID of 290.521, LPIPS of 0.787, CLIP-S of 0.179, and CLIP-I of 0.543, while maintaining a protected-image PSNR of 36.380 dB. These results support the transferability of its protection beyond the surrogate editing model.

Cross-Task Generalization. We evaluate VTEGuard on SDEdit, DiffEdit, DreamBooth, and inpainting using perturbations crafted on Stable Diffusion v1.5. As shown in Table 3, VTEGuard achieves the best values in 11 of the 12 reported task–metric combinations. Notably, in the inpainting setting, VTEGuard achieves an FID of 434.979, exceeding the runner-up AdvPaint by 28.7%, while reducing CLIP-I from 0.656 to 0.510. These results demonstrate that VTEGuard's protection extends beyond visual text editing to diverse diffusion-based editing and personalization tasks. Qualitative comparisons are provided in Figure 4.

Table 3: Results across four manipulation tasks. Higher FID and lower CLIP-S/CLIP-I indicate stronger disruption under the specified evaluation protocol.

Method	SDEdit			DiffEdit			DreamBooth			Inpainting		
	FID↑	CLIP-S↓	CLIP-I↓	FID↑	CLIP-S↓	CLIP-I↓	FID↑	CLIP-S↓	CLIP-I↓	FID↑	CLIP-S↓	CLIP-I↓
AdvPaint	187.036	0.249	0.771	198.094	0.192	0.626	260.634	0.219	0.647	338.035	**0.164**	0.656
BlurGuard	86.242	0.265	0.893	128.055	0.197	0.732	143.673	0.227	0.848	108.981	0.204	0.845
SDS	191.033	0.258	0.756	197.178	0.213	0.648	192.171	0.228	0.711	138.383	0.210	0.758
PID	191.858	0.241	0.736	219.922	0.203	0.586	251.100	0.233	0.602	130.563	0.205	0.781
LA	229.416	0.194	0.715	223.055	0.184	0.572	294.366	0.214	0.551	145.257	0.196	0.761
VTEGuard	**253.957**	**0.172**	**0.621**	**229.855**	**0.184**	**0.542**	**319.046**	**0.201**	**0.516**	**434.979**	0.166	**0.510**

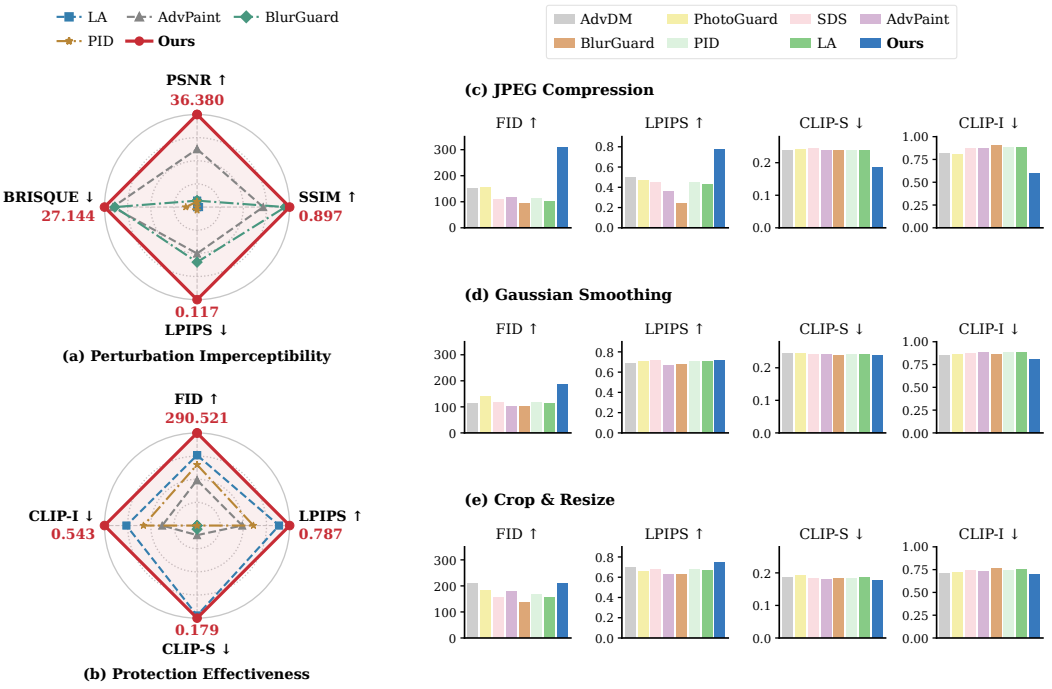


Figure 5: Cross-model transferability and robustness to input purification. (a–b) Transfer from TextDiffuser2 to AnyText2 without re-optimization. (c–e) Robustness under JPEG compression, Gaussian smoothing, and crop-and-resize.

4.5 Robustness against Input Purification

A malicious editor may attempt to remove or weaken proactive perturbations before performing image editing. We apply crop-and-resize, Gaussian smoothing, and JPEG compression before editing. Figure 5(c–e) shows that VTEGuard maintains strong protection after input purification, with a particularly clear advantage under JPEG compression in FID, LPIPS, CLIP-S, and CLIP-I. These results demonstrate consistently superior overall performance across all three robustness scenarios. Full purification settings are provided in Appendix F.

5 Conclusion

We present VTEGuard, a proactive protection framework against unauthorized visual text editing. By combining YCbCr-DCT perturbation optimization with the ESL objective, VTEGuard achieves strong protection while preserving input image quality. Experiments on VTEGuardBench show that VTEGuard outperforms the evaluated baselines across all nine metrics, reducing text editing success to 2%. Further evaluations demonstrate cross-model transferability, applicability across diverse editing and personalization tasks, and robustness under the tested input purification settings.

AI USE STATEMENT

In this work, we used generative AI tools to assist with literature search, polishing the English writing, and LaTeX formatting, including tables, captions, and manuscript layout. All AI-assisted text and literature-related suggestions were independently reviewed and verified by the authors. Generative AI tools were not used to generate experimental results, fabricate data, or make scientific conclusions. The authors take full responsibility for the final content of this work, including any text or artifacts produced with the assistance of generative AI.

REFERENCES

Aniruddha Bala, Rohit Chowdhury, Rohan Jaiswal, and Siddharth Roheda. DCT-Shield: A robust frequency domain defense against malicious image editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 18876–18884, 2025.

Haoxing Chen, Zhuoer Xu, Zhangxuan Gu, Yaohui Li, Changhua Meng, Huijia Zhu, Weiqiang Wang, et al. Diffute: Universal text editing diffusion model. *Advances in Neural Information Processing Systems*, 36:63062–63074, 2023.

Jingye Chen, Yupan Huang, Tengchao Lv, Lei Cui, Qifeng Chen, and Furu Wei. Textdiffuser-2: Unleashing the power of language models for text rendering. In *European Conference on Computer Vision*, pp. 386–402. Springer, 2024.

William June Suk Choi, Kyungmin Lee, Jongheon Jeong, Saining Xie, Jinwoo Shin, and Kimin Lee. Diffusionguard: A robust defense against malicious diffusion-based image editing. In *International Conference on Learning Representations*, volume 2025, pp. 27134–27180, 2025.

Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. DiffEdit: Diffusion-based semantic image editing with mask guidance. In *International Conference on Learning Representations*, 2023.

Cheng Cui, Yubo Zhang, Ting Sun, Xueqing Wang, Hongen Liu, Manhui Lin, Yue Zhang, Tingquan Gao, Changda Zhou, Jiaxuan Liu, Zelun Zhang, Jing Zhang, Jun Zhang, and Yi Liu. PP-OCRv5: A specialized 5m-parameter model rivaling billion-parameter vision-language models on OCR tasks. *arXiv preprint arXiv:2603.24373*, 2026.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7514–7528. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.595.

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.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

Joonsung Jeon, Woo Jae Kim, Suhyeon Ha, Sooel Son, and Sung-eui Yoon. AdvPaint: Protecting images from inpainting manipulation via adversarial attention disruption. In *International Conference on Learning Representations*, 2025.

Jiabao Ji, Guanhua Zhang, Zhaowen Wang, Bairu Hou, Zhifei Zhang, Brian Price, and Shiyu Chang. Improving diffusion models for scene text editing with dual encoders. *arXiv preprint arXiv:2304.05568*, 2023.

Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazan Almazan, and Lluis Pere De Las Heras. Icdar 2013 robust reading competition. In *2013 12th International Conference on Document Analysis and Recognition*, pp. 1484–1493. IEEE, 2013. doi: 10.1109/ICDAR.2013.221.

Jinsu Kim, Yunhun Nam, Minseon Kim, Sangpil Kim, and Jongheon Jeong. BlurGuard: A simple approach for robustifying image protection against AI-powered editing. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Ang Li, Yichuan Mo, Mingjie Li, and Yisen Wang. Pid: prompt-independent data protection against latent diffusion models. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 28421–28447, 2024.

Chumeng Liang and Xiaoyu Wu. Mist: Towards improved adversarial examples for diffusion models. *arXiv preprint arXiv:2305.12683*, 2023.

Chumeng Liang, Xiaoyu Wu, Yang Hua, Jiaru Zhang, Yiming Xue, Tao Song, Zhengui Xue, Ruhui Ma, and Haibing Guan. Adversarial example does good: Preventing painting imitation from diffusion models via adversarial examples. In *International Conference on Machine Learning*, 2023.

Ling Lo, Cheng Yu Yeo, Hong-Han Shuai, and Wen-Huang Cheng. Distraction is all you need: Memory-efficient image immunization against diffusion-based image editing. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2024.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2022.

Anish Mittal, Anush K. Moorthy, and Alan C. Bovik. No-reference image quality assessment in the spatial domain. *IEEE Transactions on Image Processing*, 21(12):4695–4708, 2012. doi: 10.1109/TIP.2012.2214050.

Chenfan Qu, Yiwu Zhong, Fengjun Guo, and Lianwen Jin. Revisiting tampered scene text detection in the era of generative ai. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 694–702, 2025.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2023.

Hadi Salman, Alaa Khaddaj, Guillaume Leclerc, Andrew Ilyas, and Aleksander Madry. Raising the cost of malicious ai-powered image editing. In *International Conference on Machine Learning*, 2023.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

Yuxiang Tuo, Yifeng Geng, and Liefeng Bo. Anytext2: Visual text generation and editing with customizable attributes. *arXiv preprint arXiv:2411.15245*, 2024a.

Yuxiang Tuo, Wangmeng Xiang, Jun-Yan He, Yifeng Geng, and Xuansong Xie. Anytext: Multilingual visual text generation and editing. In *International Conference on Learning Representations*, volume 2024, pp. 56783–56799, 2024b.

Thanh Van Le, Hao Phung, Thuan Hoang Nguyen, Quan Dao, Ngoc N Tran, and Anh Tran. Anti-dreambooth: Protecting users from personalized text-to-image synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2116–2127, 2023.

Yuxin Wang, Hongtao Xie, Mengting Xing, Jing Wang, Shenggao Zhu, and Yongdong Zhang. Detecting tampered scene text in the wild. In *European Conference on Computer Vision*, pp. 215–232. Springer, 2022.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing (TIP)*, 2004.

Liang Wu, Chengquan Zhang, Jiaming Liu, Junyu Han, Jingtuo Liu, Errui Ding, and Xiang Bai. Editing text in the wild. In *Proceedings of the 27th ACM International Conference on Multimedia*. Association for Computing Machinery, 2019. doi: 10.1145/3343031.3350929.

Tao Xiang, Hangcheng Liu, Shangwei Guo, Hantao Liu, and Tianwei Zhang. Text's Armor: Optimized local adversarial perturbation against scene text editing attacks. In *Proceedings of the 30th ACM International Conference on Multimedia*, pp. 2777–2785. Association for Computing Machinery, 2022. doi: 10.1145/3503161.3548103.

Haotian Xue, Chumeng Liang, Xiaoyu Wu, and Yongxin Chen. Toward effective protection against diffusion-based mimicry through score distillation. In *International Conference on Learning Representations*, 2024.

Jiachen Yang, Yusong Wang, Yanmei Fang, Yunshu Dai, and Fangjun Huang. Variance as a catalyst: Efficient and transferable semantic erasure adversarial attack for customized diffusion models. In *Forty-second International Conference on Machine Learning*, 2025.

Weichao Zeng, Yan Shu, Zhenhang Li, Dongbao Yang, and Yu Zhou. Textctrl: Diffusion-based scene text editing with prior guidance control. *Advances in Neural Information Processing Systems*, 37: 138569–138594, 2024.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023.

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*, pp. 586–595, 2018.

Appendix:

VTEGuard: Proactive Defense against Diffusion-Based Visual Text Editing

A VTEGuard algorithm **14**

B Dataset **14**

C Evaluation details **18**

C.1 Quantitative metrics 18

D Experimental details **21**

E More Experimental Results **21**

E.1 Qualitative Results for Cross-Model Transferability 21

E.2 Qualitative Comparison of Perturbation Visibility 22

F Protection Resilience Against Noise Purification **25**

G Transferability to black-box models **25**

A VTEGUARD ALGORITHM

VTEGuard protects images against unauthorized diffusion-based visual text editing by optimizing perturbations in the full-image YCbCr-DCT domain. Guided by the ESL objective, it encourages VAE posterior-variance expansion while assigning additional weight to low-variance latent dimensions. The perturbations are iteratively updated under a coefficient-space constraint and transformed back into the image domain to produce the protected image. Algorithm 1 summarizes the complete procedure.

Algorithm 1 Full-image DCT protection with ESL and pixel-budget clipping

Require: Input image $\mathbf{x} \in [0,1]^{H\times W\times 3}$, frozen VAE encoder $\mathcal{E}_\phi$, optional mask $\mathbf{M} \in [0,1]^{H\times W}$, coefficient budget ϵ_{freq}, pixel budget ϵ_{pix}, step size η, iterations N, temperature τ, loss weights $\lambda_{\text{lin}}, \lambda_{\text{sm}}$

Ensure: Protected image $\mathbf{x}^{\text{adv}}$ with $\|\mathbf{x}^{\text{adv}} - \mathbf{x}\|_\infty \le \epsilon_{\text{pix}}$

if $\mathbf{M}$ is not provided **then**
  $\mathbf{M} \leftarrow \mathbf{1}$ ▷ Allow perturbations over the entire image
$\mathbf{l} \leftarrow \max(\mathbf{0}, \mathbf{x} - \epsilon_{\text{pix}})$
$\mathbf{u} \leftarrow \min(\mathbf{1}, \mathbf{x} + \epsilon_{\text{pix}})$ ▷ Element-wise pixel bounds
$\mathbf{A} \leftarrow \mathcal{D}\big(\mathcal{T}(\mathbf{x}) - 128\big)$ ▷ Full-image DCT
$\mathbf{\Delta} \leftarrow \mathbf{0}$
$\mathbf{S} \leftarrow \mathbf{1}$
$\mathbf{S}_{c,0,0} \leftarrow 0, \quad \forall c \in \{Y, C_b, C_r\}$ ▷ Exclude DC coefficient updates
for $t \leftarrow 1$ **to** N **do**
  $\widetilde{\mathbf{x}} \leftarrow \mathcal{T}^{-1}\left(\mathcal{D}^{-1}(\mathbf{A} + \mathbf{\Delta}) + 128\right)$ ▷ Reconstruct RGB image
  $\widetilde{\mathbf{x}} \leftarrow \text{clip}_{[\mathbf{l},\mathbf{u}]}(\widetilde{\mathbf{x}})$ ▷ Enforce pixel budget and valid range
  $\mathbf{x}' \leftarrow (1 - \mathbf{M}) \odot \mathbf{x} + \mathbf{M} \odot \widetilde{\mathbf{x}}$ ▷ Apply spatial mask
  $\mathbf{v} \leftarrow \text{vec}\left(\mathcal{E}_\phi(2\mathbf{x}' - 1).\texttt{latent_dist.var}\right)$ ▷ VAE posterior variance
  $\mathcal{L} \leftarrow \lambda_{\text{lin}} \sum_j v_j - \lambda_{\text{sm}} \tau \log\left(\sum_j \exp(-v_j/\tau)\right)$
  $\mathbf{g} \leftarrow \nabla_{\mathbf{\Delta}} \mathcal{L}$
  $\mathbf{\Delta} \leftarrow \mathbf{\Delta} + \eta\, \mathbf{S} \odot \text{sign}(\mathbf{g})$ ▷ Gradient ascent
  $\mathbf{\Delta} \leftarrow \mathbf{S} \odot \text{clip}_{[-\epsilon_{\text{freq}}, \epsilon_{\text{freq}}]}(\mathbf{\Delta})$ ▷ Coefficient projection
$\widetilde{\mathbf{x}} \leftarrow \mathcal{T}^{-1}\left(\mathcal{D}^{-1}(\mathbf{A} + \mathbf{\Delta}) + 128\right)$
$\widetilde{\mathbf{x}} \leftarrow \text{clip}_{[\mathbf{l},\mathbf{u}]}(\widetilde{\mathbf{x}})$ ▷ Enforce final pixel budget
$\mathbf{x}^{\text{adv}} \leftarrow (1 - \mathbf{M}) \odot \mathbf{x} + \mathbf{M} \odot \widetilde{\mathbf{x}}$
return $\mathbf{x}^{\text{adv}}$

B DATASET

To comprehensively evaluate the effectiveness and generalization of image protection across diverse editing scenarios, we conduct experiments on multiple datasets covering visual-text editing, image-to-image editing, inpainting, and personalized generation. The datasets span diverse image contents, editing regions, and prompt instructions, providing a broad evaluation of protection against different image editing pipelines.

VTEGuardBench To evaluate protection against visual-text editing, we construct VTEGuardBench by selecting 100 images from IC13 (Karatzas et al., 2013) and ScenePair (Zeng et al., 2024). We select images according to three criteria: (1) the image contains at least one clearly visible and legible text region; (2) the text region can be unambiguously localized and provides sufficient spatial extent for text replacement; and (3) the selected images collectively cover diverse scene contexts and text layouts. We exclude images with extremely small or severely occluded text, ambiguous text boundaries, or substantial image degradation that may prevent reliable editing. The selection is performed independently of the protection results. The resulting images cover diverse contexts, including commercial advertisements, directional signs, safety notices, and printed

materials. All input images have a spatial resolution of 512×512 pixels. Representative samples from VTEGuardBench are shown in Figure 6.

ImageNet-Edit We use ImageNet-Edit (Kim et al., 2025) for image-to-image editing with SDEdit. The evaluation set contains 80 source images, each paired with a target editing prompt. We generate protected versions of the source images and apply the corresponding prompts to both clean and protected inputs. We compare the resulting outputs to evaluate the effect of protection on image editing. Example source images and their corresponding target editing prompts are shown in Figure 7.

Inpainting Dataset We evaluate protection against inpainting under two settings: background inpainting and foreground inpainting. For background inpainting, we select 30 portrait images from HELEN based on the face-to-image area ratio, favoring images with sufficiently large background regions for reliable editing. Each image is paired with an inpainting mask specifying the background region and five prompts describing different target backgrounds. For foreground inpainting, we collect images of diverse object categories, including animals, fruits, and cars, from publicly available sources[1]. Each image is paired with a mask specifying the foreground object to be replaced. To account for the sensitivity of inpainting models to prompt choice, we use three different editing prompts for each foreground image, allowing us to evaluate the robustness of each protection method across diverse editing instructions. All images are resized to 512×512 pixels. Example images and their corresponding inpainting masks are shown in Figure 8, and the editing prompts are provided in Table 4.

WikiArt We use WikiArt images to evaluate protection against both DiffEdit editing and DreamBooth-based style personalization. We use 55 reference images organized into five style groups: Baroque, Cubism, Expression, Fauvism, and Romanticism. These images serve as reference inputs for learning the corresponding styles. Example artworks and reference groups are shown in Figure 9.

VGGFace2 We use selected VGGFace2 images for facial editing with DiffEdit and identity personalization with DreamBooth. We use 60 reference images from five identities, organized into separate identity groups. Example portraits and identity groups are shown in Figure 9.

DreamBooth dataset We additionally evaluate DreamBooth personalization using 24 reference images from five subjects in the DreamBooth dataset: `backpack_dog`, `cat`, `dog7`, `duck_toy`, and `poop_emoji`. These subjects cover animals and everyday objects, complementing the artistic styles and human identities evaluated using WikiArt and VGGFace2. The reference images are grouped by subject for personalization. Representative images from each subject are shown in Figure 9.

A [man/woman] in an airplane
A [man/woman] standing on a beach
A [man/woman] walking in a city street
A [man/woman] in a green forest
A [man/woman] working in a modern office

Table 4: Background-inpainting prompts used for portrait images.

[1] https://www.pexels.com/



Figure 6: Representative samples from VTEGuardBench, covering diverse real-world scenes and text layouts, including advertisements, directional signs, safety notices, and promotional materials.

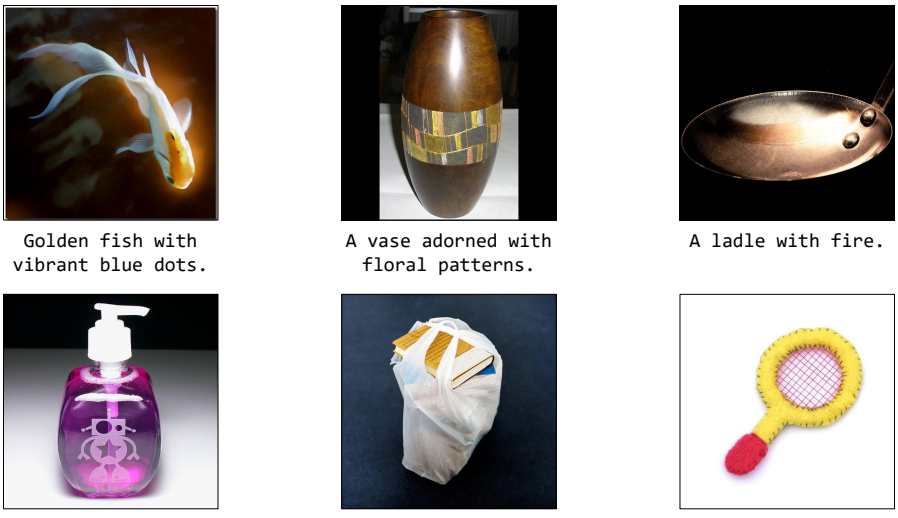


Figure 7: Example source images and target editing prompts from ImageNet-Edit. Each prompt is applied to both the clean and protected source image for image-to-image editing with SDEdit.

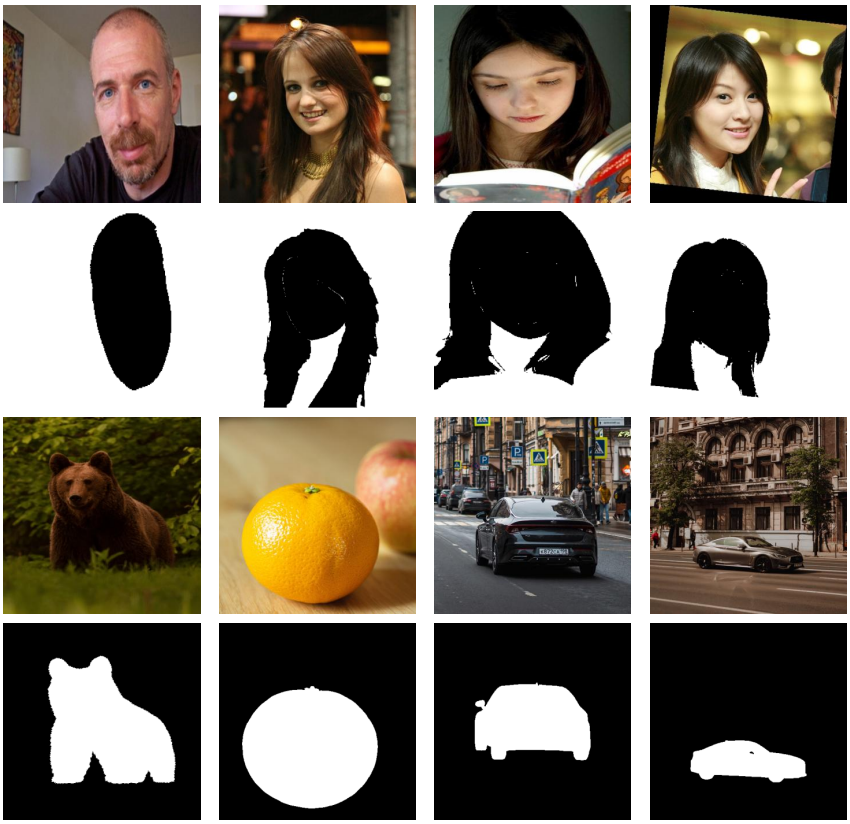


Figure 8: Example images and corresponding inpainting masks used in our evaluation. The top two rows show portrait images and background masks from HELEN for background inpainting, while the bottom two rows show object images and foreground masks for foreground inpainting.

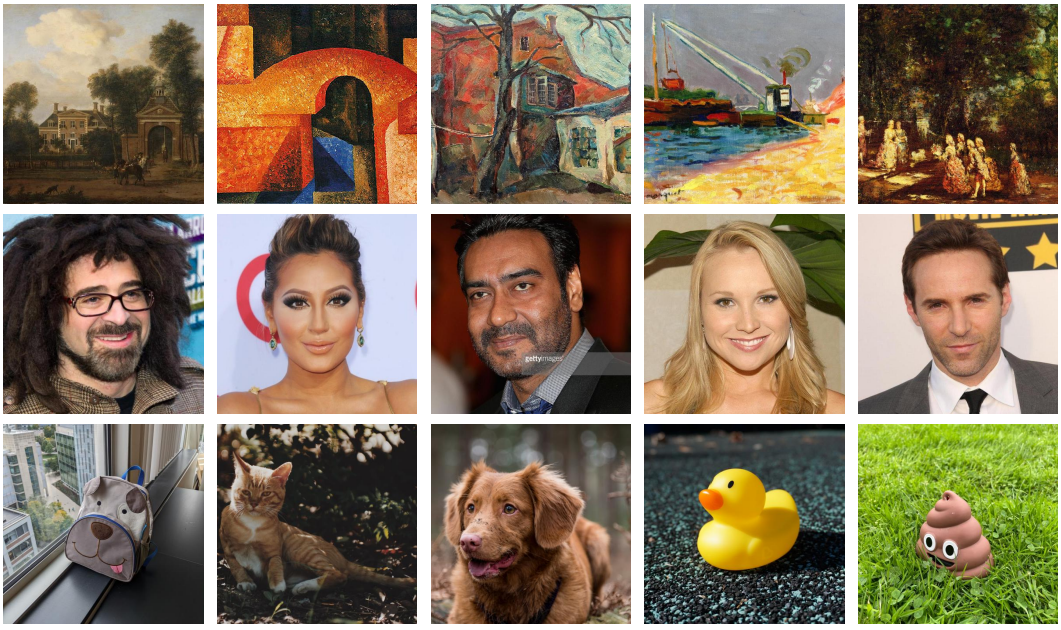


Figure 9: Example reference images used for DreamBooth personalization. Images are grouped by artistic style for WikiArt, identity for VGGFace2, and subject for the DreamBooth dataset.

C EVALUATION DETAILS

We evaluate image protection from two complementary perspectives: input imperceptibility and downstream protection effectiveness. Eight quantitative metrics characterize pixel differences, perceptual similarity, feature distributions, image quality, semantic alignment, and text recognition.

Let x_i and x_i^p denote the original and protected input images, respectively. Let y_i and y_i^p denote their corresponding outputs under the same editing instruction. The target text is denoted by t_i, and its predefined editing region by R_i.

Perturbation imperceptibility is assessed by comparing x_i with x_i^p, whereas protection effectiveness is assessed by comparing y_i with y_i^p. Larger output differences do not necessarily indicate failed text editing. We therefore additionally measure whether the requested target text is correctly recognized within the editing region.

C.1 QUANTITATIVE METRICS

C.1.1 FRÉCHET INCEPTION DISTANCE (FID)

FID measures the distributional difference between two image sets using features extracted by a pretrained Inception network. We calculate it separately for original versus protected inputs and clean versus protected edits. The metric follows the formulation introduced by (Heusel et al., 2017):

$$\mathrm{FID}(\mathcal{A}, \mathcal{B}) = \|\mu_{\mathcal{A}} - \mu_{\mathcal{B}}\|_2^2 + \mathrm{Tr}\left[\Sigma_{\mathcal{A}} + \Sigma_{\mathcal{B}} - 2\left(\Sigma_{\mathcal{A}}^{1/2}\Sigma_{\mathcal{B}}\Sigma_{\mathcal{A}}^{1/2}\right)^{1/2}\right], \tag{1}$$

where $\mu_{\mathcal{A}}$ and $\Sigma_{\mathcal{A}}$ are the empirical mean and covariance of features from image set $\mathcal{A}$. The corresponding quantities for $\mathcal{B}$ are defined analogously.

Lower input FID indicates better preservation of the original feature distribution. Higher editing FID indicates greater distributional deviation from clean edits, without directly establishing text editing failure.

Our implementation uses 2,048-dimensional features from torchvision's ImageNet-pretrained Inception-V3 (Szegedy et al., 2016) and unbiased sample covariances. Consequently, these values are not directly interchangeable with FID values obtained using the standard TensorFlow FID feature extractor.

C.1.2 STRUCTURAL SIMILARITY INDEX MEASURE (SSIM)

SSIM measures image similarity through local luminance, contrast, and structure. Following (Wang et al., 2004), the local similarity between images a and b is expressed as

$$\mathrm{SSIM}(a, b) = \frac{(2\mu_a\mu_b + C_1)(2\sigma_{ab} + C_2)}{(\mu_a^2 + \mu_b^2 + C_1)(\sigma_a^2 + \sigma_b^2 + C_2)}, \tag{2}$$

where μ_a and μ_b are local mean intensities, σ_a^2 and σ_b^2 are local variances, and σ_{ab} is the local covariance. Constants C_1 and C_2 stabilize the computation.

We calculate SSIM on RGB images and average the local scores across spatial positions and channels. Higher $\mathrm{SSIM}(x_i, x_i^p)$ indicates better input fidelity. Lower $\mathrm{SSIM}(y_i, y_i^p)$ indicates greater structural deviation in the edited output.

C.1.3 PEAK SIGNAL-TO-NOISE RATIO (PSNR)

PSNR quantifies pixel fidelity through the mean squared error between two images. For images $a, b \in \mathbb{R}^{H\times W\times C}$, the mean squared error is

$$\mathrm{MSE}(a, b) = \frac{1}{HWC}\sum_{h=1}^{H}\sum_{w=1}^{W}\sum_{c=1}^{C}(a_{hwc} - b_{hwc})^2. \tag{3}$$

PSNR is then defined as

$$\mathrm{PSNR}(a, b) = 10 \log_{10} \left(\frac{L^2}{\mathrm{MSE}(a, b)} \right), \tag{4}$$

where L is the maximum pixel value. We normalize RGB intensities to $[0, 1]$, giving $L = 1$, and report PSNR in decibels.

Higher input PSNR indicates smaller protection-induced perturbations. Lower editing PSNR indicates larger pixel differences between clean and protected edits.

C.1.4 Learned Perceptual Image Patch Similarity (LPIPS)

LPIPS measures perceptual differences using deep feature representations. It compares channel-normalized features and applies learned channel weights, following (Zhang et al., 2018):

$$\mathrm{LPIPS}(a, b) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} \left\| w_l \odot \left(\hat{f}_l(a)_{hw} - \hat{f}_l(b)_{hw} \right) \right\|_2^2, \tag{5}$$

where $\hat{f}_l(\cdot)$ denotes the normalized feature representation at layer l. The vector w_l contains learned channel weights, and H_l, W_l are the feature-map dimensions.

We use the AlexNet-based LPIPS model. Lower input LPIPS indicates better perceptual fidelity, whereas higher editing LPIPS indicates greater perceptual deviation from the clean edit.

C.1.5 Blind/Referenceless Image Spatial Quality Evaluator (BRISQUE)

BRISQUE assesses image quality without requiring a reference image. It extracts natural-scene statistics from locally normalized luminance and neighboring coefficient products. A pretrained regression model maps these statistics to a quality score.

The locally normalized luminance coefficient at position (h, w) is

$$\hat{I}(h, w) = \frac{I(h, w) - \mu(h, w)}{\sigma(h, w) + \epsilon}, \tag{6}$$

where $\mu(h, w)$ and $\sigma(h, w)$ are the local luminance mean and standard deviation. The resulting score can be written as

$$\mathrm{BRISQUE}(I) = g_{\mathrm{SVR}}(\phi_{\mathrm{NSS}}(I)), \tag{7}$$

where ϕ_{NSS} extracts natural-scene-statistics features and g_{SVR} is the pretrained support vector regressor.

We evaluate protected inputs and protected edited outputs separately. Lower scores generally indicate better predicted visual quality.

C.1.6 CLIP Image Similarity (CLIP-I)

We use CLIP-I to measure feature similarity between clean and protected edits. Let $E_I(\cdot)$ denote the pretrained CLIP image encoder. The score is

$$\text{CLIP-I}_i = \frac{E_I(y_i)^\top E_I(y_i^p)}{\|E_I(y_i)\|_2 \, \|E_I(y_i^p)\|_2}. \tag{8}$$

The representations are obtained from the CLIP model introduced by (Radford et al., 2021). We use the ViT-L/14 checkpoint with its standard 224-pixel image preprocessing.

Lower CLIP-I indicates greater deviation from the clean edit in CLIP feature space. We report the raw cosine similarity without multiplying it by 100.

C.1.7 CLIP TEXT–IMAGE SIMILARITY (CLIP-S)

CLIP-S evaluates the alignment between the protected edited image and a textual description containing the requested target string. For sample i, we construct the description

$$d_i = \texttt{A photo of scene text reading "}t_i\texttt{".}$$

Using the image and text encoders of CLIP (Radford et al., 2021), the score is

$$\text{CLIP-S}_i = \frac{E_I(y_i^p)^\top E_T(d_i)}{\|E_I(y_i^p)\|_2 \, \|E_T(d_i)\|_2}, \tag{9}$$

where $E_T(\cdot)$ denotes the text encoder.

Lower CLIP-S indicates weaker alignment with the target description. CLIP-S measures broad image–text alignment and does not establish whether the target string is rendered exactly.

C.1.8 OCR-BASED TARGET TEXT EDITING SUCCESS RATE (TESR)

TESR measures the proportion of protected edits in which OCR correctly recognizes the requested target text within the predefined editing region. It is defined as

$$\text{TESR} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left[\text{norm}\left(\text{OCR}(y_i^p|_{R_i})\right) = \text{norm}(t_i)\right]. \tag{10}$$

where $\mathbb{I}[\cdot]$ is the indicator function. The reference t_i is the requested replacement text, rather than the original text.

We use PaddleOCR with the PP-OCRv5 server detection and recognition models (Cui et al., 2026). Every method uses the same predefined region R_i for each image. The cropped region is enlarged twofold using bicubic interpolation and padded with a 32-pixel white border before recognition.

Recognized fragments are arranged in reading order and joined with a single space. Normalization applies Unicode NFKC, case folding, and whitespace trimming and collapsing. Punctuation is preserved. Success requires equality between the complete normalized strings; substring matches and partial matches are rejected. Missing detections are counted as failures.

TESR is reported as a proportion in $[0, 1]$ and denoted TESR $\downarrow$, where lower values indicate fewer successful target text edits. For example, a TESR of 0.02 corresponds to a 2% success rate. This metric evaluates OCR-based target-text correctness without requiring preservation of the requested font, layout, or background.

D EXPERIMENTAL DETAILS

For all compared methods, we follow their official implementations to optimize the corresponding perturbations. We retain the method-specific projected gradient descent (PGD) (Madry et al., 2018) optimization settings provided by the original implementations, while fixing the number of optimization iterations to 300 for all comparisons throughout the paper. For a fair comparison, we further constrain all methods to the same ℓ_∞ perturbation budget of $\epsilon = 16/255$. In addition, we fix the random seeds during editing and generation to ensure reliable and reproducible comparisons across different protection methods.

L2 objective in the ablation study. The ℓ_2 objective in Table 2 is defined over the VAE posterior variance:

$$\mathcal{L}_{\mathrm{L2}}(x') = \left\| \mathrm{vec}\left(\sigma_\phi^2(x')\right) \right\|_2, \tag{11}$$

where x' denotes the protected image and $\sigma_\phi^2(x')$ is the posterior variance predicted by the frozen VAE encoder. We maximize this objective via gradient ascent to increase the overall magnitude of the posterior variance. This is the unsquared Euclidean norm of the variance vector, rather than a reconstruction loss or a distance between clean and protected representations. For a nonzero variance vector $\mathbf{v}$, $\partial \mathcal{L}_{\mathrm{L2}}/\partial v_j = v_j/\|\mathbf{v}\|_2$. Thus, larger variance components receive greater weight in the variance-space gradient, whereas ESL provides a uniform base gradient with additional emphasis on low-variance components.

We also evaluate our protective perturbations across a range of image editing and personalization tasks, including inpainting, SDEdit, DiffEdit, and DreamBooth.

Inpainting For the inpainting task, we use the Stable Diffusion 1.5 inpainting pipeline. We follow the default inference configuration, using 50 denoising steps, a guidance scale of 7.5, and an editing strength of 1.0. The same inpainting masks and text prompts are applied to both clean and protected inputs for paired evaluation.

SDEdit For image-to-image editing with SDEdit, we use 50 inference steps with a guidance scale of 7.5 and an editing strength of 0.5. For each source image, the same target prompt and generation settings are used for the clean and protected versions to ensure a controlled comparison.

DiffEdit For DiffEdit, we use 50 inference steps, a guidance scale of 7.5, and an editing strength of 0.8. We keep the editing prompt and all generation parameters identical between clean and protected inputs, such that differences in the resulting images can be attributed to the applied protection.

DreamBooth For DreamBooth personalization, our main experiments adopt full fine-tuning of both the UNet and text encoder, while keeping the VAE frozen. Training is performed at a resolution of 512×512 pixels for 1,000 optimization steps. We use `sks` as the instance identifier. For paired comparison, models trained on clean and protected reference images are evaluated using exactly the same prompts and random seeds.

E MORE EXPERIMENTAL RESULTS

E.1 QUALITATIVE RESULTS FOR CROSS-MODEL TRANSFERABILITY

To complement the quantitative cross-model evaluation in the main paper, we provide additional qualitative results for transfer from TextDiffuser2 to AnyText2. The protective perturbations are optimized using TextDiffuser2 and directly applied to AnyText2 without re-optimization. As shown in Fig. 10, VTEGuard remains effective under this unseen text-editing model, substantially disrupting the rendering of the requested target text while preserving the visual quality of the protected input. We further report quantitative results for the same TextDiffuser2-to-AnyText2 transfer setting in Tab. 5. VTEGuard achieves the highest FID and LPIPS, as well as the lowest CLIP-S, CLIP-I, and TESR, demonstrating its strong cross-model transferability and effectiveness in preventing target-text rendering on AnyText2.

Table 5: Quantitative cross-model transferability from TextDiffuser2 to AnyText2. Protective perturbations are optimized on TextDiffuser2 and directly transferred to AnyText2.

Method	**FID↑**	**LPIPS↑**	**CLIP-S↓**	**CLIP-I↓**	**TESR↓**
AdvDM	135.750	0.4812	0.2296	0.8047	0.53
PhotoGuard	212.598	0.5315	0.2298	0.7303	0.56
SDS	168.642	0.5552	0.2438	0.7822	0.52
AdvPaint	186.467	0.4927	0.2288	0.7651	0.55
BlurGuard	86.338	0.2121	0.2324	0.8994	0.58
PID	220.492	0.5624	0.2346	0.6929	0.49
LA	<u>241.575</u>	<u>0.7212</u>	<u>0.1802</u>	<u>0.6274</u>	<u>0.15</u>
VTEGuard(Ours)	**290.521**	**0.7873**	**0.1786**	**0.5430**	**0.05**

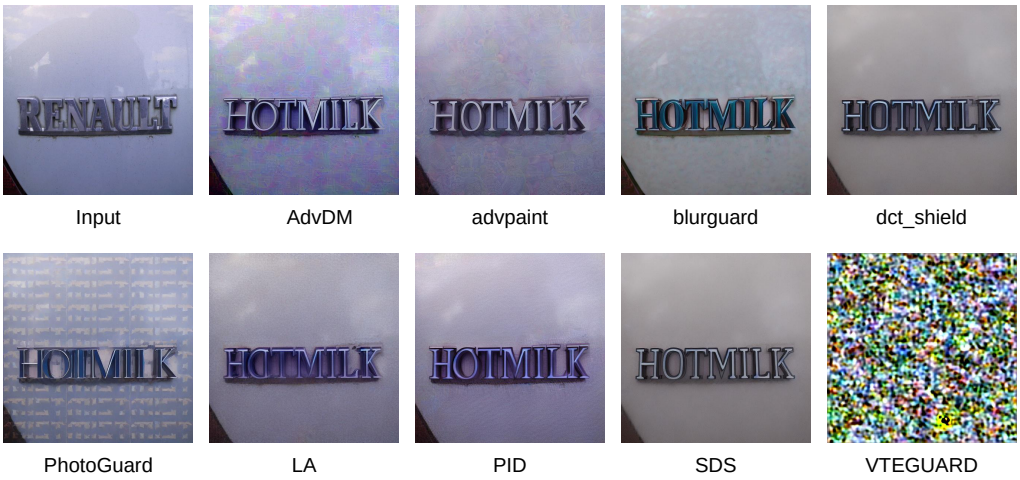
E.2 Qualitative Comparison of Perturbation Visibility

Figure 11 provides qualitative comparisons of perturbation visibility on two representative images with different visual content. For each example, we show the full protected image together with magnified local regions to expose subtle perturbation artifacts. Several baseline methods introduce visible color shifts, structured patterns, or fine-grained texture distortions under magnification. In contrast, VTEGuard remains visually close to the unprotected input across both examples and different local regions.

These qualitative results complement the quantitative imperceptibility metrics reported in the main paper, indicating that the strong protection effectiveness of VTEGuard does not rely on visually conspicuous input degradation.

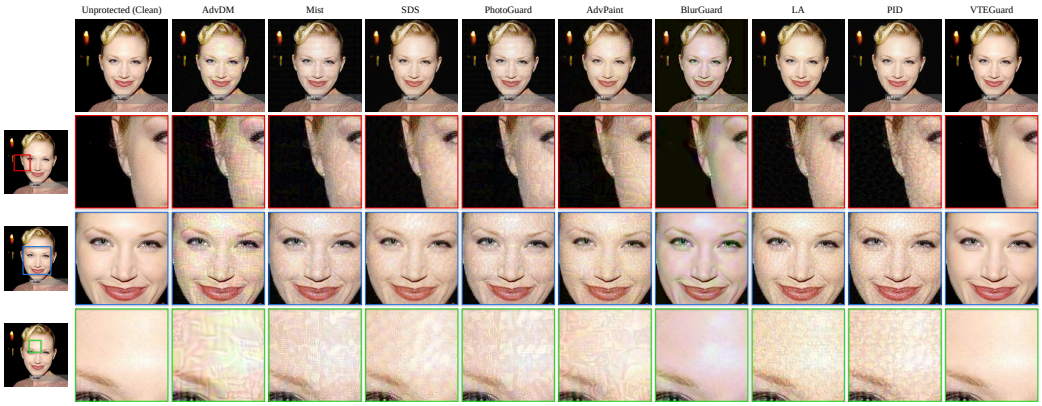


(a) Editing “GOOD” to “WHEN”.

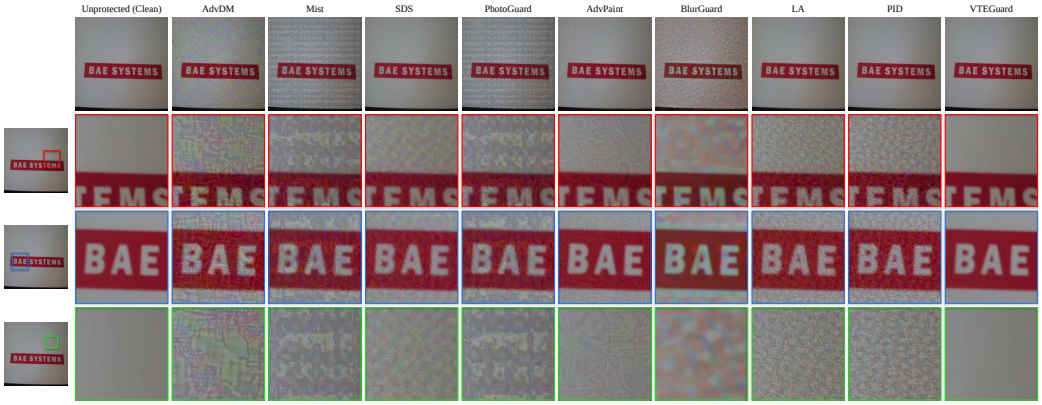


(b) Editing “RENAULT” to “HOTMILK”.

Figure 10: **Qualitative results of cross-model transfer from TextDiffuser2 to AnyText2.** Perturbations are optimized on TextDiffuser2 and directly transferred to AnyText2 without re-optimization. Two representative examples are shown: (a) replacing “GOOD” with “WHEN”, and (b) replacing “RENAULT” with “HOTMILK”. The same editing settings are used for the clean and protected inputs across all protection methods.



(a) Example on a portrait image.



(b) Example on a scene-text image.

Figure 11: **Qualitative comparison of perturbation visibility across protection methods.** We show two representative examples with different image content. For each example, the first row presents the full protected images, while the following rows show magnified local regions indicated by the colored boxes. Several baseline methods exhibit visible color shifts, structured patterns, or fine-grained texture artifacts under magnification, whereas VTEGuard preserves local visual appearance more closely to the unprotected input.

Table 6: Protection effectiveness under JPEG compression.

Method	FID↑	LPIPS↑	CLIP-S↓	CLIP-I↓
AdvDM	150.3858	0.4977	0.2392	0.8168
PhotoGuard	157.1343	0.4662	0.2404	0.8068
SDS	109.2909	0.4452	0.2436	0.8677
AdvPaint	117.0850	0.3618	0.2390	0.8709
BlurGuard	93.7010	0.2445	0.2380	0.9020
PID	112.8604	0.4525	0.2388	0.8820
LA	102.5496	0.4271	0.2383	0.8837
VTEGuard(Ours)	**310.2439**	**0.7781**	**0.1851**	**0.5931**

Table 7: Protection effectiveness under Gaussian Smoothing.

Method	FID↑	LPIPS↑	CLIP-S↓	CLIP-I↓
AdvDM	113.6091	0.6838	0.2427	0.8555
PhotoGuard	139.1207	0.7089	0.2440	0.8657
SDS	115.2672	0.7166	0.2410	0.8718
AdvPaint	100.4387	0.6649	0.2415	0.8872
BlurGuard	102.3243	0.6740	0.2388	0.8630
PID	115.6067	0.7055	0.2410	0.8845
LA	111.9141	0.7012	0.2405	0.8854
VTEGuard(Ours)	**184.6189**	**0.7203**	**0.2382**	**0.8027**

F Protection Resilience Against Noise Purification

We further evaluate the robustness of VTEGuard and the baseline methods against three input transformations: JPEG compression, Gaussian smoothing, and crop-and-resize. These operations may attenuate protective perturbations before the images are passed to the editing model.

JPEG compression. We apply JPEG compression with a quality factor of $Q = 75$ to the protected images before editing. This lossy compression may attenuate the optimized perturbations. Quantitative results and qualitative comparisons are presented in Table 6 and Figure 12, respectively.

Gaussian smoothing. We apply a 7×7 Gaussian kernel with a standard deviation of 1.0 pixel to the protected images before editing. The same transformation is applied to clean inputs to obtain matched references. Quantitative results and qualitative comparisons are provided in Table 7 and Figure 13, respectively.

Crop-and-resize. We remove a 64-pixel border from each side of the protected image by center cropping, then resize the remaining region to 512×512 pixels. This transformation evaluates robustness to changes in spatial alignment and resampling. Quantitative results and qualitative comparisons are reported in Table 8 and Figure 14, respectively.

For all settings, the transformations are applied after perturbation optimization and before image editing. All compared methods use the same purification configuration and downstream editing settings.

Across the three settings, VTEGuard achieves the best values on all four reported metrics, indicating persistent disruption of edited outputs after purification.

G Transferability to Black-Box Models

We conduct additional experiments to evaluate the black-box transferability of VTEGuard and the baseline protection methods across different diffusion model versions. Specifically, for SDEdit and DiffEdit, perturbations are optimized against Stable Diffusion 1.5 and directly evaluated on Stable Diffusion 2.1. For the inpainting task, perturbations are generated using Stable Diffusion 1.5

Table 8: Protection effectiveness under crop-and-resize purification.

Method	FID↑	LPIPS↑	CLIP-S↓	CLIP-I↓
AdvDM	<u>208.2458</u>	<u>0.6933</u>	0.1845	<u>0.7043</u>
PhotoGuard	181.7146	0.6526	0.1912	0.7175
SDS	155.2259	0.6726	0.1834	0.7418
AdvPaint	176.9737	0.6280	<u>0.1792</u>	0.7328
BlurGuard	137.0914	0.6306	0.1836	0.7607
PID	166.5389	0.6810	0.1840	0.7417
LA	155.2407	0.6666	0.1848	0.7468
VTEGuard(Ours)	**208.7199**	**0.7466**	**0.1760**	**0.6913**

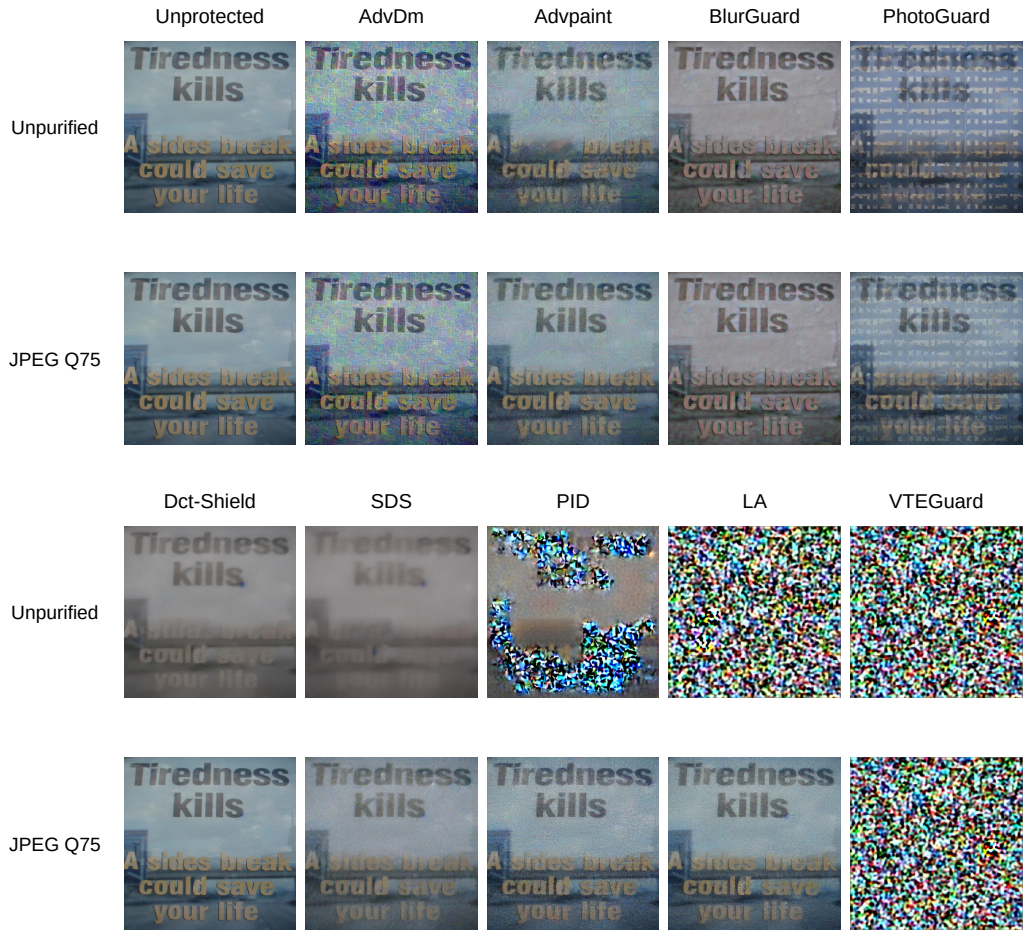


Figure 12: **Qualitative comparison after JPEG purification** ($Q = 75$)**.** Image editing results after applying JPEG compression to images protected by different methods. Results are divided into two panels for visualization.

Inpainting and transferred to Stable Diffusion 2 Inpainting. Importantly, the transferred perturbations are applied to the target models without any re-optimization or access to the target-model parameters during perturbation generation.

These source and target models correspond to different generations of Stable Diffusion with substantially different model parameters, providing a practical black-box transfer setting. This evaluation measures whether perturbations optimized for one diffusion model remain effective when the down-

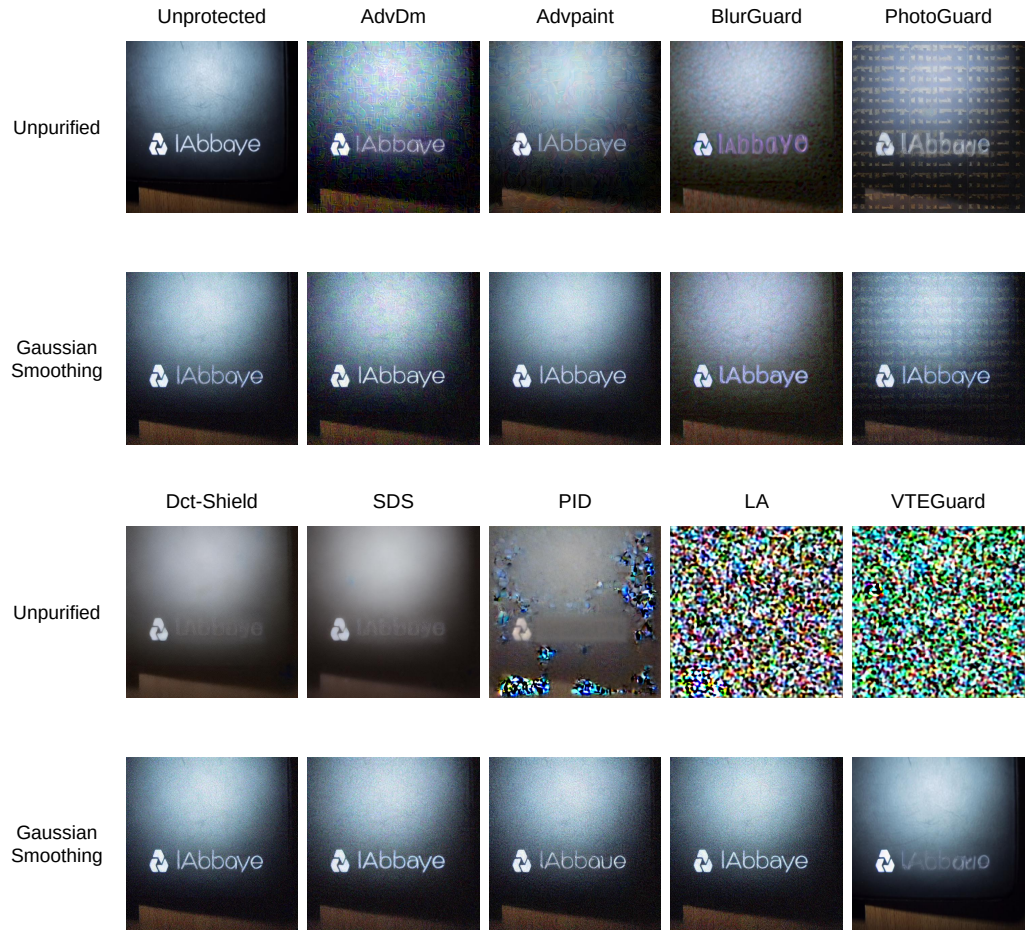


Figure 13: **Qualitative comparison after Gaussian Smoothing purification.** Image editing results after applying Gaussian Smoothing to images protected by different methods. Results are divided into two panels for visualization.

Table 9: Protection effectiveness under black-box transfer on SDEdit from Stable Diffusion 1.5 to Stable Diffusion 2.1.

Method	SSIM↓	LPIPS↑	FID↑	CLIP-S↓	CLIP-I↓
AdvPaint	**0.2938**	0.6731	245.5681	0.1934	0.7643
BlurGuard	0.5794	0.3965	142.5023	0.2243	0.8692
AdvDM	0.3220	0.7454	280.3382	0.2023	0.7521
Mist	0.3424	0.6463	**280.7545**	0.1897	0.7521
PhotoGuard	0.3473	0.6498	269.2655	0.1966	0.7518
SDS	0.3691	0.6683	222.3999	0.2212	0.7610
LA	0.3669	0.7309	265.8809	0.1746	0.6812
PID	0.3705	0.6592	219.5125	0.2099	0.7449
VTEGuard(Ours)	0.3431	**0.7648**	274.8973	**0.1687**	**0.6440**

stream editor is replaced by an unseen model. The full quantitative results are reported in Tables 9, 10, and 11.

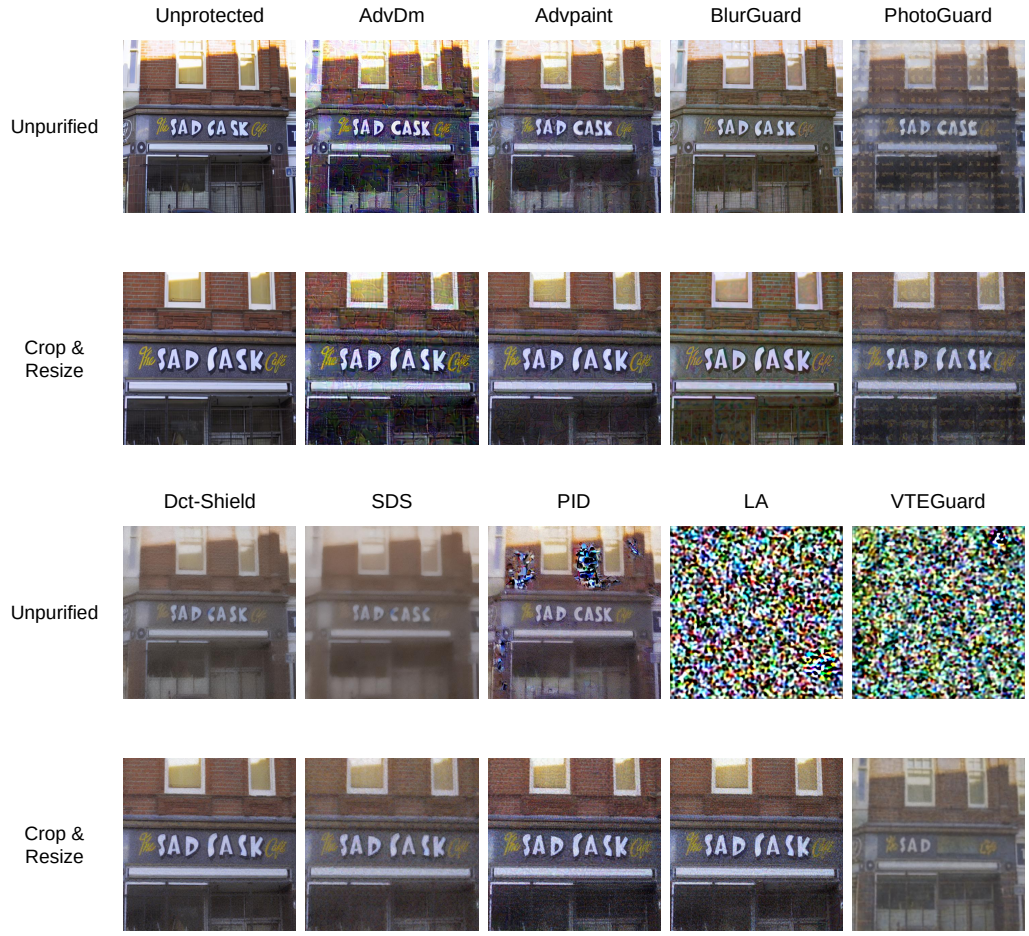


Figure 14: **Qualitative comparison after Crop-and-Resize purification.** Image editing results after applying crop-and-resize purification to images protected by different methods. Results are divided into two panels for visualization.

Table 10: Protection effectiveness under black-box transfer on DiffEdit from Stable Diffusion 1.5 to Stable Diffusion 2.1.

Method	SSIM↓	LPIPS↑	FID↑	CLIP-S↓	CLIP-I↓
AdvPaint	**0.3757**	0.6695	216.8508	0.1853	0.6821
BlurGuard	0.6567	0.3411	122.2064	0.1864	0.8226
AdvDM	<u>0.4215</u>	0.6230	200.5717	0.2029	0.7505
Mist	0.4975	0.5759	215.7465	<u>0.1840</u>	0.6858
PhotoGuard	0.5053	0.5845	222.6033	0.1893	0.7000
SDS	0.4920	0.6803	180.6079	0.2043	0.6954
LA	0.4896	<u>0.7730</u>	<u>225.8597</u>	0.1845	0.6829
PID	0.4769	0.6659	221.5786	0.1995	<u>0.6794</u>
VTEGuard	0.4708	**0.7839**	**230.7595**	**0.1834**	**0.6746**

Table 11: Protection effectiveness under black-box transfer from Stable Diffusion 1.5 Inpainting to Stable Diffusion 2 Inpainting.

Method	**SSIM↓**	**LPIPS↑**	**FID↑**	**CLIP-S↓**	**CLIP-I↓**
AdvPaint	<u>0.3536</u>	<u>0.7468</u>	<u>263.0184</u>	0.2108	<u>0.6901</u>
BlurGuard	0.6749	0.3040	87.8902	0.2394	0.8792
DCT-Shield	0.5184	0.5372	124.4216	0.2318	0.8229
DiffusionGuard	0.5861	0.4331	107.6574	0.2309	0.8464
AdvDM	0.4489	0.5967	168.5497	0.2271	0.7862
Mist	0.4465	0.6077	213.1472	0.2199	0.7450
PhotoGuard	0.4470	0.6147	212.6024	0.2187	0.7414
SDS	0.4730	0.6313	138.7763	0.2324	0.7701
LA	0.4297	0.6158	196.2038	<u>0.1943</u>	0.7049
PID	0.4739	0.5589	126.8356	0.2328	0.8123
VTEGuard(Ours)	**0.1427**	**0.9070**	**419.6873**	**0.1434**	**0.5089**