Appendix

A. Detailed Experiment Setups

A.1. Implementation Details of FACELOCK

FACELOCK optimizes perturbation on facial disruption and feature embedding disparity that prevent biometric recognition post-editing. The pseudocode of FACELOCK is presented in Algorithm 1. More specifically, the facial recognition loss function f_{FR} is defined as the negative of the similarity score between the input images computed by the CVLFACE model[1], and the feature disparity loss function f_{FE} is computed as the weighted sum of the layer-wise feature embedding distances across the feature extractor network. As mentioned in Sec 5, we also include the untargeted latent-wise loss from EditShield[2] as a regularization term to stabilize the protection results. The hyper-parameters used in our implementation are summarized in **Tab. A1**.

Algorithm 1 FACELOCK

Input: Input image $\mathbf{x}$, VAE $\mathcal{E}, \mathcal{D}$ in the diffusion model, step size α, number of steps N, overall perturbation budget ϵ, regularization weight λ, facial recognition loss function f_{FR}, feature disparity loss function f_{FE}

Initialize perturbation $\boldsymbol{\delta} \leftarrow N(0, \mathbf{I})$, and the protected image $\mathbf{x}' \leftarrow \mathbf{x} + \boldsymbol{\delta}$
Compute the latent embedding of the input image $\mathbf{z} \leftarrow \mathcal{E}(\mathbf{x})$
for $n = 1$ to N **do**
    Compute the latent embedding of the protected image $\mathbf{z}' \leftarrow \mathcal{E}(\mathbf{x}')$
    Compute the decoded image from the latent embedding $\mathbf{x}_d \leftarrow \mathcal{D}(\mathbf{z}')$
    Compute the facial recognition loss $l_{\text{FR}} \leftarrow f_{\text{FR}}(\mathbf{x}_d, \mathbf{x})$
    Compute the feature disparity loss $l_{\text{FE}} \leftarrow f_{\text{FE}}(\mathbf{x}_d, \mathbf{x})$
    Compute the latent loss (regularization term) $l_{\text{L}} \leftarrow \|\mathbf{z}' - \mathbf{z}\|_2^2$
    Update the perturbation $\boldsymbol{\delta} \leftarrow \boldsymbol{\delta} + \alpha \cdot \text{sign}(\nabla_{\mathbf{x}'}(l_{\text{FR}} + l_{\text{FE}} + \lambda \cdot l_{\text{L}}))$
    $\boldsymbol{\delta} \leftarrow \text{clip}(\boldsymbol{\delta}, -\epsilon, \epsilon)$
    Update the protected image: $\mathbf{x}' \leftarrow \mathbf{x} + \boldsymbol{\delta}$
end for

Return: The protected image $\mathbf{x}'$

Table A1. Hyper-parameters used for the implementation.

Norm	**perturbation budget** ϵ	**step size** α	**number of steps** N	λ
l_∞	0.02	0.003	100	0.2

A.2. Implementation Details of Baselines

In addition to using previous methods [1, 2] as baselines, we also compare our FACELOCK approach against several widely used techniques in the adversarial machine learning field. These methods are summarized in Algorithms 2, 3, and 4. To ensure a fair comparison, we use the same hyper-parameters settings in Tab. A1.

A.3. Image Editing Details

Models. For image editing, we use the open-source instruction-guided diffusion model InstructPix2Pix [9] hosted on Hugging Face[2] as our primary target model. We use the hyper-parameters presented in **Tab. A2**. We use the same seed setting when comparing edits on the unprotected images and the images protected by different methods to ensure that the edit images are are modified in the same way and that the different editing effects are due to the protection methods instead of random seeds.
Dataset. For the human portrait images used in our experiments, we utilize a filtered subset of the CelebA-HQ dataset[3], a high-quality human face attribute dataset widely used in the facial analysis community. The dataset consists of $2,000$ human portrait images ensuring diversity across various demographic groups, including race, age, and gender, to enhance

[1]The model is available on https://github.com/mk-minchul/CVLface
[2]The model is available on https://huggingface.co/timbrooks/instruct-pix2pix
[3]The dataset is available on https://www.kaggle.com/datasets/lamsimon/celebahq/data

Algorithm 2 Untargeted Encoder Attack

Input: Input image $\mathbf{x}$, VAE $\mathcal{E}$ in the diffusion model, step size α, number of steps N, overall perturbation budget ϵ

Initialize perturbation $\boldsymbol{\delta} \leftarrow N(0, \mathbf{I})$, and the protected image $\mathbf{x}' \leftarrow \mathbf{x} + \boldsymbol{\delta}$
Compute the latent embedding of the input image $\mathbf{z} \leftarrow \mathcal{E}(\mathbf{x})$
for $n = 1$ to N **do**
    Compute the latent embedding of the protected image $\mathbf{z}' \leftarrow \mathcal{E}(\mathbf{x}')$
    Compute the latent loss $l \leftarrow \|\mathbf{z}' - \mathbf{z}\|_2^2$
    Update the perturbation $\boldsymbol{\delta} \leftarrow \boldsymbol{\delta} + \alpha \cdot \text{sign}(\nabla_{\mathbf{x}'} l)$
    $\boldsymbol{\delta} \leftarrow \text{clip}(\boldsymbol{\delta}, -\epsilon, \epsilon)$
    Update the protected image $\mathbf{x}' \leftarrow \mathbf{x} + \boldsymbol{\delta}$
end for

Return: The protected image $\mathbf{x}'$

Algorithm 3 VAE Attack

Input: Input image $\mathbf{x}$, target image $\mathbf{x}_{\text{tgt}}$,VAE $\mathcal{E}$, $\mathcal{D}$ in the diffusion model, step size α, number of steps N, overall perturbation budget ϵ

Initialize perturbation $\boldsymbol{\delta} \leftarrow N(0, \mathbf{I})$, and the protected image $\mathbf{x}' \leftarrow \mathbf{x} + \boldsymbol{\delta}$
for $n = 1$ to N **do**
    Compute the decoded image $\mathbf{x}_d \leftarrow \mathcal{D}(\mathcal{E}(\mathbf{x}'))$
    Compute the loss $l \leftarrow \|\mathbf{x}_d - \mathbf{x}_{\text{tgt}}\|_2^2$
    Update the perturbation $\boldsymbol{\delta} \leftarrow \boldsymbol{\delta} - \alpha \cdot \text{sign}(\nabla_{\mathbf{x}'} l)$
    $\boldsymbol{\delta} \leftarrow \text{clip}(\boldsymbol{\delta}, -\epsilon, \epsilon)$
    Update the protected image $\mathbf{x}' \leftarrow \mathbf{x} + \boldsymbol{\delta}$
end for

Return: The protected image $\mathbf{x}'$

Algorithm 4 CW L_2 Attack

Input: Input image $\mathbf{x}$, VAE $\mathcal{E}$ in the diffusion model, step size α, number of steps N, overall perturbation budget ϵ, weight c

Initialize $\mathbf{w} \leftarrow \mathbf{0}$
Compute the latent embedding of the input image: $\mathbf{z} \leftarrow \mathcal{E}(\mathbf{x})$
for $n = 1$ to N **do**
    Compute the protected image $\mathbf{x}' \leftarrow \frac{1}{2}(\tanh(\mathbf{w}) + 1)$
    Compute the latent embedding of the protected image $\mathbf{z}' \leftarrow \mathcal{E}(\mathbf{x}')$
    Compute the L_2 loss $l_{L_2} \leftarrow \|\mathbf{x}' - \mathbf{x}\|_2^2$
    Compute the latent loss $l_{\text{L}} \leftarrow -\|\mathbf{z}' - \mathbf{z}\|_2^2$
    Update $\mathbf{w} \leftarrow \mathbf{w} - \alpha \cdot \nabla_{\mathbf{w}}(l_{L_2} + c \cdot l_L)$
end for
Compute $\boldsymbol{\delta} \leftarrow \text{clip}(\frac{1}{2}(\tanh(\mathbf{w}) + 1) - \mathbf{x}, -\epsilon, \epsilon)$
Compute the protected image $\mathbf{x}' \leftarrow \mathbf{x} + \boldsymbol{\delta}$

Return: The protected image $\mathbf{x}'$

the representativeness of our experiments. For the editing prompts, we manually selected 25 prompts across three categories: facial feature modification, accessory adjustments, and background alternations. These prompts were specifically selected to produce noticeable changes across a wide range of images, avoiding those that would fail to affect a certain subset (*e.g.*, "*Let the person wear glasses*" will be ineffective for individuals who already wear glasses, which is a significant portion of the dataset). The specific prompts utilized in our experiments are listed in **Tab.** A3 for detailed reference.

Table A2. Hyper-parameters used for the image editing process.

image size	inference steps	image guidance scale	text guidance scale
512×512	50	1.5	7.5

Table A3. Editing prompts categorized into facial feature modifications, accessory adjustments, and background alterations.

Category	Prompts
Facial Feature Modifications	❶ Turn the person's hair pink; ❷ Let the person turn bald; ❸ Let the person have a tattoo; ❹ Let the person wear purple makeup; ❺ Let the person grow a mustache; ❻ Turn the person into a zombie; ❼ Change the skin color to Avatar blue; ❽ Add elf-like ears; ❾ Add large vampire fangs; ❿ Apply Goth style makeup.
Accessory Adjustments	❶ Let the person wear a police suit; ❷ Let the person wear a bowtie; ❸ Let the person wear a helmet; ❹ Let the person wear sunglasses; ❺ Let the person wear earrings; ❻ Let the person smoke a cigar; ❼ Place a headband in the hair; ❽ Place a tiara on the top of the head.
Background Alterations	❶ Let it be snowy; ❷ Change the background to a beach; ❸ Add a city skyline background; ❹ Add a forest background; ❺ Change the background to a desert; ❻ Set the background in a library; ❼ Let the person stand under the moon;

A.4. Evaluation Metrics

PSNR, SSIM, and LPIPS scores. In our experiments, we compute the PSNR and SSIM scores using the torchmetrics library[4], while the LPIPS score is computed using the lpips library[5]. All these three metrics are computed by comparing the similarity between the edited image without defense and the edited image with defense. A lower similarity score (lower PSNR, SSIM score and higher LPIPS score) indicates better protection. PSNR and SSIM primarily focus on pixel-level statistical information, while LPIPS evaluates the similarity of high-level semantic features, capturing perceptual differences that are more aligned with human visual perception.

CLIP-S score. In the main paper, we utilize the CLIP-S metric to assess the prompt fidelity by computing the similarity between the image embedding shift and the text embedding in the CLIP embedding space:

$$\text{CLIP-S} = \frac{(E_{\text{edit}} - E_{\text{src}}) \cdot E_{\text{prompt}}}{\|E_{\text{edit}} - E_{\text{src}}\| \|E_{\text{prompt}}\|}, \tag{A1}$$

where E_{src} denotes the CLIP image embedding of the source image, E_{edit} denotes the CLIP image embedding of the edited image, and E_{prompt} denotes the CLIP text embedding of the prompt instruction. This formulation is particularly suitable for our experiments because the prompts are designed as instructions describing the expected transformation or modification from the source image to the edited image.

CLIP-SD score. Following PhotoGuard's evaluation metric [1], an alternative approach to assess the prompt fidelity is to compute the cosine similarity directly between the embedding of the edited image and the embedding of the descriptive text prompt in the CLIP embedding space:

$$\text{CLIP-SD} = \frac{E_{\text{edit}} \cdot E_{\text{desc}}}{\|E_{\text{edit}}\| \|E_{\text{desc}}\|}, \tag{A2}$$

where E_{desc} denotes the CLIP text embedding of the descriptive text prompt. We report the CLIP-SD score for each method in **Tab. A4**. From the table, we observe that, except for the VAE method, all defense methods show a worse defense effect compared to the "No Defense" scenario. This aligns with the analysis presented in Sec 4, where we discussed how CLIP-based similarity metrics often overemphasize the elements from the prompt, leading to a prioritization of over-editing. To generate the descriptive text prompts, we leverage ChatGPT based on the prompt instructions provided in Tab. A3.

Table A4. Quantitative evaluation on prompt fidelity using CLIP-SD. The ↓ indicates that a lower CLIP-SD score is preferred for a successful defense.

Method	No Defense	PhotoGuard	EditShield	Untargeted Encoder	CW L2	VAE	FACELOCK(ours)
CLIP-SD↓	0.272±0.029	0.283±0.029	0.277±0.027	0.284±0.024	0.277±0.027	0.270±0.029	0.283±0.024

[4]This library can be installed from https://lightning.ai/docs/torchmetrics/stable/
[5]This library can be installed from https://pypi.org/project/lpips/

CLIP-I score. In the main paper, we utilize the CLIP-I metric to assess the image integrity by computing the similarity between the edited image embedding and the source image embedding in the CLIP embedding space:

$$\text{CLIP-I} = \frac{E_{\text{edit}} \cdot E_{\text{src}}}{\|E_{\text{edit}}\| \|E_{\text{src}}\|}. \tag{A3}$$

The CLIP-I metric is used as a general indicator of the preservation effect, providing an overall measure of how similar the edited image is to the source image in the CLIP embedding space. While this serves as a useful first step in generally evaluating image integrity, it does not specifically address biometric integrity, which is central to protecting human portrait images.

FR score. In the main paper, we utilize the CVLFACE model to compute the facial recognition similarity score between the edited and source image to indicate the preservation effect of biometric integrity:

$$\text{FR} = \text{CVLFACE}(I_{\text{edit}}, I_{\text{src}}), \tag{A4}$$

where I_{src} denotes the source image, and I_{edit} denotes the edited image. Unlike other general image similarity metrics, the CVLFACE model is tailored to assess the consistency of facial features, making it more suitable for evaluating how well the identity of the person is preserved after the image has been edited. The FR score plays a key role in assessing whether the protection method effectively disrupts the biometric identity of the person in the image.

B. Additional Experiment Results

B.1. Qualitative Results on Background Alternation

Figure A1. Qualitative results of background alternation edits across various defense methods. Images in green frames denote successful defense.

B.2. Qualitative Results on Accessory Adjustment

Figure A2. Qualitative results of accessory adjustment edits across various defense methods. Images in green frames denote successful defense.

B.3. Qualitative Results on Facial Feature Modification

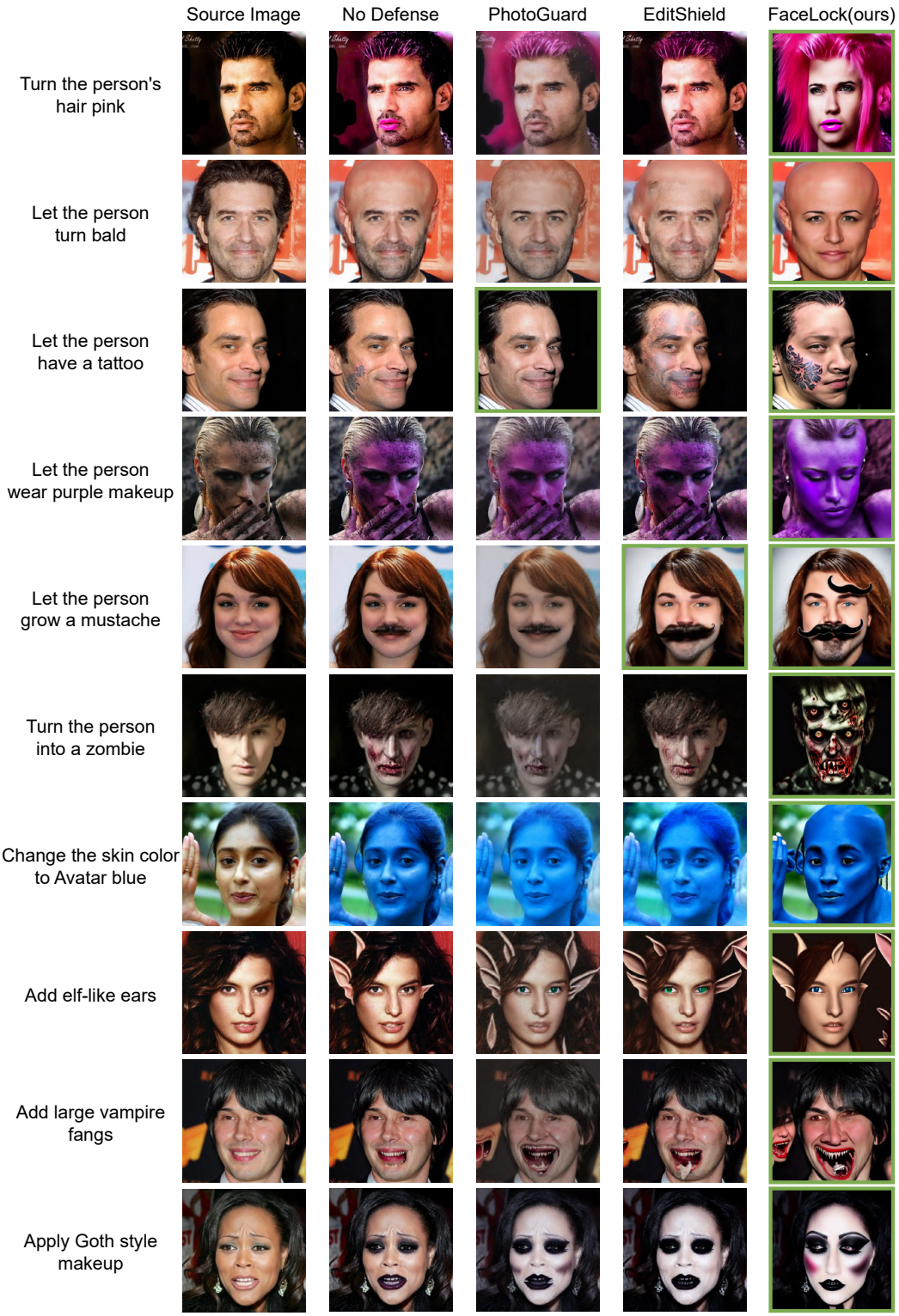

Figure A3. Qualitative results of facial feature modification edits across various defense methods. Images in green frames denote successful defense.

B.4. Qualitative Results Against Purification

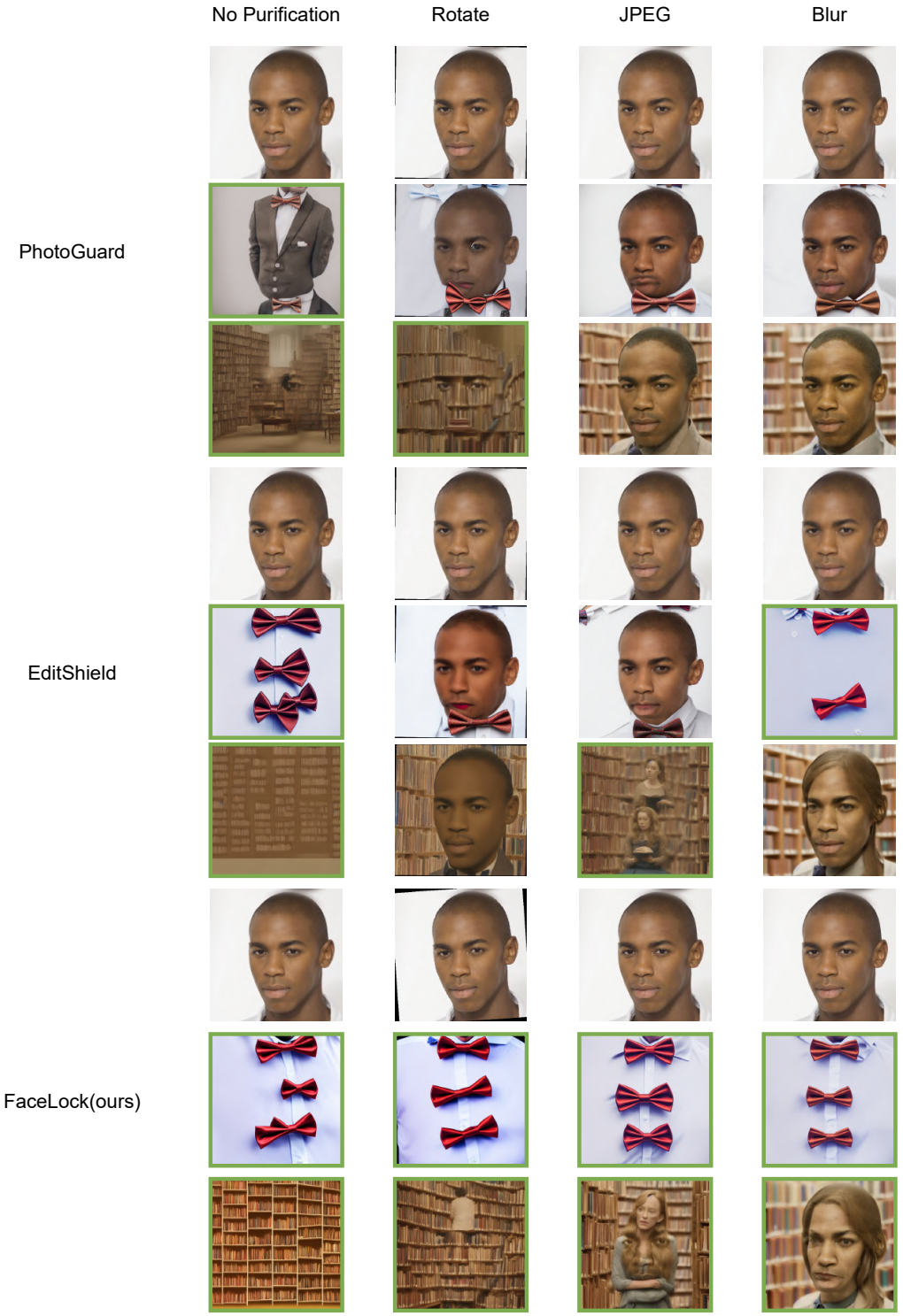

Figure A4. Qualitative results of edits on protected images after applying purification methods. Each block shows: purified protected images (1st row), edits with the instruction "Let the person wear a bowtie", and edits with the instruction "Set the background in a library". Purification methods include random rotation (-10, 10), JPEG compression (quality 75), and Gaussian blurring ($k = 5, \sigma = 1.5$). Images in green frames denote successful defense.

B.5. Results on Other Datasets

To further evaluate the effectiveness of FACELOCK, we compare its performance against existing baselines on a subset of the Flickr-Faces-HQ (FFHQ) dataset[60]. As shown in **Tab. A5**, FACELOCKachieves the lowest FR score of 0.356, demonstrating its strong identity protection while maintaining competitive performance across other key metrics.

Table A5. Quantitative evaluation on the FFHQ dataset.

Method	CLIP-S ↓	PSNR ↓	SSIM ↓	LPIPS ↑	CLIP-I ↓	FR ↓
No Defense	0.108	-	-	-	0.860	0.820
PhotoGuard	**0.095**	**14.76**	0.555	0.515	0.681	0.449
EditShield	0.098	18.92	**0.532**	0.480	0.753	0.633
FACELOCK	0.099	17.02	0.538	**0.542**	**0.680**	**0.356**

B.6. Results on Other Purification Methods

Table A6. Robustness comparison against other purification methods.

	LPIPS ↑		FR ↓	
Method	Color Jitter	DiffPure	Color Jitter	DiffPure
PhotoGuard	0.275	0.311	0.686	0.691
EditShield	0.303	**0.316**	0.593	0.610
FACELOCK	**0.319**	0.314	**0.371**	**0.504**

(a) FACELOCK (b) PhotoGuard (c) EditShield

Figure A5. Qualitative results of edits on protected images after applying other purification methods. Images in green frames denote successful defense.

To further assess the effectiveness of FACELOCK, we evaluate its robustness against other purification techniques, namely Color Jitter and DiffPure [48]. As presented in **Tab. A6**, FACELOCKconsistently achieves the lowest FR scores (0.371 and 0.504) across both purification methods, demonstrating its ability to disrupt identity features after purification. While prior

methods primarily interfere with edits, they fail to prevent identity retention post-purification. In contrast, FACELOCK ensures stronger identity removal while maintaining competitive LPIPS values, reinforcing its effectiveness as a defense mechanism. Qualitative results in **Fig.** A5 further supports these findings, showing that FACELOCK more effectively prevents identity recovery after purification.

B.7. Impact of the FR Model

To analyze the impact of the FR model, we conduct an ablation study comparing protection strength and efficiency with and without it. As shown in **Tab.** A7, incorporating the FR model reduces the FR score from 0.534 to 0.316, achieving over 40% improvement in identity protection. However, this comes with a slight increase in processing time per image (16s $\rightarrow$ 20s). Despite the added computational cost, these results highlight the necessity of the FR model for ensuring stronger identity protection.

Table A7. Quantitative results on the effectiveness and efficiency impact of the FR model.

Setting	Time/Image	FR (↓)
w/o FR Model	∼16s	0.534
w/ FR Model	∼20s	0.316