

A. Experiment Details

A.1. Hardware and DreamBooth Training Details

All the experiments are conducted on an Ubuntu 20.04.6 LTS (focal) environment with 503GB RAM, 10 GPUs (NVIDIA® RTX® A5000 24GB), and 32 CPU cores (Intel® Xeon® Silver 4314 CPU @ 2.40GHz). Python 3.10.12 and Pytorch 1.13 are used for all the implementations. For the DreamBooth full training mode, we use the 8-bit Adam optimizer [24] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ under bfloat16-mixed precision and enable the xformers for memory-efficient training. For calculating prior loss, we use 200 images generated from Stable Diffusion v2-1-base with the class prompt “a photo of a person”. The weight for prior loss is set to 1. For the instance prompt, we use “a photo of sks person”. During the meta-learning process, we regularly delete the temporary models and store the surrogates back to the CPU to save GPU memory. It takes about 3 GPU hours to craft perturbations for an instance under this strategy.

B. Baseline Methods and Metrics

B.1. Evaluation Metrics

In this section, we describe the evaluation metrics used in our experiments in more detail. For our proposed CLIP-IQAC, we calculate the CLIP score difference between “a good photo of [class]” and “a bad photo of [class]”. For calculating SDS and IMS-VGGNet, we leverage the APIs for face recognition and face embedding extraction in the deep face library [48]. In terms of graphical quality, we found that the commonly used metric, BRISQUE [32] is not a faithful metric when we conduct additional data transformations like Gaussian Filtering. We thus omit this score when presenting results in the main text. For instance, the BRISQUE score of a fully poisoned Dreambooth is better than the clean one, as shown in Tab. 8. Among all the metrics considered, we found that SDS and IMS-VGG are more aligned with our perception of evaluating Dreambooth’s personalized generation performance. The SDS score indicates whether a subject is presented in the generated image, while the IMS-VGGNet score measures the similarity between the generated image and the subject. Compared to graphical distortion, semantic distortion is more important when the user wants to prevent the unauthorized generation of their images. Overall, MetaCloak achieves the best performance among all the considered baselines.

Migrating the Metrics Variance. To migrate the variance for calculating metrics, we obtain the mean and standard variance in the following manner. For the instance i and its j -th metric, its k -th observation value is defined as $m_{i,j,k}$. For the j -th metric, the mean value is obtained with $\sum_{i,k} m_{i,j,k}/(N_i N_k)$, where N_i is the instance number for that particular dataset, and N_k is the image generation num-

ber. And the standard variance is obtained with the formula $\sqrt{\sum_{i,k} (m_{i,j,k} - \mu_j)^2 / (N_i N_k)}$, where μ_j is the mean of the j -th metric. Moreover, the Wilcoxon signed-rank significance test for the j -th metric between methods is conducted on sequence $\{m_{i,j,k}\}_{i,k}$. We found that these measures provide more faithful indicators of quality change compared to statistics solely over instances population.

C. More Experiments Results

C.1. Results under Standard Training

We present the results of MetaCloak under the standard training setting in Tab. 5. As we can see, MetaCloak can effectively degrade the DreamBooth’s personalized generation performance under the standard training setting. In terms of reference-based semantic metrics, we can see that our method is better in data protection than the previous SOTA in terms of IMS-VGG. Since VGGFaceNet is specially trained on facial datasets and thus more aligned with facial representation, we believe the results on IMS-VGG are more convincing than those on IMS-CLIP. In terms of other metrics, our method is also more effective than the previous SoTA.

C.2. More Visualizations

We visualize the generated images of Dreambooth trained on data perturbed by MetaCloak and other baselines in Fig. 11. As we can see, compared to other baselines, MetaCloak can robustly fool DreamBooth into generating images with low quality and semantic distortion under data transformations. In contrast, other baselines are sensitive to data transformation defenses. In this setting, DreamBooth’s generation ability is retained since the images generated are of high quality. These results demonstrate that MetaCloak is more robust in defense of data transformation.

C.3. More Datasets

To validate the effectiveness of MetaCloak on non-face data, we conduct additional experiments on the DreamBooth-official subject dataset [43], which includes 30 subjects of 15 different classes. Nine out of these subjects are live subjects (dogs and cats), and 21 are objects. Two inference prompts are randomly selected for quality evaluation. As shown in Tab. 6, the results demonstrate that our method can still successfully degrade the generation performance on those inanimate subjects.

C.4. Training DreamBooth on Replicate

We test the effectiveness of MetaCloak in the wild by training DreamBooth on the Replicate platform [40]. The Replicate platform is an online training-as-service platform that allows users to upload their own images and train DreamBooth on them. The generated image of the trained DreamBooth is

Dataset	Method	SDS ↓	IMS_{CLIP} ↓	IMS_{VGGNet} ↓	CLIP-IQAC ↓	LIQE ↓
VGGFace2	Clean	0.897 ± 0.302	0.814 ± 0.075	0.438 ± 0.658	0.456 ± 0.348	0.992 ± 0.088
	ASPL	0.341 ± 0.471	0.607 ± 0.083	-0.342 ± 0.822	-0.457 ± 0.208	0.352 ± 0.477
	EASPL	0.356 ± 0.475	0.586 ± 0.098	-0.455 ± 0.780	-0.489 ± 0.223	0.219 ± 0.413
	FSMG	0.423 ± 0.488	0.611 ± 0.077	-0.234 ± 0.843	-0.402 ± 0.213	0.312 ± 0.464
	AdvDM	0.933 ± 0.241	0.674 ± 0.081	0.111 ± 0.821	-0.177 ± 0.253	0.898 ± 0.302
	Glaze	0.966 ± 0.174	0.762 ± 0.057	0.541 ± 0.544	0.012 ± 0.262	0.992 ± 0.088
	PhotoGuard	0.967 ± 0.174	0.791 ± 0.064	0.548 ± 0.524	0.243 ± 0.284	1.000 ± 0.000
	MetaCloak	0.296 ± 0.448	0.662 ± 0.073	-0.051 ± 0.838	-0.380 ± 0.256	0.180 ± 0.384
CelebA-HQ	Clean	0.810 ± 0.389	0.763 ± 0.119	0.181 ± 0.784	0.470 ± 0.264	0.984 ± 0.124
	ASPL	0.809 ± 0.389	0.669 ± 0.079	-0.238 ± 0.825	-0.195 ± 0.275	0.883 ± 0.322
	EASPL	0.761 ± 0.421	0.656 ± 0.066	-0.346 ± 0.817	-0.226 ± 0.268	0.867 ± 0.339
	FSMG	0.684 ± 0.462	0.654 ± 0.084	-0.445 ± 0.781	-0.169 ± 0.255	0.773 ± 0.419
	AdvDM	0.849 ± 0.354	0.760 ± 0.062	0.331 ± 0.673	0.323 ± 0.287	1.000 ± 0.000
	Glaze	0.864 ± 0.338	0.784 ± 0.097	0.230 ± 0.756	0.486 ± 0.214	0.992 ± 0.088
	PhotoGuard	0.967 ± 0.174	0.805 ± 0.072	0.471 ± 0.557	0.424 ± 0.265	0.992 ± 0.088
	MetaCloak	0.407 ± 0.485	0.666 ± 0.071	-0.654 ± 0.670	-0.354 ± 0.191	0.406 ± 0.491

Table 5. Results of different methods under the *Stand. Training* setting with the corresponding std (\pm) on VGGFace2 and CelebA-HQ.

shown in Fig. 6. As we can see, MetaCloak can effectively degrade DreamBooth’s personalized generation performance in this setting. As can be seen, MetaCloak can effectively degrade the personalized generation performance of DreamBooth under both Full-FT and LoRA-FT settings, demonstrating that MetaCloak can seriously threaten Dreambooth’s online training services.

C.5. More Results on Adversarial Purification

In the SR defense, we first conduct image resizing with a scale factor of 1/4 and then use the SR model to reconstruct the image. In the TVM defense, we first resize the image to a size of 64x64 for computation feasibility. Then, we use the TVM model to reconstruct the image and then conduct two super-resolution processes and one resize process to align the image size with the original image. The DreamBooth trained on data purified by JPEG compression, SR, and TVM are shown in Fig. 8. As we can see, SR defense is the only one that can effectively purify the adversarial perturbation while maintaining the image quality. Compared to SR defense, TVM defense distorts the face significantly, and JPEG defense introduces some artifacts to the image.

Details of TVM optimization. We implemented the TVM defense in the following steps:

1. Resize the instance image to 64×64 pixels. This is done for the computational feasibility of the optimization problem.
2. Generate a random dropout mask $X \in \mathbb{R}^{64 \times 64 \times 3}$ using a Bernoulli distribution with probability $p = 0.02$. This

represents the pixel dropout rate.

3. Solve the TVM optimization problem:

$$\min_Z \|(1 - X) \odot (Z - x)\|_2 + \lambda_{TV} TV_2(Z), \quad (11)$$

where x is the original image, Z is the reconstructed one, and λ_{TV} is the weight on TV term. The first term ensures Z stays close to the original image x , and the second term minimizes the total variation, which reduces noise and enforces smoothness.

4. Reshape the optimized Z back to the original size $64 \times 64 \times 3$.
5. Conduct two super-resolution steps (with a resizing process in the middle) to upsample to 512×512 pixels.

More Results against Advanced Purification. To demonstrate the effectiveness of our method against the latest purification, we additionally present the performance of MetaCloak against two more recent purification methods, DiffPure [34] and IMPRESS [19] in Tab. 7. We set the purification iteration for IMPRESS as $N = 3k$, and the optimal perturbation time steps for DiffPure as $t^* = 8$. The results show that these purifications still can’t fully recover the original generation performance, demonstrating MetaCloak’s robustness.

C.6. Trade-off of Effectiveness and Stealthiness

To study the effectiveness of MetaCloak under different radii, we conducted experiments with different radii under the Trans. Training setting. As shown in Tab. 9, increasing

“a S^ backpack in the jungle” (Clean)*



“a S^ backpack in the jungle” (Protected by MetaCloak)*



“a S^ stuffed animal in the jungle” (Clean)*



“a S^ stuffed animal in the jungle” (Protected by MetaCloak)*



Table 6. More results of MetaCloak on the Dreambooth-official dataset under the Trans. Training setting. The identifier S^* is set to *sk*.










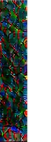










Clean Ref.	Training on clean images					Training on poisoned images			
									
									

Figure 6. Effectiveness of our method in the wild. Dreambooth training on the replicate platform under two training settings, including full fine-tuning and LoRA-based fine-tuning.

Defenses	SDS	IMS_{CLIP}	IMS_{VGG}	CLIP-IQAC	LIQE
×	0.401 ± 0.485	0.641 ± 0.118	-0.225 ± 0.852	-0.314 ± 0.266	0.445 ± 0.497
+IMPRESS	0.541 ± 0.493	0.691 ± 0.080	-0.074 ± 0.858	-0.307 ± 0.222	0.789 ± 0.408
+DiffPure	0.743 ± 0.430	0.640 ± 0.081	0.361 ± 0.673	0.020 ± 0.289	0.938 ± 0.242
Oracle*	0.903	0.790	0.435	0.329	0.984

Table 7. Resilience against advanced defenses on VGGFace2 under *Trans. Training*. Oracle* denotes training on clean data.

the radius can effectively improve the effectiveness of MetaCloak. However, when the radius is too large, the stealthiness of injected noise will also be compromised since some specific noise patterns will overwhelm the image content, as shown in Fig. 7. We conclude that the study of how to further improve the stealthiness of MetaCloak under large radii is an important future direction.

C.7. Resilience under Low Poisoning Ratio

To study the effectiveness of MetaCloak under low poisoning rates, we conduct experiments with different poisoning rates from {0%, 25%, 50%, 75%, 100%} under the two training settings. As shown in Tab. 8 and Fig. 10, increasing the poisoning rate can effectively improve the effectiveness of MetaCloak. However, when the poisoning rate is too low, the effectiveness of MetaCloak will be compromised since there is some knowledge leakage. How to effectively protect data under a low poisoning rate is an important future direction.

C.8. Improving Stealthiness with ReColorAdv

To further improve the stealthiness of our method, we explore replacing the ℓ_∞ -norm constrained attack with ReColorAdv [25], which generates adversarial images by applying a single pixel-wise function f_g with parameters g to every color value on the LUV color space. To make the perturbation imperceptible, f_g is bounded such that for each pixel \mathbf{c}_i , $\|f_g(\mathbf{c}_i) - \mathbf{c}_i\|_\infty \leq \epsilon$, and ϵ is set as 0.06, 0.08 and 0.10 respectively to observe the method’s performance under different restrictions. To enforce the bound, we optimize the parameters g with PGD, projecting g back to the ϵ ball after every gradient step. The results are shown in Fig. 13. As we can see from the figure, ReColorAdv crafts more uniform and global perturbations, which preserves dependencies between features such as the relationship between light and shadow and the shape boundaries. Unlike the ℓ_∞ -norm attack, the defense effect of ReColorAdv is reflected in the odd color of the generated images instead of collapsed patterns and strips. However, ReColorAdv becomes less effective when the inferring prompt is different from the prompt used in the perturbation crafting process, which is "A photo of sks person" in our case. In conclusion, we show that the ReColorAdv is promising to improve the stealthiness of our method further.

C.9. Understanding Why MetaCloak works

To understand why MetaCloak works, we visualize the learned noise and conduct wavelet analysis on the noise. We first visualize the original images, perturbed images, and the corresponding noise in Fig. 14. As can be seen from Fig. 15, the noise learned by MetaCloak is significantly sharper than the noise generated by the previous method. Thus, the pattern of our noise is less likely to be obscured and turn blurry when encountering defensive measures such as Gaussian transformation and is more likely to fool subject-driven generative models such as Dreambooth. We further plot the distribution of the scales of the learned noise in Fig. 16. As can be observed, the scales of the noise are polarized, clustering at both extreme values, and thus more resistant to perturbations.

Setting	Portion (clean/poison)	BRISQUE	SDS	IMS_{CLIP}	IMS_{VGG}	CLIP-IQA	CLIP-IQA-C
Stand. Training	Clean	14.610 \pm 4.560	0.958 \pm 0.060	0.781 \pm 0.072	0.314 \pm 0.427	0.818 \pm 0.045	0.397 \pm 0.113
	Mostly Clean(3/1)	14.700 \pm 10.405	0.928 \pm 0.047	0.785 \pm 0.042	0.460 \pm 0.325	0.799 \pm 0.109	0.410 \pm 0.203
	Half-and-half (2/2)	16.123 \pm 9.162	0.897 \pm 0.103	0.785 \pm 0.029	0.362 \pm 0.315	0.733 \pm 0.093	0.267 \pm 0.191
	Mostly Poison(1/3)	17.801 \pm 5.931	0.794 \pm 0.121	0.761 \pm 0.063	0.225 \pm 0.468	0.670 \pm 0.061	0.113 \pm 0.110
	Fully poisoned (4/0)	19.868 \pm 2.051	0.068 \pm 0.116	0.581 \pm 0.044	-0.299 \pm 0.640	0.360 \pm 0.085	-0.520 \pm 0.119
Trans. Training	Clean	19.063 \pm 4.070	0.934 \pm 0.092	0.756 \pm 0.104	0.299 \pm 0.357	0.750 \pm 0.083	0.286 \pm 0.042
	Mostly Clean(3/1)	24.385 \pm 9.997	0.911 \pm 0.077	0.794 \pm 0.044	0.474 \pm 0.216	0.763 \pm 0.068	0.346 \pm 0.129
	Half-and-half (2/2)	25.809 \pm 0.996	0.840 \pm 0.062	0.769 \pm 0.049	0.424 \pm 0.194	0.715 \pm 0.122	0.305 \pm 0.247
	Mostly Poison(1/3)	23.588 \pm 5.067	0.655 \pm 0.299	0.728 \pm 0.070	0.197 \pm 0.499	0.592 \pm 0.146	0.066 \pm 0.312
	Fully poisoned (4/0)	12.982 \pm 0.935	0.486 \pm 0.156	0.668 \pm 0.079	-0.277 \pm 0.636	0.534 \pm 0.058	-0.252 \pm 0.030

Table 8. Performance of MetaCloak under low poisoning rate. The number in the portion column denotes the portion of clean images.

Radius r	BRISQUE	SDS	IMS_{CLIP}	IMS_{VGG}	CLIP-IQAC	LIQE
Clean	21.783 \pm 12.540	0.903 \pm 0.291	0.790 \pm 0.076	0.435 \pm 0.657	0.329 \pm 0.354	0.984 \pm 0.124
4/255	19.810 \pm 8.828	0.832 \pm 0.369	0.760 \pm 0.094	0.207 \pm 0.815	-0.095 \pm 0.318	0.883 \pm 0.322
8/255	17.271 \pm 8.658	0.619 \pm 0.480	0.705 \pm 0.118	0.016 \pm 0.856	-0.278 \pm 0.311	0.625 \pm 0.484
16/255	17.794 \pm 6.707	0.393 \pm 0.476	0.668 \pm 0.077	0.067 \pm 0.842	-0.448 \pm 0.203	0.359 \pm 0.480
32/255	16.962 \pm 5.707	0.163 \pm 0.367	0.628 \pm 0.082	-0.081 \pm 0.851	-0.532 \pm 0.178	0.266 \pm 0.442

Table 9. Performance of MetaCloak under the Trans. Training setting with different perturbation radii.

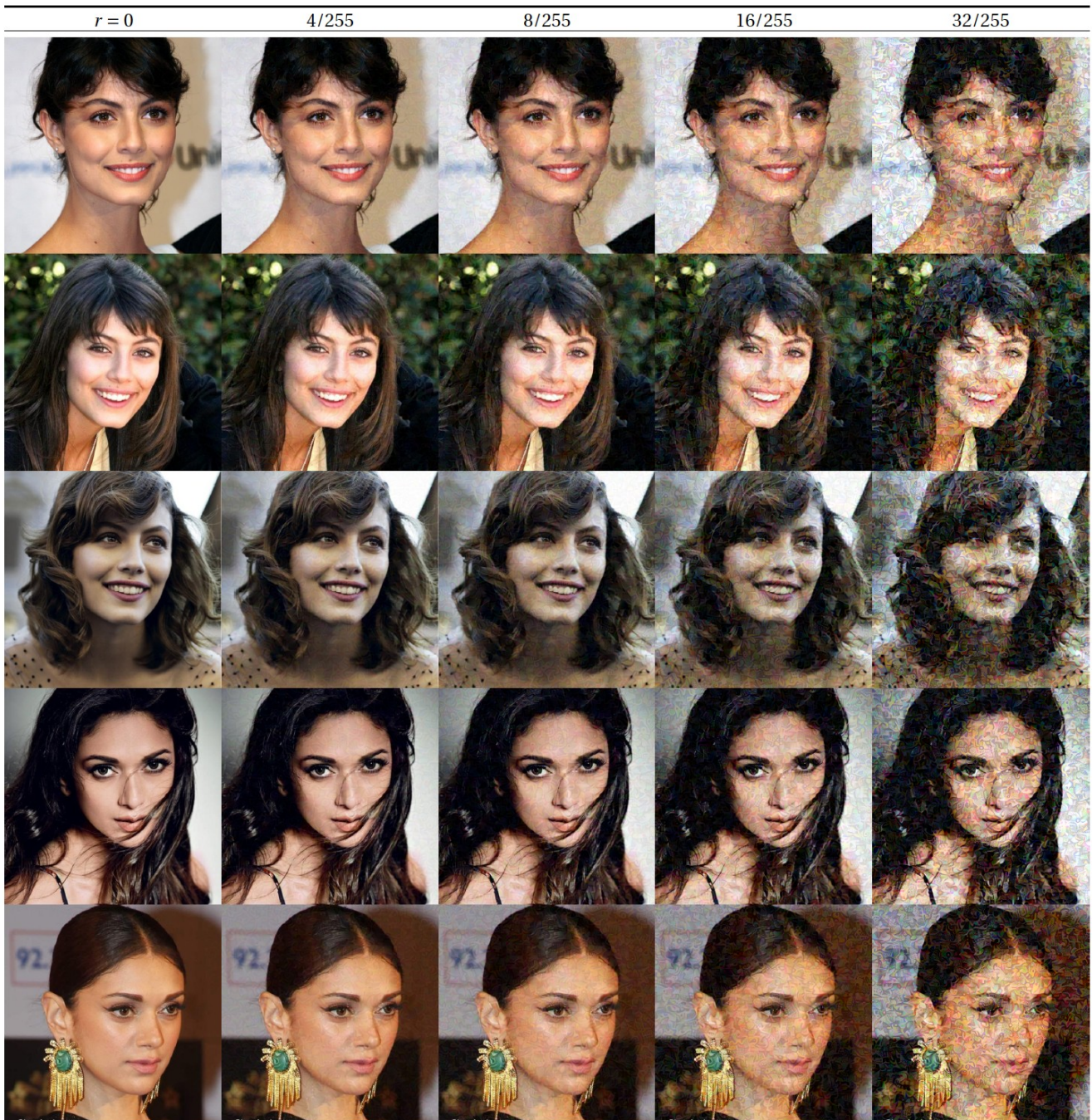


Figure 7. Visualization of perturbed images from VGGFace2 under different attack radii.

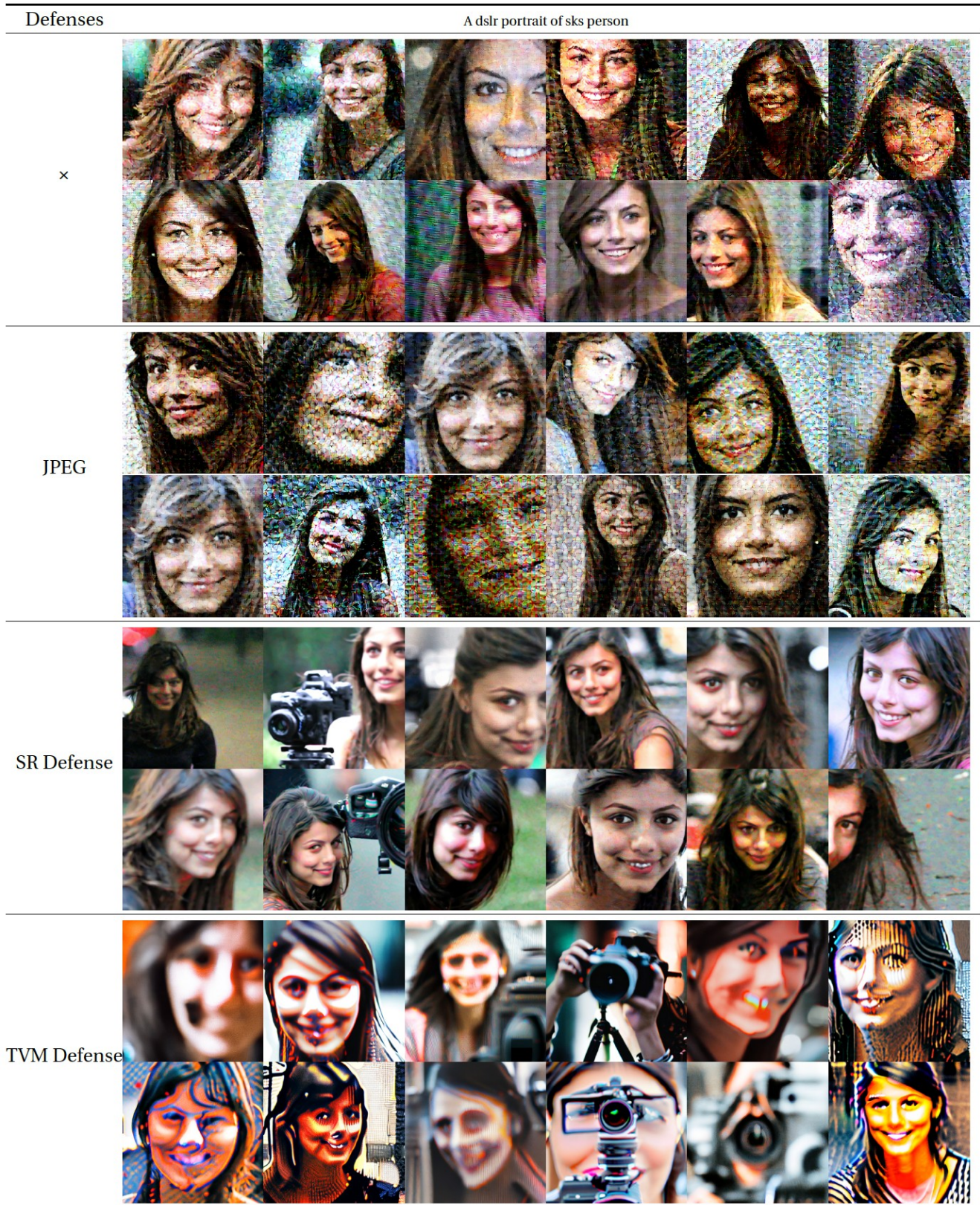


Figure 8. Visualizations of generated images of Dreambooth trained with various adversarial purifications under Trans. Training setting.

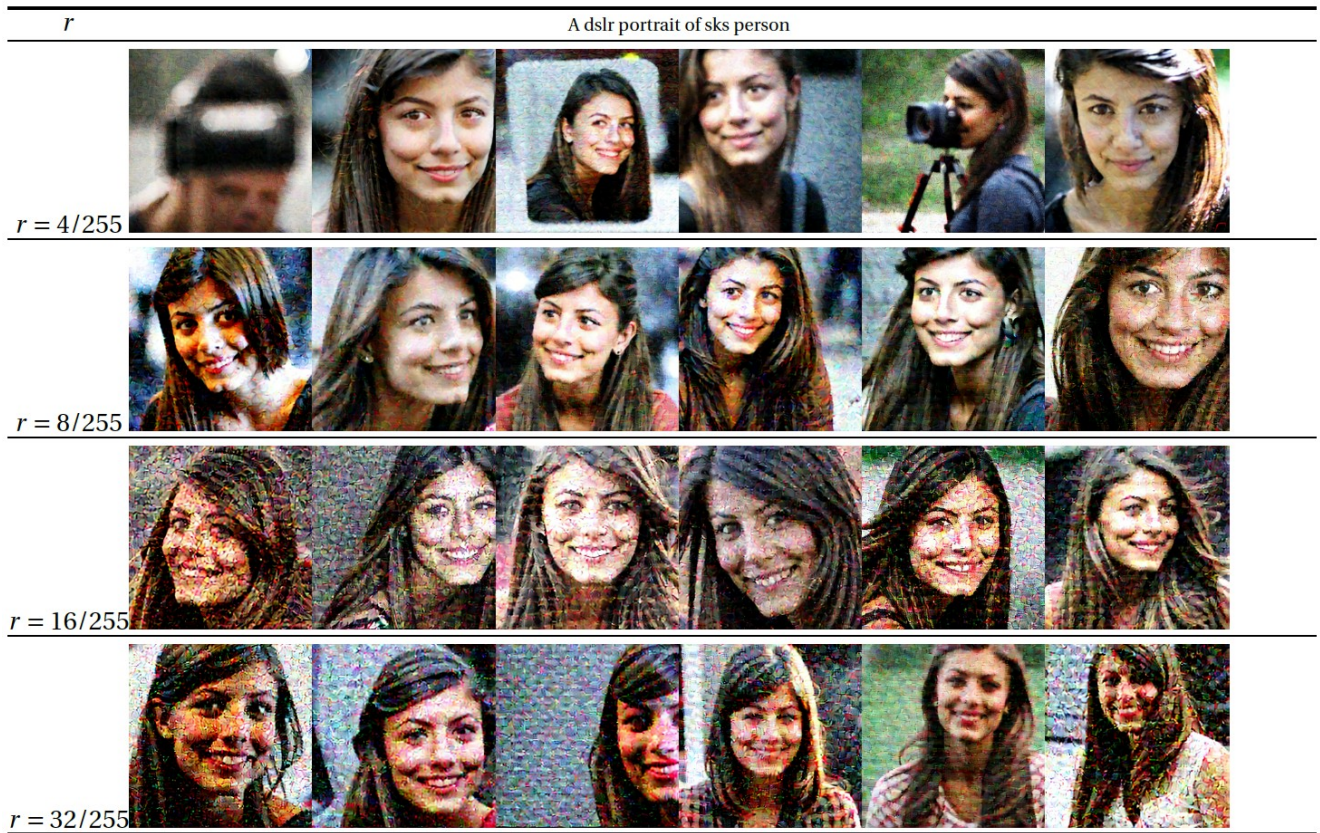


Figure 9. Performance of MetaCloak under different perturbation radii under Trans. Training setting.

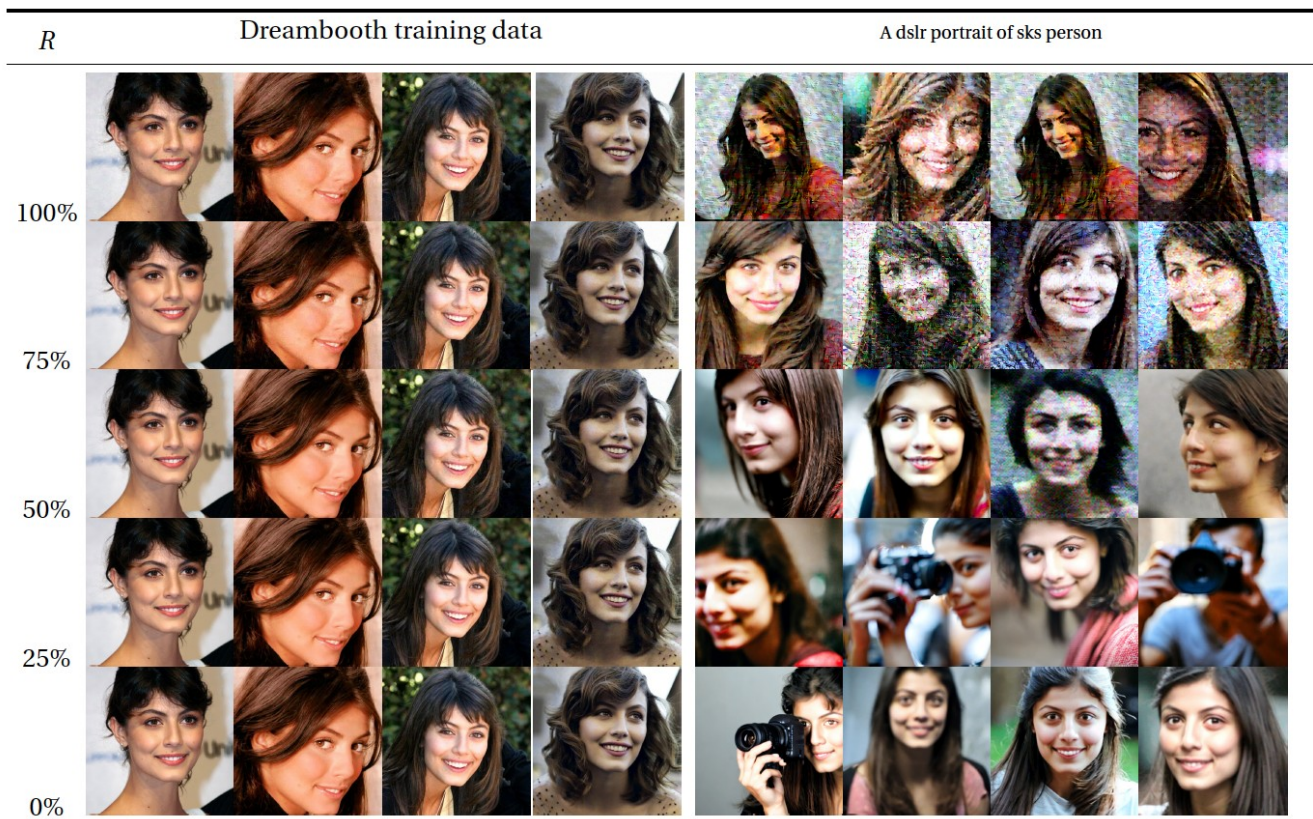


Figure 10. Performance of MetaCloak under different poisoning rates R . The Inferring prompt is “A dslr portrait of sks person”.



Figure 11. Visualization of Dreambooth’s generated images on VGGFace2. DreamBooths are trained on data perturbed by different methods under Trans. Training. The first column denotes the Dreambooth trained on clean data. The inferring prompt is “A photo of sks person”.



Figure 12. Effect of each component in MetaCloak on the quality of the generated images. The inferring prompt is “a dslr photo of a person”.

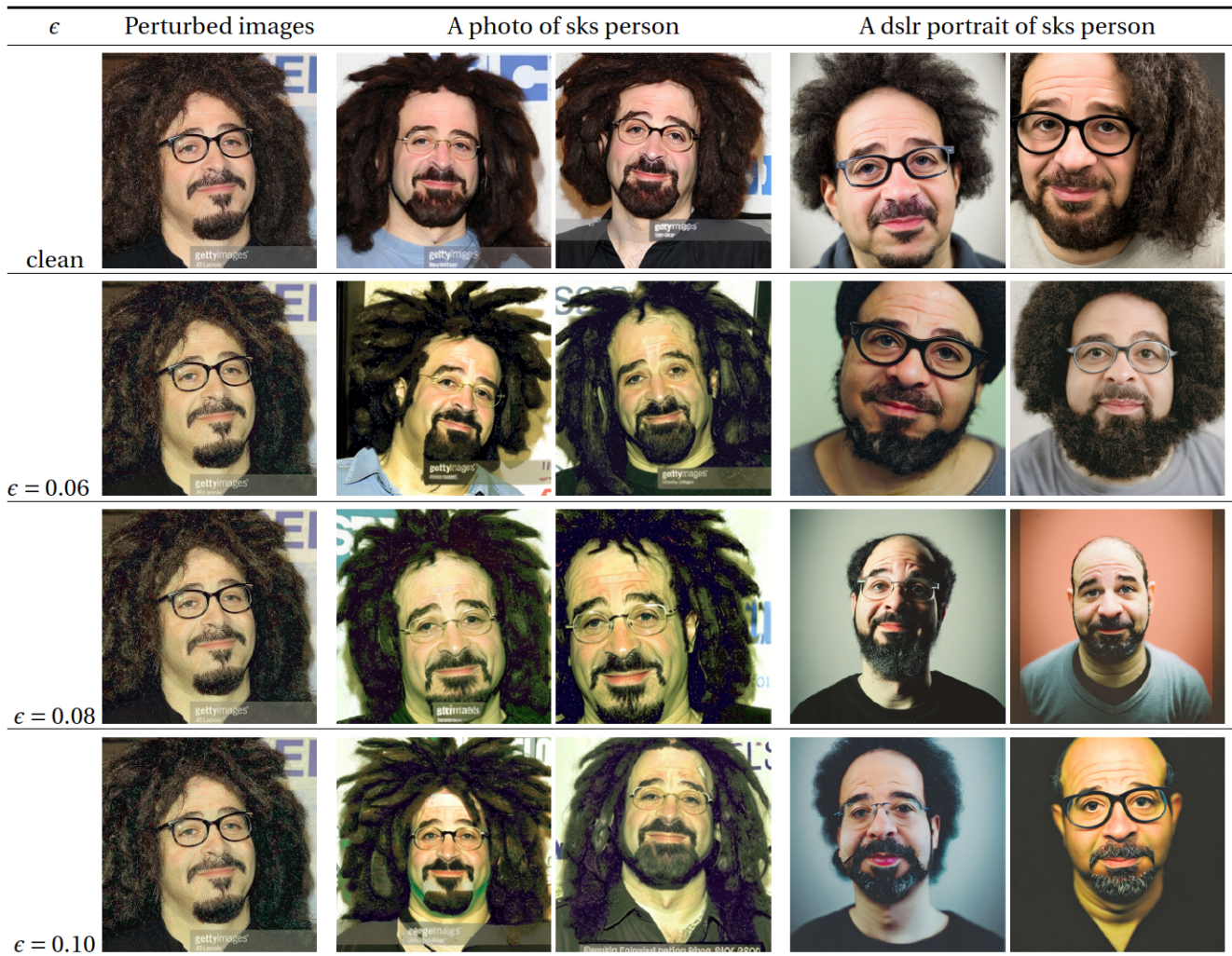


Figure 13. The Perturbed images by ReColorAdv and the generated images of DreamBooth trained on perturbed data.

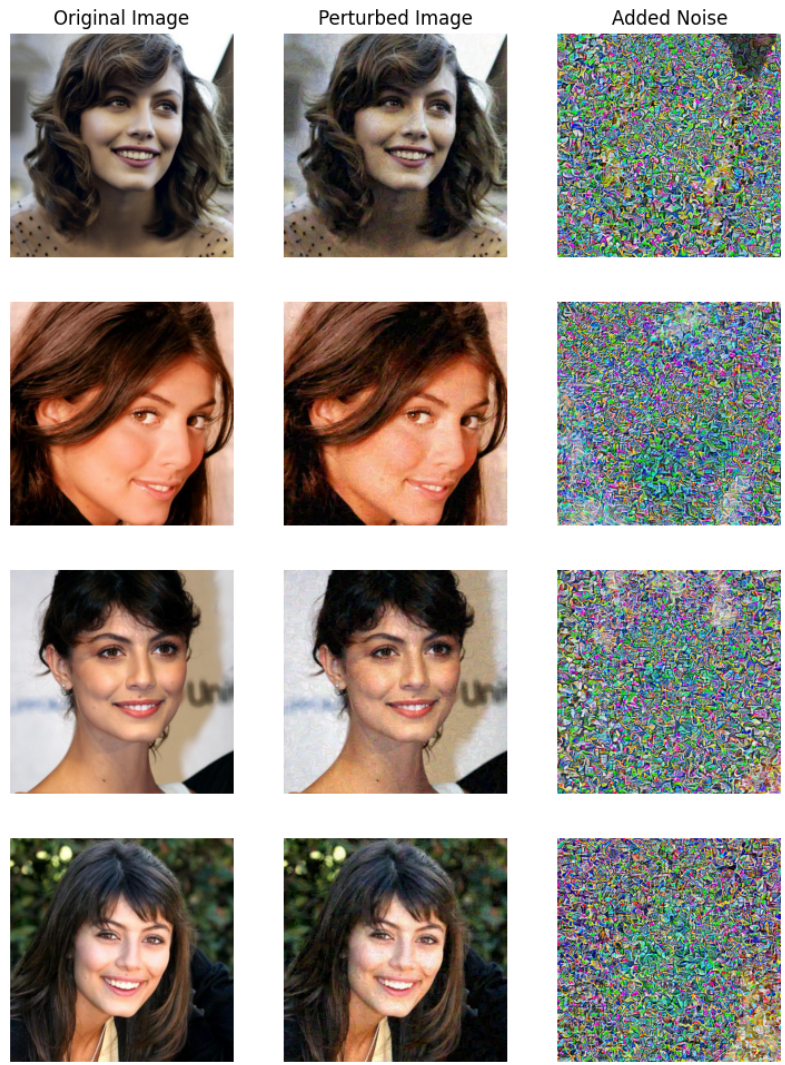
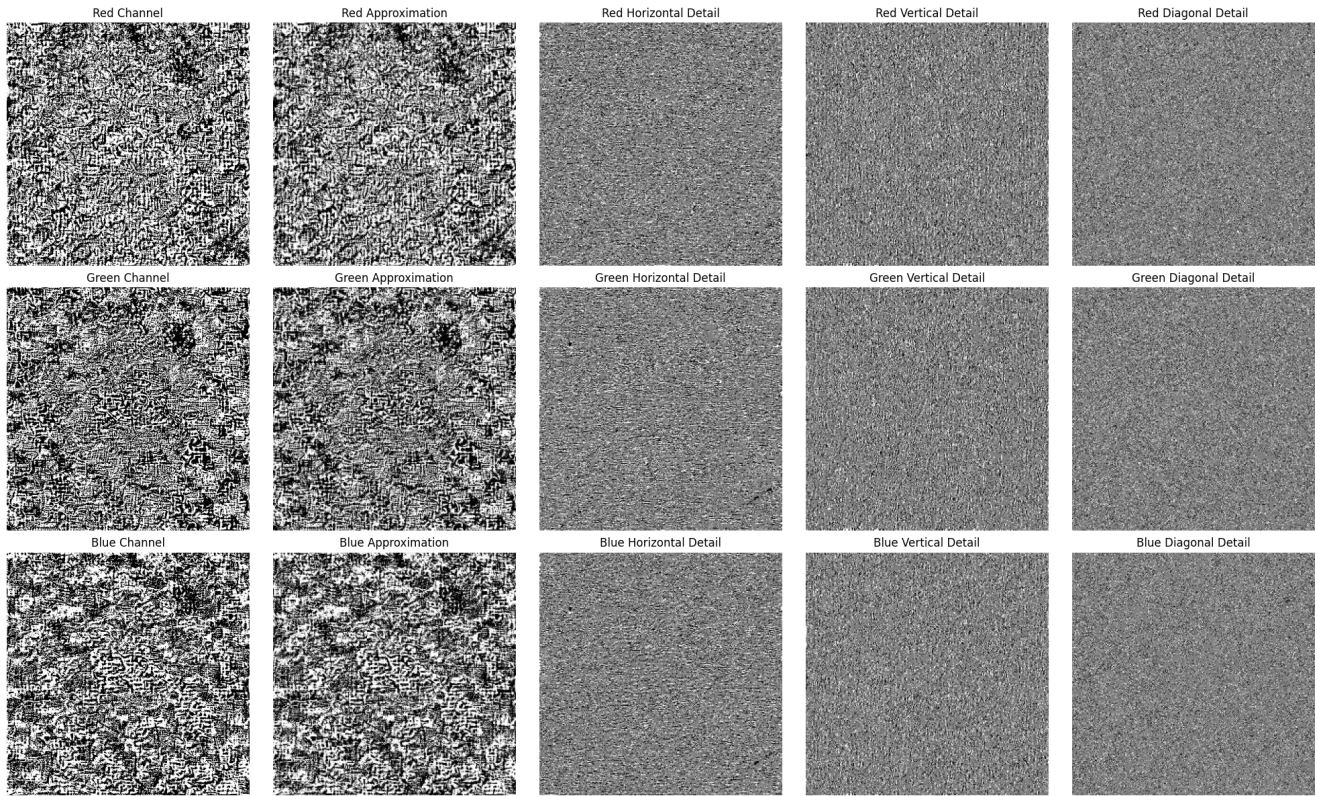
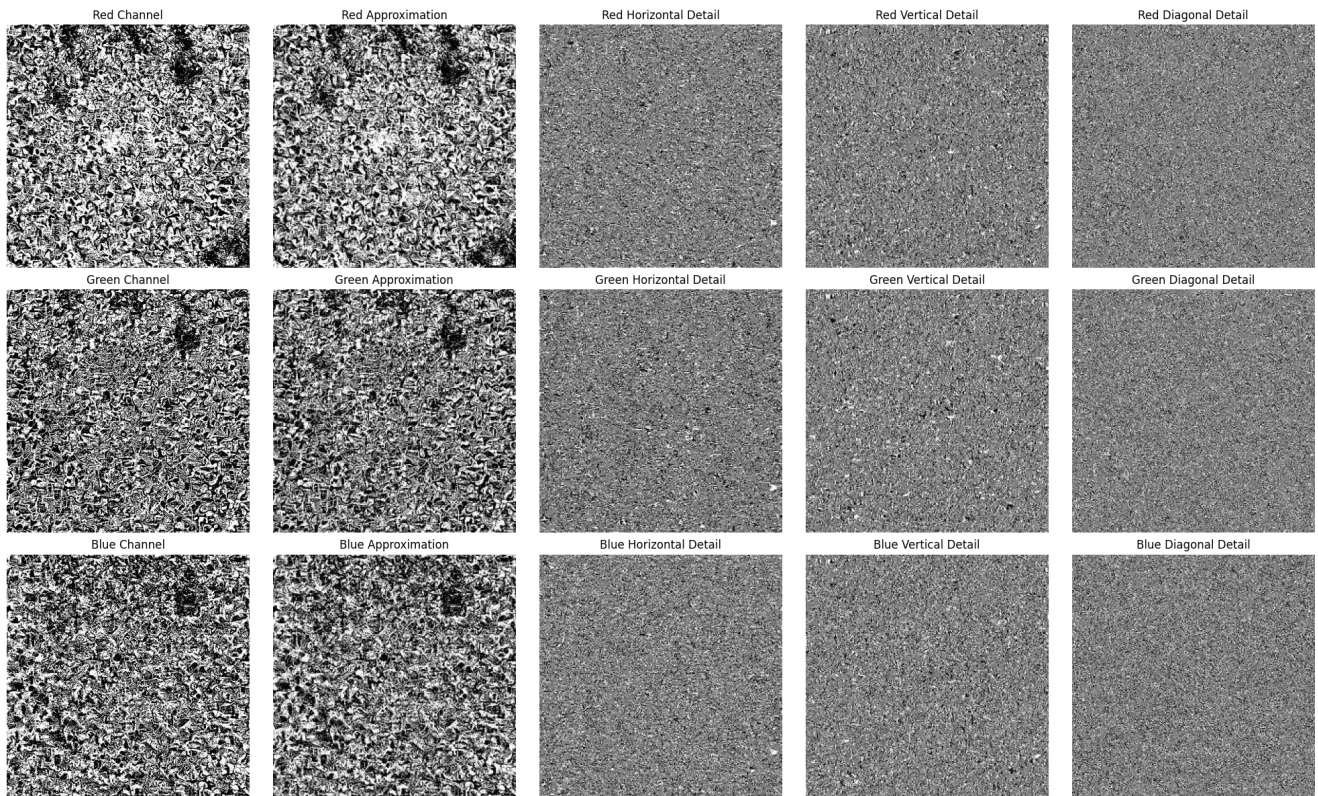


Figure 14. The original images, perturbed images, and the corresponding noise generated by MetaCloak on a sampled instance on the VGGFace2 dataset.



(a) ASPL



(b) MetaCloak

Figure 15. Comparison of wavelet decomposition of noise generated by ASPL and MetaCloak.

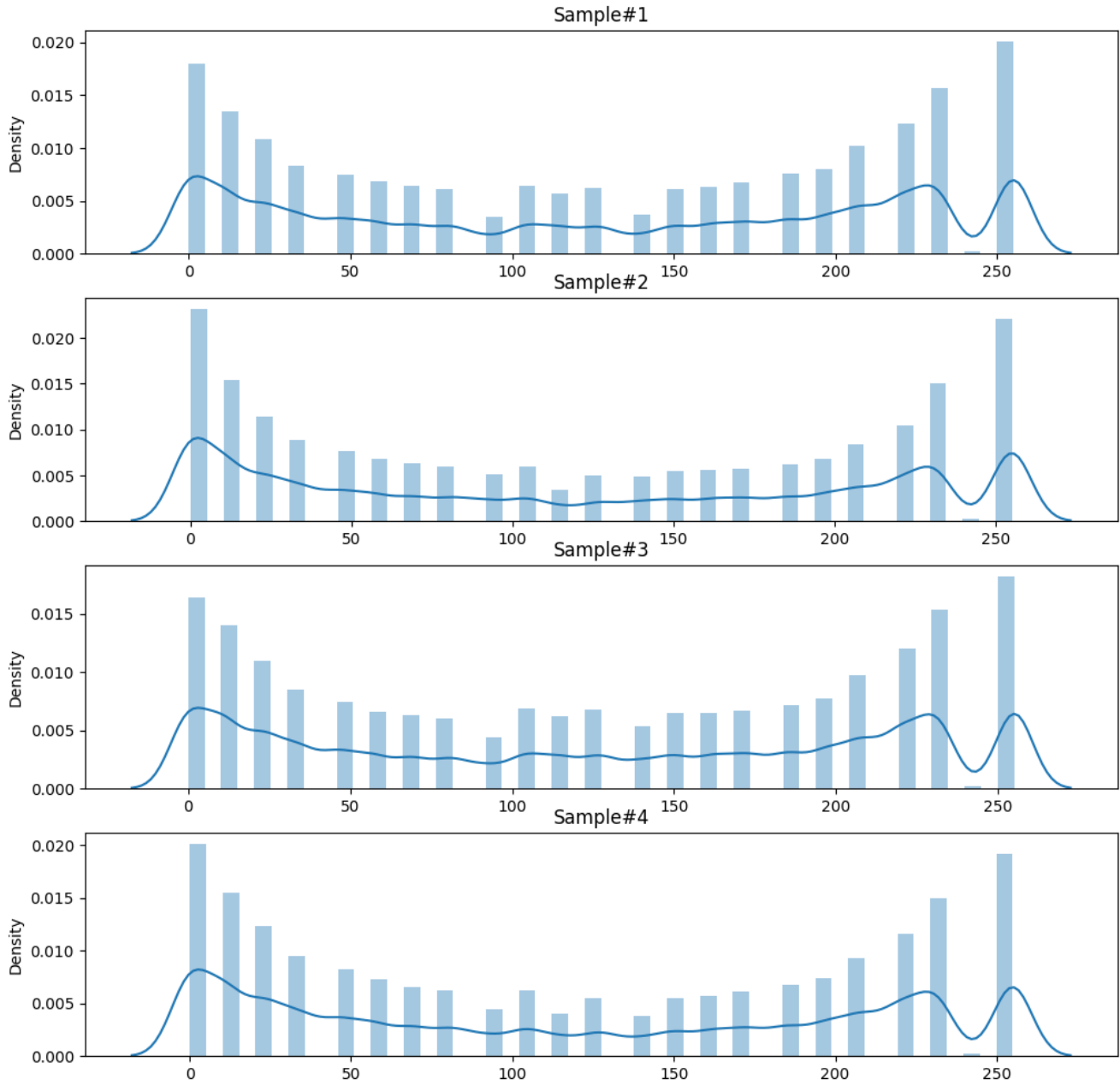


Figure 16. The density of noise scale generated by MetaCloak. We re-scale the pixel value of noise to the range $[0, 255]$ for visualization.