A. Human Performance

In computer vision and machine learning, human performance is typically seen as the benchmark for AI models. However, in the case of image attribution, the scenario reverses—AI significantly outperforms humans. This is highlighted by an experiment conducted by one of our co-authors, who has extensive experience with AI-generated images. Tasked with attributing 650 images to their correct source generators, the co-author achieved only a **37.23%** accuracy rate. This figure, while better than the 7.69% random chance level, is markedly inferior to the accuracy of our top AI classifier that has 90%+ accuracy. This outcome underlines the exceptional challenge of image attribution, where even well-informed individuals struggle. It shows the necessity of AI in assisting humans with tasks that are beyond their natural proficiency, emphasizing AI's potential to enhance human performance in specialized domains.

From the perspective of the human evaluator, differentiating between certain AI image generators and others can be nuanced yet discernible. The Latent Consistency Models (LCM) [46], at 2 and 4 steps, are notable for their occasional oversmooth artifacts, a result of undersampling, making them easier to identify compared to other models. DALL-E 3 [5] is distinguished by its tendency to produce surreal, cartoonish images, though these often exhibit repetitive patterns. DALL-E 2 [60], on the other hand, is characterized by a unique 'sharp' visual artifact, likely a consequence of its pixel diffusion process in the decoder, setting it apart from other models. Midjourney versions 5.2 and 6 [53] typically deliver the highest quality images, sometimes with a distinctive cinematic style.

Real images, however, are more straightforward to identify. One can often look at the detailed object regions—like hands and text—where AI-generated images tend to falter. The naturalistic photo style of real images also serves as a key differentiation factor from AI-generated content. Other generators, such as SD 1.5 [64], SD 2.0 [64], SDXL [58], SDXL Turbo [66], Kandinsky 2.1 [61], and Stable Cascade [56], present a greater challenge for human evaluators to distinguish due to the subtlety of their differences.

B. Data and Implementation Details

GPT-4 Generated Prompts. Building upon Section 3 of our main paper, this section describes the methodology behind generating creative and surreal prompts using GPT-4 [1]. As illustrated in Fig. 11 in the supplemental, our process begins with the formulation of system-level instructions directing GPT-4 to act as an assistant for writing text prompts. We then supply a specific context and a collection of several hundred exemplary prompts. This setup enables GPT-4 to synthesize and generate new, innovative prompts based on the provided examples and context.

Image Generation. We employed 12 T2I diffusion models to generate RGB images without watermarks, and the generated image sizes are as follows:

- **512 $\times$ 512:** Kandinsky 2.1, SD 1.1, SD 1.2, SD 1.3, SD 1.4, SD 1.5, SD 2.0, SDXL Turbo
- **1024 $\times$ 1024:** DALL-E 2, DALL-E 3, LCM (2 steps), LCM (4 steps), Midjourney 5.2, Midjourney 6, SDXL, Stable Cascade

We also use 5000 real images from the MS-COCO [41] 2017 validation set.

More Visualizations of Hyperparameter Variations. As an extension of Fig. 2 in the main paper, we show more image generations with hyperparameter variations in Fig. 12 in the supplemental.

Training Data. For Sections 4.1 and 5 in the main paper, we view image attribution as a 13-way classification task with 12 text-to-image diffusion models and 1 set of real images. An exception is the cross-domain generalization study, where we exclude real images as a 13^{th} class because there are no real images for the GPT-4 generated prompts. It's important to note that we use 3200 training, 450 validation, and 450 testing images per class.

For Section 4.2, we analyze four hyperparameters: Stable Diffusion checkpoint, scheduler type, number of sampling steps, and initialization seed. When training classifiers for SD checkpoints, schedulers, and sampling steps, we use 20000 training, 2500 validation, and 2500 testing images per class. For seeds, we use 3200 training, 450 validation, and 450 testing images per class.

For Section 4.3, we run inference using the EfficientFormer [40] trained with text prompts from Sec. 4.1. For SDXL Inpainting [58] and Photoshop Generative Fill [57], we use 450 images from each of the 13 classes. For Magnific AI [47], we use 10 images from each of DALL-E 3, Midjourney 6, and SDXL Turbo.

Data Augmentation. During training, we first resize each image to have a shorter edge of size 224 using bicubic interpolation, then center crop the image to size 224×224, and finally randomly flip the image horizontally with probability 0.5. During validation and testing, we only resize and center crop the images.

Image Attributors. We selected three network architectures for the image attribution task, and we use the code implementation from MMPretrain [13]. Our primary architecture is EfficientFormer-L3 [40] trained from scratch because it is a lightweight transformer. Moreover, we employ a pretrained, frozen transformer backbone attached to a linear probe (LP) or multilayer perceptron (MLP). The backbone is either CLIP ViT-B/16 [59] or DINOv2 ViT-L/14 [55], and the MLP consists of three linear layers with sigmoid activation and hidden dimension 256. For the linear probe and MLP classifier heads, there are 768 channels in the input

System: You are an AI assistant to help generate image text prompts.

User: I am trying to generate thousands of novel prompts for AI to generate images using tools, such as Midjourney, Stable Diffusion, and DALL-E. You can use your prior knowledge abour popular prompts, and I am also providing you a few hundred examplar prompts in below:

Cat looking at it and 'on top of the world, cyberpunk poster, in the style of Erik Jones, social media portraiture, sparth, vibrant color fields, graffiti and street art, highly detailed figures, hyper-detailed
……

Could you learn from these prompts and generate 30 more prompts that are sufficiently diverse and novel covering various content and styles.

GPT 4

Response:
A surreal depiction of a city skyline turning into a wave, in the style of Salvador Dali
A cybernetic eagle soaring over a futuristic cityscape, rendered in the style of Moebius
An old oak tree sheltering a boy, a dog and a horse, reflecting Johanna Basford's style
A steampunk table setting, complete with cogs and gears, in a dingy Victorian room, Max Ernst style
The Milky Way stretched across a desert landscape, in vivid neon colors, reflecting the style of Brandon Woelfel
An armored dinosaur in a dramatic rainforest battle, inspired by classic paleoart
An underwater castle made of coral, filled with merfolk, in the style of Ohnaka Akane
A miniaturized New York City, with tiny taxis and pedestrians, mimicking Tatsuya Tanaka's miniature calendar
A fiery phoenix rising from a cityscape, reflecting the style of Simon Stålenhag's retro futuristic scenes
A regal king standing heroically on a dragon's head, detailed in the style of Yoshitaka Amano
A romantic scene of two people kissing under cherry blossom trees, digital oil painting style of Leonid Afremov
A dark space station inhabited with multi-species aliens living harmoniously, vibrant color tones and intricate details, H.R Giger's surrealism style
A demon and an angel playing chess on the edge of a cliff during a storm, in the style of Francisco Goya
An 8k ultra-detailed mural of graffiti covering an entire city block, reflecting the style of Banksy
A hyper-detailed print of a wise owl reading a spellbook, in the style of Brian Froud's fantasies
……

Figure 11. An illustration of how we use the GPT-4 API to massively generate thousands of creative and surreal prompts.

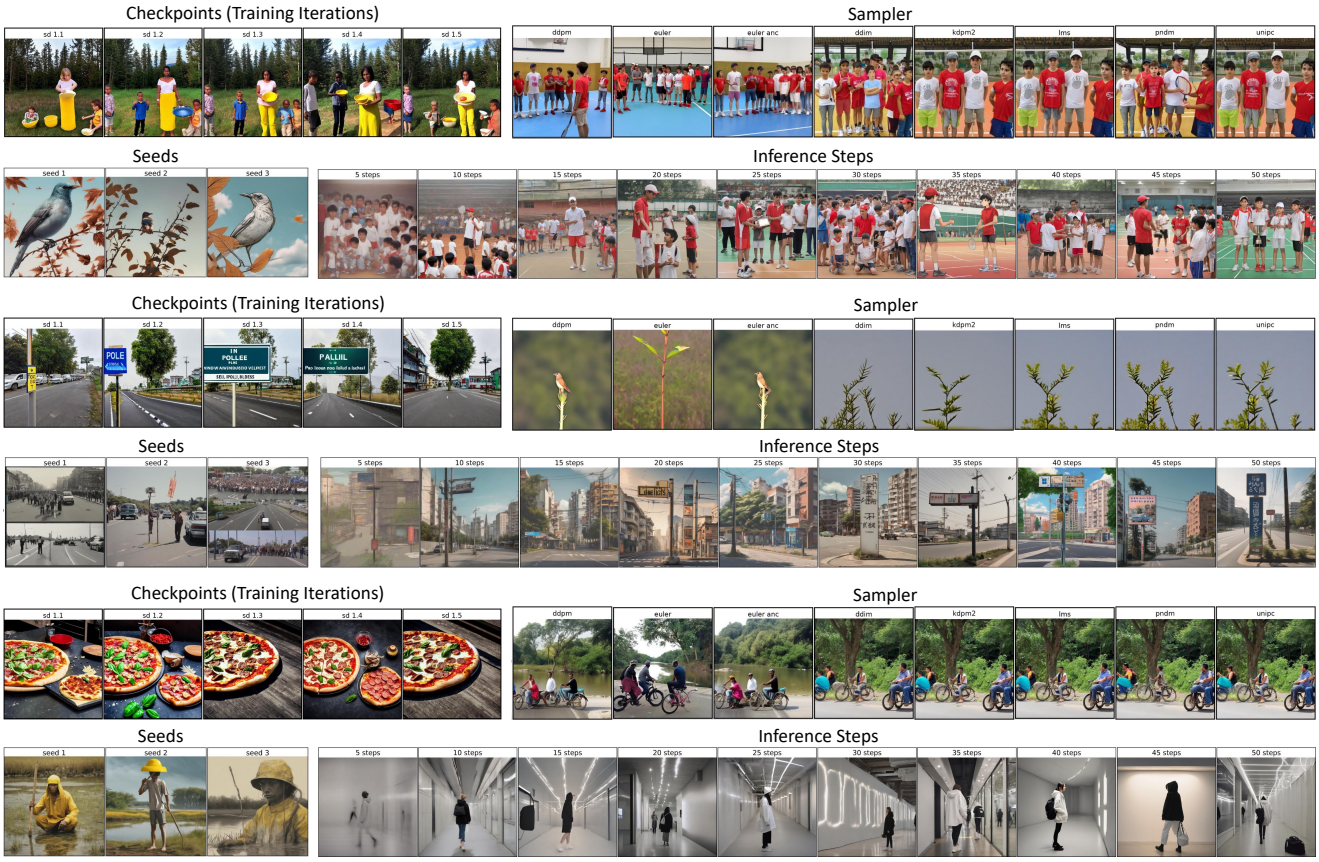

Figure 12. More examples showcasing the diversity in generated images influenced by varying hyperparameters: different model checkpoints within the same architecture, diverse scheduling algorithms, varied initialization seeds, and a range of inference steps.

feature map for CLIP+LP and CLIP+MLP, and 1024 channels for DINOv2+LP and DINOv2+MLP.

To train image attributors with text prompts, we compute text embeddings using a pretrained CLIP [59] text encoder. Then, we concatenate image embeddings from the backbone with text embeddings as input to the classifier head.

For all image attributors, we set a batch size of 128 and train for 2000 epochs. We use the checkpoint with the best

validation accuracy. Additionally, we utilize the AdamW optimizer [45] with learning rate 0.0002 and weight decay 0.05. The learning rate scheduler has a linear warm-up period of 20 epochs, followed by a cosine annealing schedule with a minimum learning rate of 0.00001.

Perspective Fields. We use the code implementation from [35]. Each input to the attributor trained on Perspective Fields has a size of $640 \times 640 \times 3$. The first 640×640 channel contains latitude values, and the next two 640×640 channels contain gravity values. We adapt the code from [35] to visualize the Perspective Field on a black image in Fig. 7 of the main paper.

How Gram Matrix Relates to Image Style. Gatys *et al*. [24] characterize the texture of an image by computing correlations between feature channels in each layer of a convolutional neural network. These correlations are given by the Gram matrix, which is the inner product of vectorized feature maps. Extending their method to image style, Gatys *et al*. [25] incorporate feature correlations, *i.e*. Gram matrices, from multiple layers of the network to obtain a multi-scale representation of the image that extracts texture details without the global arrangement. Intuitively, employing different layers of the network leads to style representations at varying scales because features capture more complex information in later network layers. Thus, we aggregate Gram matrices from three layers of a pretrained VGG network to train our image attributor on image style representations.

Adapting to New Text-to-Image Diffusion Models. Our work provides a seamless integration pathway for new generative models. For instance, to incorporate a new generator such as SD 2.0, one would simply generate approximately 5,000 images, add them to the existing training dataset, and retrain the models. This process typically requires around three days using a single RTX 4090 GPU. We intend to continually update our image attributor to include popular new open-source generators. Moreover, should there be a model not yet incorporated, anyone could replicate this integration process independently, as we plan to release all related code and datasets to the community.

C. Additional Experiments

C.1. Color Analysis

In addition to studying image style and image composition pattern, we examine whether different generators produce images with distinct color schemes. We use 100 images generated from a set of fixed prompts for our analysis. In Fig. 13, we visualize the density distribution of pixel values in each RGB color channel. We discover that Kandinsky 2.1 [61], Midjourney 5.2 [53], and Stable Cascade [56] often generate images with a wider range of pixel intensity values. In Fig. 14, we observe that these three generators often create images with glow and shadow effects, which can lead to higher and lower intensities.

C.2. Comparison of Frozen vs. Fine-tuned CLIP/ DINOv2 Backbone

In Section 4.1 of the main paper, we evaluated the accuracy of a frozen CLIP [59] backbone connected with a linear probe and MLP, and a frozen DINOv2 [55] backbone with a similar configuration. In this section, we compare using a frozen and fine-tuned backbone for the CLIP and DINOv2 linear probes. Table 5 indicates that a CLIP backbone provides marginally better performance than a DINOv2 backbone when the backbone is frozen. However, the reverse holds true when the backbone is fine-tuned.

C.3. Image Resolutions

The default EfficientFormer [40] takes inputs of size 224×224. We examine the performance of using five additional image resolutions between 128×128 and 1024×1024 for image attribution. As illustrated on the left side of Fig. 15, accuracy tends to increase as image resolution increases.

C.4. Cropped Image Patches

Our previous experiments use most, if not all, image pixels for the image attribution task. We also explore the opposite: how few pixels are necessary to achieve good performance? Inspired by [11, 87], we crop a single patch of each image and then train EfficientFormer [40] on these patches instead of the full-sized images. Specifically, we first resize each original image to have a shorter edge of size 512, then center crop the image to create a patch of size $k \times k$, and finally resize the patch to 224×224. We utilized $k = [2, 4, 8, 16, 32, 64, 128, 256]$ and resized images using bicubic interpolation. On the right side of Fig. 15, we see that accuracy increases with image patch size. Remarkably, even training an image attributor on 2×2 patches can lead to 22.29% accuracy, which is well above the random chance accuracy of 7.69%.

C.5. Potential Application of Model Stealing

It's important to note that our research might facilitate 'model stealing,' or the reverse engineering of a model's architecture. As an initial experiment, we projected 20 images generated from each of the four most recent non-open-source models—'Adobe Firefly Image 3' [2], 'SD 3' [73], 'SD 3 Turbo' [73], and 'Meta AI Imagine' [52]—into the t-SNE feature embedding space of our pretrained image attributor. As illustrated in Fig. 16, we observe that images from 'Adobe Firefly Image 3' appear similar to those from 'Midjourney 5.2' and real images. Meanwhile, 'SD 3' and 'SD 3 Turbo' are closer to 'Stable Cascade' and 'Midjourney 6', and 'Meta AI Imagine' largely overlaps with 'DALL-E 3'. This comparative analysis could lay the groundwork for inferring the architectures of non-open-source models based on those already known.

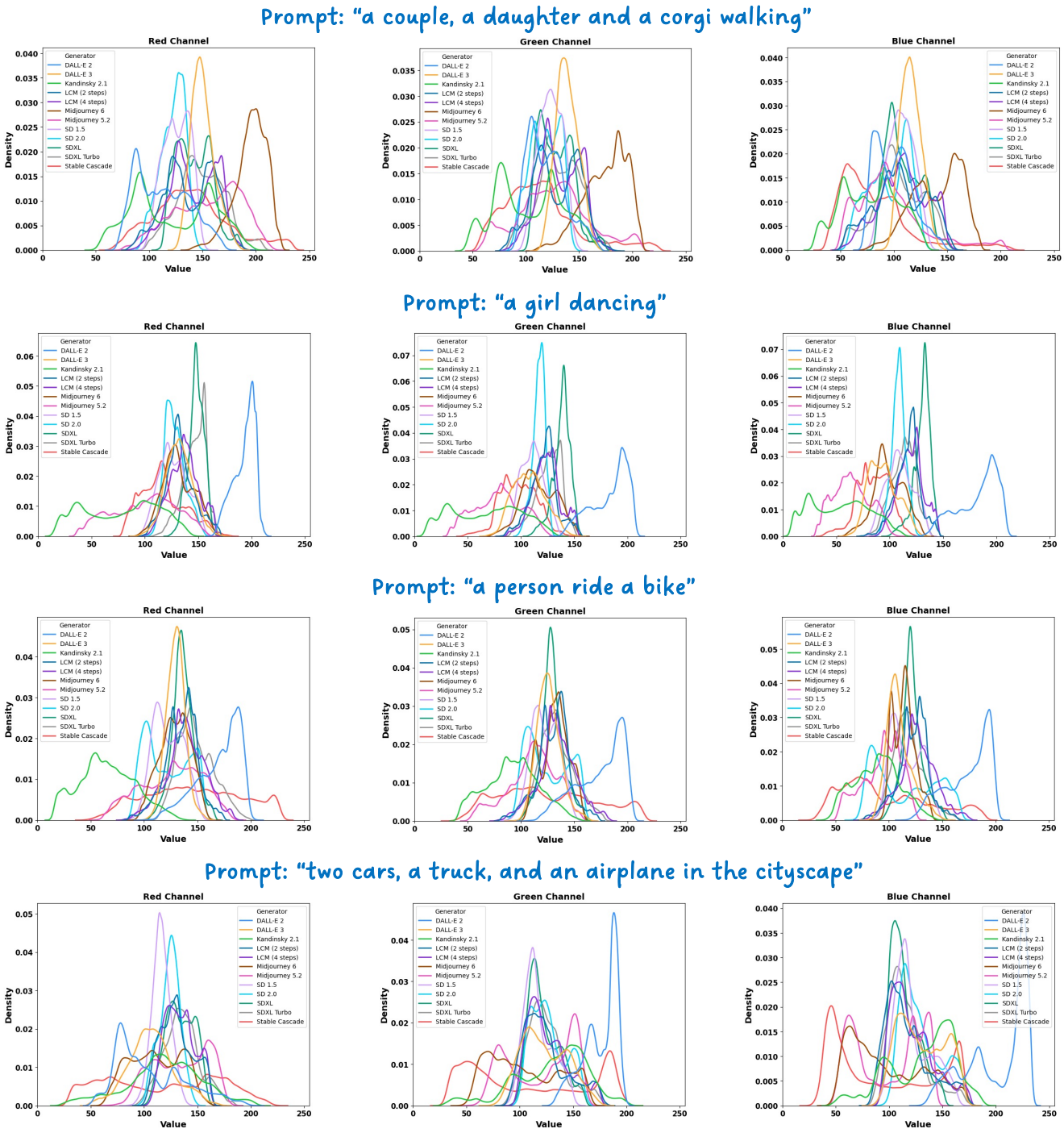

Figure 13. Density distribution of pixel values in RGB color channels after averaging 100 images for each prompt and generator. Kandinsky 2.1 [61], Midjourney 5.2 [53], and Stable Cascade [56] tend to create images covering a wider range of pixel intensities.

D. Elaboration on Results in the Main Paper

In this section, we expand upon the results from the experiments performed in the main paper. Figure 17 and Table 6 showcase the confusion matrices and evaluation metrics for the image attributors in Sec. 4.1. Furthermore, Figure 18 and Table 7 present the confusion matrices and evaluation metrics for the cross-domain generalization study in Sec. 4.1. Additionally, Figure 19 illustrates the confusion matrices for post-editing enhancements in Sec. 4.3. Lastly, Figure 20 visualizes the averaged segmentation masks across generators for two additional prompts, which is an extension of our image composition analysis in Sec. 5.

Takeaways from high-frequency perturbations. Prior works have predominantly claimed that classifiers in tasks like 'real vs. fake' and image attribution primarily learn from discriminative information in the high-frequency do-

Figure 14. Visualization of 100 images averaged together for each prompt and generator. Consistent with our observations in Fig. 13, we see that Kandinsky 2.1 [61], Midjourney 5.2 [53], and Stable Cascade [56] often produce images with glow and shadow effects.

	CLIP + LP		DINOv2 + LP	
Backbone	Frozen	Fine-tuned	Frozen	Fine-tuned
Accuracy	70.15%	95.31%	67.68%	96.67%
Precision	69.95%	95.51%	67.36%	96.71%
Recall	70.15%	95.32%	67.68%	96.67%
F1	70.00%	95.34%	67.45%	96.67%

Table 5. Quantitative comparison of using a frozen or fine-tuned backbone to train CLIP [59] and DINOv2 [55] linear probes. CLIP achieves higher accuracy than DINOv2 when the backbone is frozen, but the opposite is true when the backbone is fine-tuned.

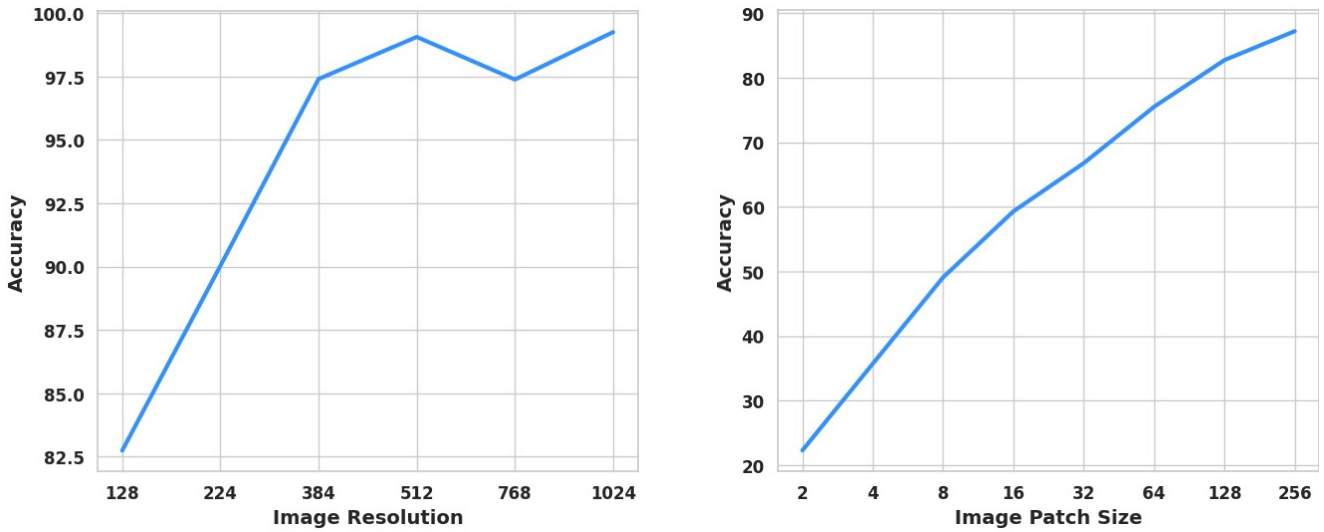

Figure 15. **Left:** Accuracy of our EfficientFormer [40] image attributor across six image resolutions on the 13-way classification task. In general, accuracy increases as image resolution increases. **Right:** Accuracy of EfficientFormer across eight image patch sizes. Interestingly, using 2×2 image patches can achieve 22.29% accuracy, whereas the probability of randomly guessing the correct generator is $\frac{1}{13}$, corresponding to 7.69%.

main. While we concur that high-frequency details can be crucial for discrimination, our work has demonstrated that even when these details are altered, the classifier can still identify highly discriminative features and attain decent ac-

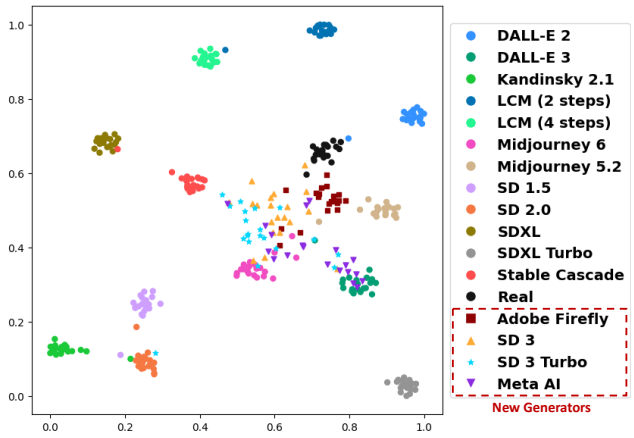

Figure 16. The t-SNE visualization of 4 unseen new generators in the feature space of our pretrained image attributor.

curacy. Our finding does not contradict earlier claims, but rather suggests a shift in perspective, showing that reliance only on high-frequency details may not be necessary.

E. Grad-CAM Visualizations

Figure 21 showcases the Grad-CAM [26, 68] heatmaps for image attributors trained on various image types, including the original RGB images, images after high-frequency perturbations, and mid-level representations. We observe that the image attributors trained on RGB images and images after high-frequency perturbations tend to pay attention to smooth image regions, such as the sky or ground. Nonetheless, even though the attributors focus on varied image regions, it remains difficult to explain how they make their decisions for each image.

F. Broader Impacts

We acknowledge that text-to-image diffusion models pretrained on large-scale, uncurated web data may produce biases and errors. Additionally, we use text prompts that are based on captions of MS-COCO [42] images and GPT-4 [1] outputs, which may generate images of people.

	E.F. (scratch)	CLIP+LP	CLIP+MLP	DINOv2+LP	DINOv2+MLP
Accuracy	90.03/90.96	70.15/71.44	73.09/74.19	67.68/69.44	71.33/73.08
Precision	90.07/90.98	69.95/71.30	73.13/74.12	67.36/69.09	71.20/72.91
Recall	90.03/90.96	70.15/71.44	73.09/74.19	67.68/69.44	71.33/73.08
F1	90.04/90.96	70.00/71.25	73.07/74.12	67.45/69.17	71.23/72.93

Table 6. Additional quantitative evaluation of image attributors for 13-way classification, consisting of 12 generators and a set of real images. The values (percentages) represent training each attributor *Without / With* text prompts.

	Train on MS-COCO	Train on GPT-4	Train on Both
Accuracy	89.04/69.24	71.07/79.35	85.78/81.06
Precision	89.07/70.38	71.81/79.29	85.88/80.87
Recall	89.04/69.24	71.07/79.35	85.78/81.06
F1	88.99/68.44	71.06/79.21	85.78/80.86

Table 7. Cross-domain generalization in image attributors. The amount of training and testing data was kept consistent across trials, and an equal number of images was sourced from MS-COCO and GPT-4 prompts for the 'Train on Both' trial. The values (percentages) represent testing on images from *MS-COCO / GPT-4* prompts.

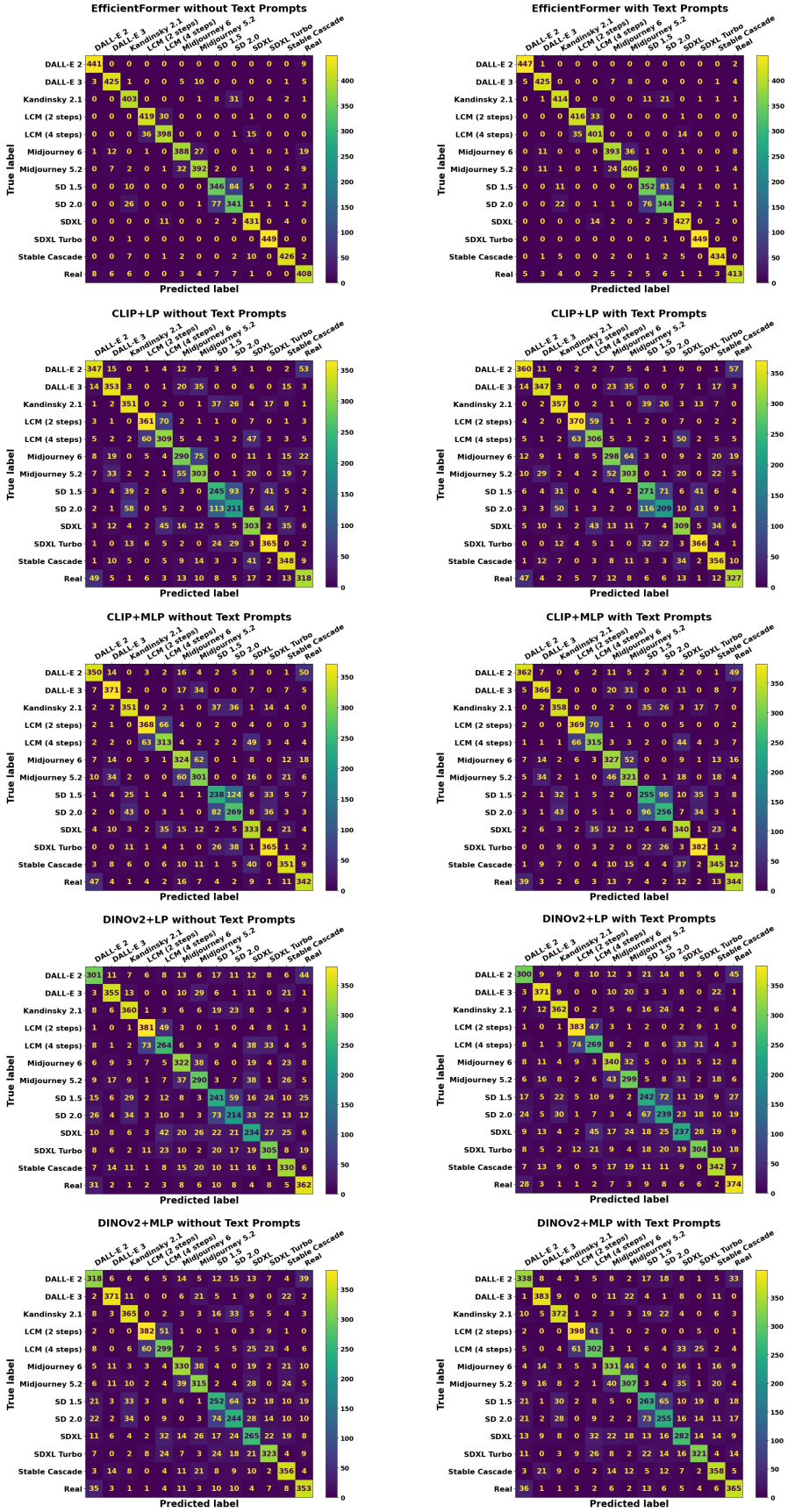

Figure 17. Confusion matrices for image attributors in Sec. 4.1. The backbone for the CLIP and DINOv2 models is frozen.

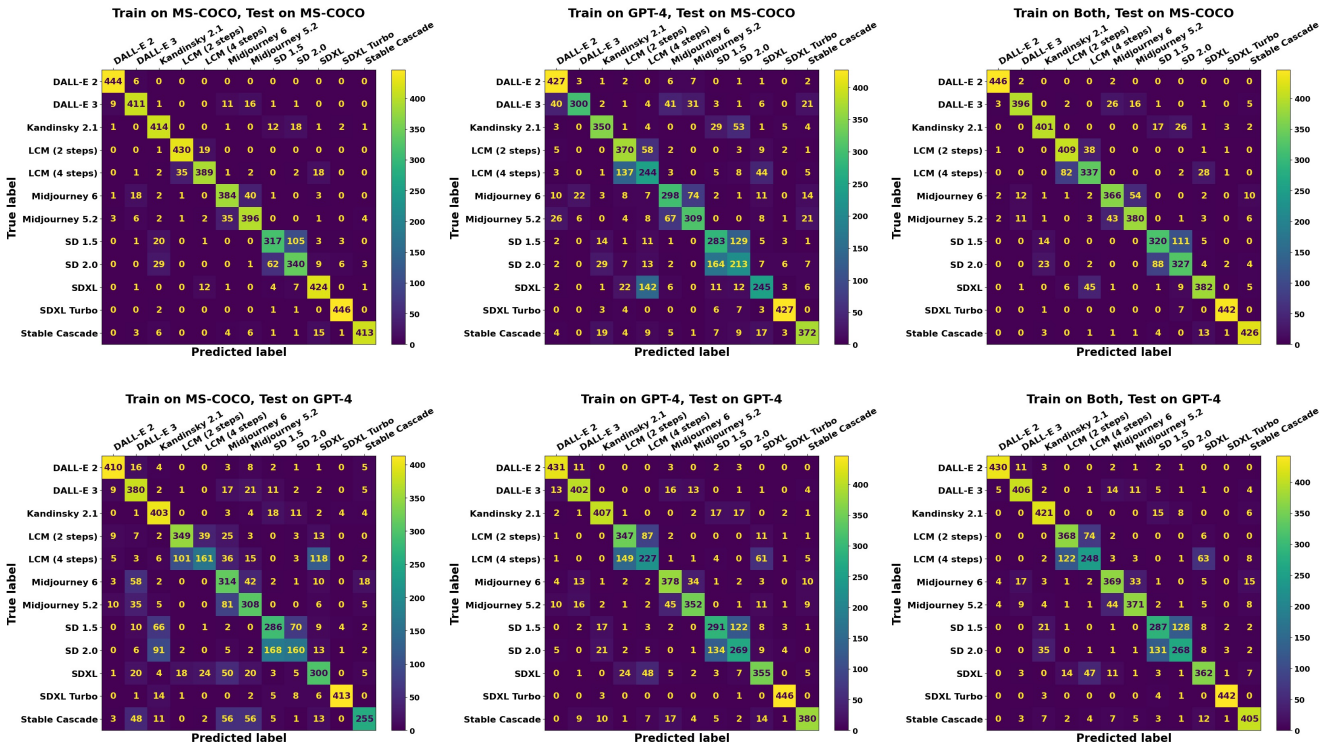

Figure 18. Confusion matrices for cross-domain generalization in Sec. 4.1.

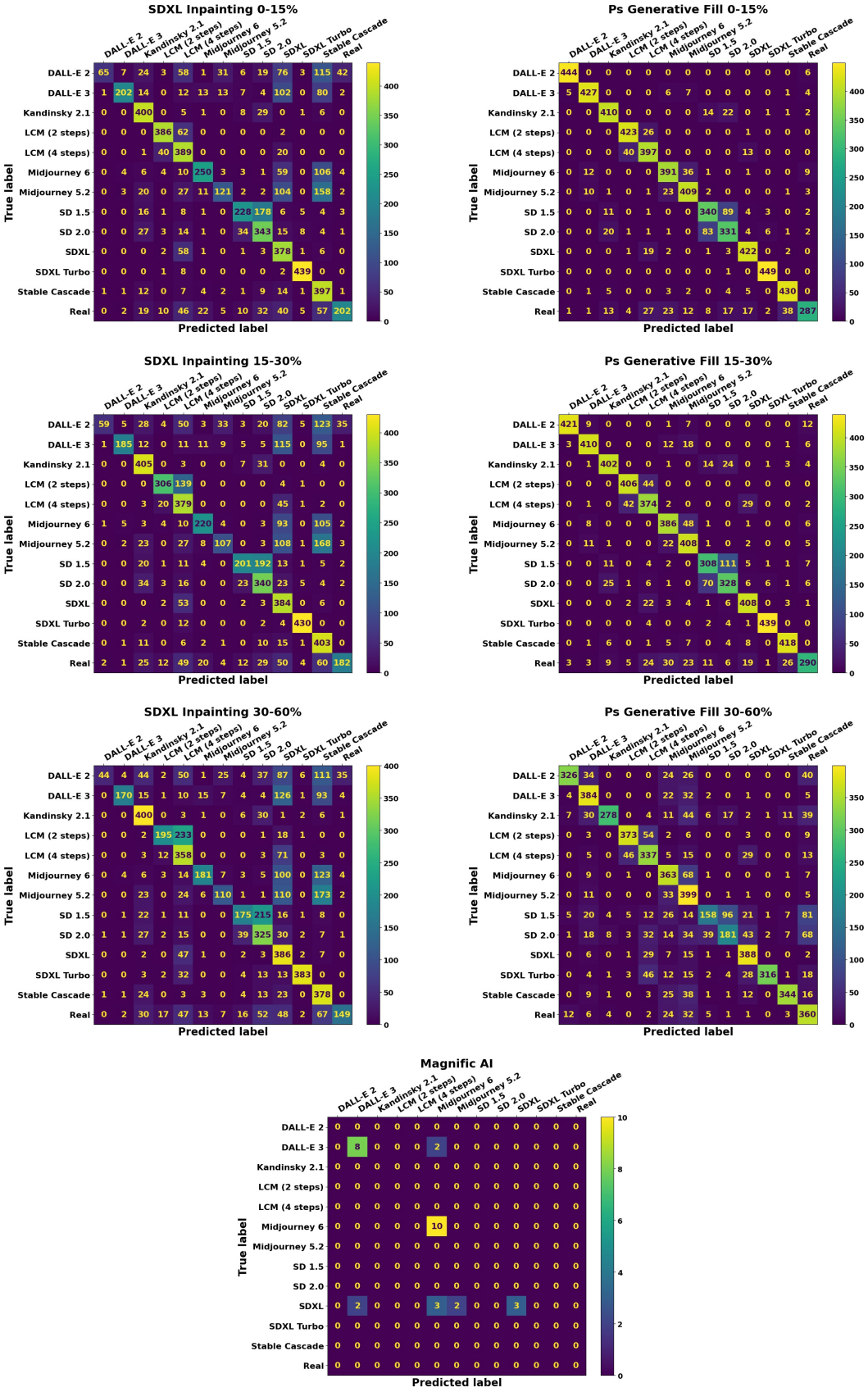

Figure 19. Confusion matrices for evaluating on post-edited images in Sec. 4.3.

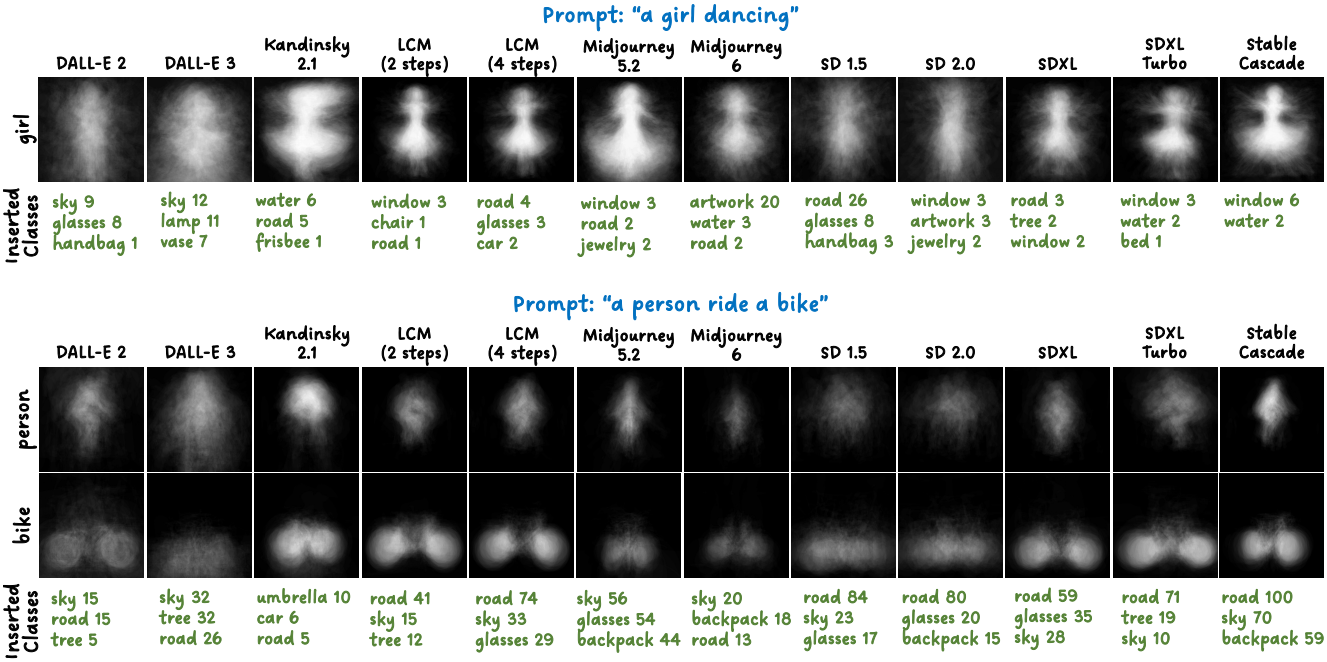

Figure 20. Additional image composition analyses across generators. We show the averaged segmentation masks for each semantic class indicated on the left side. We also list the top three inserted classes and the number of images (out of 100) with these classes.

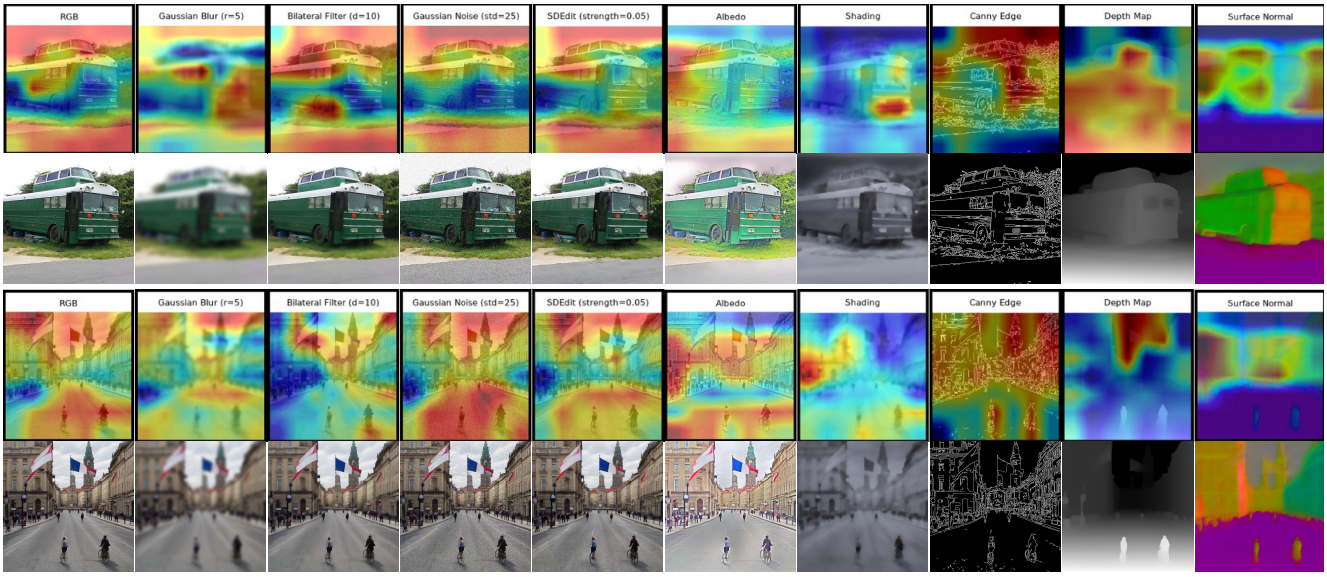

Figure 21. Grad-CAM [26, 68] visualizations for image attributors trained on each image type, where each column represents a distinct attributor. The first and third rows illustrate the Grad-CAM heatmaps overlaid on the input images. The second and fourth rows show the input images without Grad-CAM. The first example on the top is based on a real image from MS-COCO [41], while the second example on the bottom is based on a fake image generated by SDXL Turbo [66]. We notice that the attributors trained on RGB images and images after high-frequency perturbations often focus on relatively smooth image regions, such as the sky or ground.