

Supplementary Material for PolyJuice

Appendix

In our main paper, we propose PolyJuice for performing unrestricted adversarial attacks on synthetic image detectors. Here, we provide some additional details to support our main results. The appendix section is structured as follows:

1. Implementation Details in § A
 - (a) T2I Model Settings in § A.1
 - (b) Computing Steering Directions in § A.2
 - (c) Hyperparameter Search for λ_t in § A.3
 - (d) Calibrating the Confidence Threshold of an SID in § A.4
 - (e) Implementation of Spectral Fingerprint Analysis in § A.5
 - (f) Implementation of Projected Steering Directions in § A.6
 - (g) More Visualizations on the Effect of PolyJuice in Image Generation Process in § A.7
2. Qualitative Analysis of Generated Attacks in § B
 - (a) Common Patterns in Successful Attacks in § B.1
3. Additional Results in § C
 - (a) Spectral Fingerprint Analysis in § C.1
 - (b) Realness Shift in T2I Latent Space in § C.2
 - (c) Additional image-space visualizations on the effect of PolyJuice § C.3

A Implementation Details

A.1 T2I Model Settings

We provide the generation settings for SDv3.5, FLUX_[dev], and FLUX_[sch] in Tab. 5. All models are available on Huggingface Models [6] with the corresponding Model ID provided in Tab. 5.

Table 5: Settings for T2I Models.

Model	Model ID	Num Steps	Guidance Scale	Max Seq. Length
SDv3.5	stabilityai/stable-diffusion-3.5-large	28	3.5	256
FLUX _[dev]	black-forest-labs/FLUX.1-dev	50	3.5	512
FLUX _[sch]	black-forest-labs/FLUX.1-schnell	4	0	256

A.2 Computing Steering Directions

In Algorithm 1, we provide a pseudocode for calculating the steering direction δ_t at a given timestep t . Algorithm 1 requires precomputed T2I latents $Z_t \in \mathbb{R}^{N \times d_Z}$, where N is the number of generated images and d_Z is the dimensionality of the T2I latent space. For efficient calculation of the top- k eigenvalues and corresponding eigenvectors, we use the LOBPCG algorithm [9].

Algorithm 1 Compute Steering Direction at Timestep t

Require: $Z_t \in \mathbb{R}^{N \times d_Z}$, $Y \in \mathbb{R}^{N \times 2}$ (one-hot SID predicted labels)

- 1: $Z_t \leftarrow Z_t - \text{MEAN}(Z_t, \text{dim} = 0)$ ▷ Center the data
 - 2: $C \leftarrow Z_t^\top \cdot Y$ ▷ Cross-covariance
 - 3: $A \leftarrow C \cdot C^\top$ ▷ Kernel matrix in Eq. (3)
 - 4: $(\sigma, U) \leftarrow \text{LOBPCG}(A, k = 2)$ ▷ Top-2 eigenpairs
 - 5: $\delta_t \leftarrow \sigma_0 \cdot U_0 + \sigma_1 \cdot U_1$ ▷ Weighted steering direction in Eq. (4)
 - 6: **return** δ_t
-

28 **Note.** There is a typo in Eq. (3) of the main paper, where \mathbf{Z} is mistakenly written as \mathbf{K}_{ZZ} . Therefore,
 29 the correct equation will be equal to the one shown in Algorithm 1 as

$$\begin{aligned} \arg \max_U \text{Tr} \left\{ \mathbf{U}^\top \mathbf{Z} \mathbf{H} \mathbf{K}_{\mathbf{Y}\mathbf{Y}} \mathbf{H} \mathbf{Z}^\top \mathbf{U} \right\}, \\ \text{s.t. } \mathbf{U}^\top \mathbf{U} = \mathbf{I}. \end{aligned} \quad (8)$$

30 Subsequently, the solution for \mathbf{U} is the eigenvectors corresponding to the d -largest eigenvalues of
 31 $\mathbf{A} := \mathbf{Z} \mathbf{H} \mathbf{K}_{\mathbf{Y}\mathbf{Y}} \mathbf{H} \mathbf{Z}^\top$. This typo is fixed in the revised version.

32 A.3 Hyperparameter Search for λ_t

33 In Eq. (6), PolyJuice steers a T2I latent \mathbf{Z}_t using the steering direction δ_t and a magnitude coefficient
 34 $\lambda_t \in [0, \infty)$. In all steering approaches, finding the correct magnitude for adding the steering
 35 direction to the normal flow requires a hyperparameter search [2, 7, 5], and PolyJuice is no exception.
 36 As a result, there is a need for finding an optimal λ_t .

37 To efficiently limit the search space for the set $\{\lambda_t\}_{t=0}^{T-1}$, we consider $\lambda_t = \lambda \cdot \mathbb{1}\{a \leq t \leq b\}$, that
 38 is, we only apply the steering directions δ_t at a constant magnitude λ over some continuous interval
 39 $[a, b] \subseteq [0, T)$. We then use a simple hyperparameter search based on the Optuna framework [1] to
 40 find (λ, a, b) . For any text caption $c \in \mathcal{C}$, we define a *budget*, or the maximum number of attempts
 41 that a T2I model can make to deceive an SID. We find that PolyJuice can find a majority of the
 42 successful attacks within few attempts (e.g. 10), as we show in Fig. 8. Further, for all captions, we
 43 generate all attacks with the same random seed (0) for determinism.

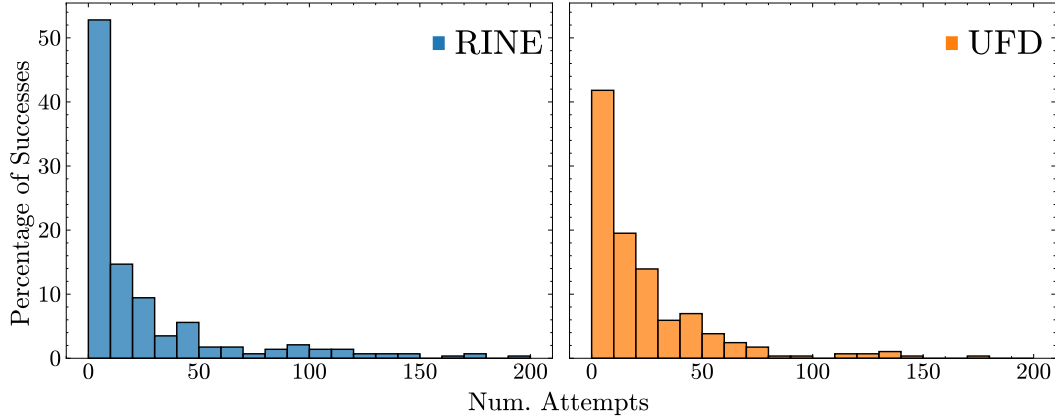


Figure 8: Number of attempts to find successful attacks using FLUX_[dev].

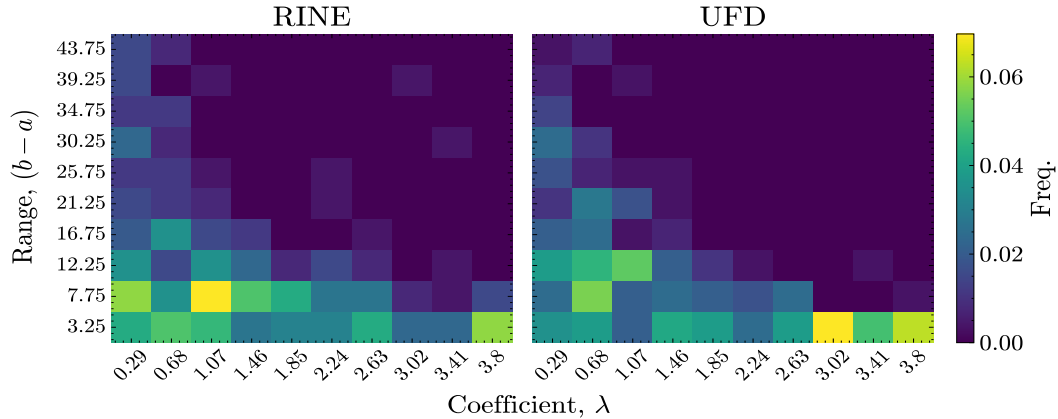


Figure 9: Heatmap showing λ and corresponding range $(b - a)$ for FLUX_[dev].

44 We also analyze the hyperparameters corresponding to successful attacks as heatmaps. From Fig. 9,
 45 we observe that there is a negative correlation between the constant magnitude coefficient λ and the

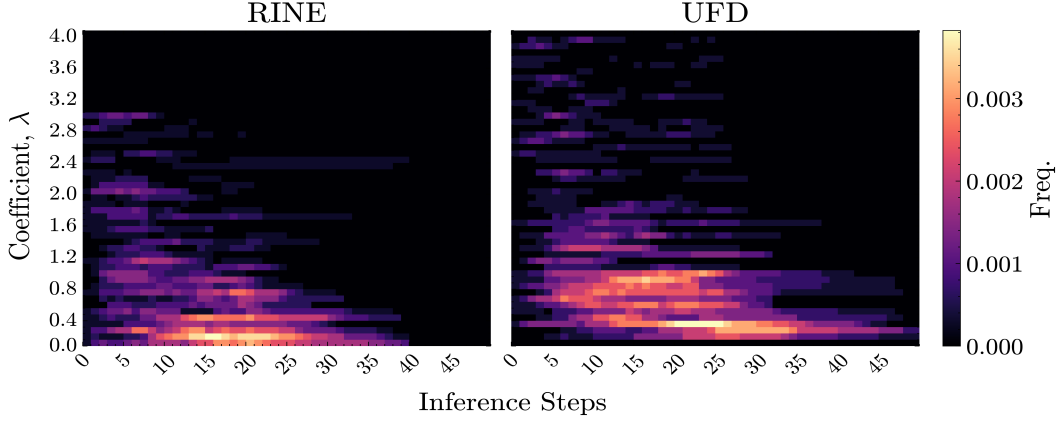


Figure 10: Heatmap showing λ and PolyJuice steering steps for FLUX_[dev].

length of the continuous interval $[a, b]$ over which PolyJuice is applied. For higher values of λ , the steering direction δ_t is applied to only a few steps, while for smaller values, the steering is done on a large number of steps.

Previously, Esser et al. [4] show that intermediate timesteps are more important in the image generation process of flow matching-based T2I models. This finding is supported by Fig. 10, which depicts that PolyJuice-steering overall prefers intermediate steps and smaller coefficients.

A.4 Calibrating the Confidence Threshold of an SID

For calibrating the threshold τ of SID models, we follow the algorithm used by Ojha et al. [8]. As shown in Algorithm 2, the algorithm requires the same number of *real* and *fake* samples and their corresponding predicted confidence scores as the input and return the value of the best threshold τ_{best} that maximizes the accuracy. In practice, for real images, we use a set of images from the training split of COCO, while the fake images are generated by T2I models.

In Tab. 1 we calibrate UFD using 20K real images and 20K generated images mixed from all the T2I models. For each T2I-SID pair in Tab. 2, we calibrate the SIDs (Cal. in Tab. 2) using the following setting. For n successful PolyJuice attacks ($n < 1000$), we use $2n$ real images, n synthetic images generated by the unsteered T2I models, and n PolyJuice-steered images.

Algorithm 2 Find the Best Threshold τ for SIDs (from [8])

Require: Ground truth labels y_{true} , SID predicted scores y_{pred} \triangleright where, $|y_{\text{true}} = 0| = |y_{\text{true}} = 1|$

- 1: $\text{indices} \leftarrow \text{ARGSORT}(y_{\text{true}})$
- 2: $y_{\text{true}}, y_{\text{pred}} \leftarrow y_{\text{true}}[\text{indices}], y_{\text{pred}}[\text{indices}]$ \triangleright Sort y_{true} and y_{pred} according to y_{true}
- 3: $N \leftarrow \text{LEN}(y_{\text{true}})$
- 4: **if** $\max(y_{\text{pred}}[0 : \lfloor N/2 \rfloor]) \leq \min(y_{\text{pred}}[\lfloor N/2 \rfloor : N])$ **then** \triangleright Perfectly separable real and fake
- 5: **return** $\frac{1}{2} (\max(y_{\text{pred}}[0 : \lfloor N/2 \rfloor]) + \min(y_{\text{pred}}[\lfloor N/2 \rfloor : N]))$
- 6: **end if**
- 7: $\text{best_acc}, \tau_{\text{best}} \leftarrow 0, 0$
- 8: **for all** $\tau \in y_{\text{pred}}$ **do** \triangleright Greedily test each y_{pred} as a threshold
- 9: $\text{temp} \leftarrow y_{\text{pred}}$
- 10: **for** $i = 0$ to $N - 1$ **do**
- 11: $\text{temp}[i] \leftarrow \mathbb{1}\{\text{temp}[i] \geq \tau\}$
- 12: **end for**
- 13: $\text{acc} \leftarrow \frac{1}{N} \sum_{i=0}^{N-1} \mathbb{1}\{\text{temp}[i] = y_{\text{true}}[i]\}$
- 14: **if** $\text{acc} \geq \text{best_acc}$ **then**
- 15: $\tau_{\text{best}} \leftarrow \tau$
- 16: $\text{best_acc} \leftarrow \text{acc}$
- 17: **end if**
- 18: **end for**
- 19: **return** τ_{best}

62 A.5 Implementation of Spectral Fingerprint Analysis

63 Given a generated image $\mathbf{x}^{(i)}$, we assume that the high-frequency details in $\mathbf{x}^{(i)}$ are a sum of (i) a
 64 deterministic component arising from the generative model, i.e. the fingerprint F , and (ii) a random
 65 component $\mathbf{w}^{(i)} \sim \mathcal{N}(\mathbf{0}, I)$ [3].

66 For each group of images, we first apply a denoising filter and estimate a noise residual by computing
 67 the difference from the original image. Then we visualize the average residuals in the frequency
 68 domain, as shown in Figs. 6 and 19. The algorithm for computing spectral fingerprints is provided in
 69 Algorithm 3.

Algorithm 3 Computing Spectral Fingerprint of Images

Require: A set of images X

- 1: $X_{\text{denoised}} \leftarrow \text{NL-Mean}(X)$ ▷ Denoise the images
 - 2: $X_{\text{Residual}} \leftarrow X - X_{\text{denoised}}$ ▷ Compute the residuals
 - 3: $F \leftarrow \text{FFT}(X_{\text{Residual}})$ ▷ Perform Fast Fourier Transform
 - 4: $S \leftarrow \|\text{FFT_SHIFT}(F)\|$ ▷ Shift the zero-frequency component to the center of the spectrum
 - 5: $S_{\log} \leftarrow \log(1 + S)$
 - 6: **return** S_{\log}
-

70 A.6 Implementation of Projected Steering Directions

71 Fig. 5 shows the update directions of PolyJuice-steered T2I models in a projected subspace. For a
 72 given timestep t , we first solve the eigenvalue problem associated with Eq. (3), to obtain an orthogonal
 73 projection matrix U_t . The PolyJuice-steering direction at the current timestep, δ_t is computed from a
 74 convex combination over the columns of U_t , as described by Eq. (4).

75 Next, for each latent \mathbf{z}_t , we map the latent to the subspace by computing $U_t^\top \mathbf{z}_t$. We also consider
 76 the latent at the next timestep $t + 1$ and similarly project it as $U_t^\top \mathbf{z}_{t+1}$. Subsequently, the update
 77 vector to the next step can be defined as $\mathbf{u}_t = U_t^\top (\mathbf{z}_{t+1} - \mathbf{z}_t)$.

78 For both unsteered and PolyJuice-steered T2I latents, we plot the unit update direction $\hat{\mathbf{u}}_t$, positioned
 79 at the corresponding mapped points $U_t^\top \mathbf{z}_t$.

80 A.7 Experiments Compute Resources

81 For all experimental steps—including dataset generation, direction computation, and attacks—we
 82 used eight NVIDIA RTX A6000 GPUs, each with 48 GB of memory. The primary computational
 83 bottleneck arises from the memory requirements of the T2I models during image generation; PolyJuice
 84 itself adds negligible overhead. For the highest image resolution considered in this paper, image
 85 generation consumed approximately 75% of GPU memory, equivalent to 36 GB.

86 B Qualitative Analysis of Generated Attacks

87 We provide some additional qualitative examples of successful attacks from PolyJuice-steered T2I
 88 models in Figs. 11 to 16. In general, most of the images look realistic, even though we do not
 89 explicitly enforce any realism constraint. However, we notice that there are some characteristics of
 90 PolyJuice-generated attacks against specific SID models, which we discuss later in § B.1.



Figure 11: Attacks generated by PolyJuice-steered FLUX_[dev] model that were able to deceive RINE.



Figure 12: Attacks generated by PolyJuice-steered SDv3.5 model that were able to deceive UFD.



Figure 13: Attacks generated by PolyJuice-steered SDv3.5 model that were able to deceive RINE.



Figure 14: Attacks generated by PolyJuice-steered FLUX_[sch] model that were able to deceive UFD.



Figure 15: Attacks generated by PolyJuice-steered FLUX_{sch} model that were able to deceive RINE.



Figure 16: Attacks generated by PolyJuice-steered FLUX_{dev} model that were able to deceive UFD.

91 B.1 Common Patterns in Successful Attacks



Figure 17: (Top) Unsteered T2I-generated images that RINE correctly detects as fake. (Bottom) PolyJuice-steered images that successfully deceive RINE as real.

92 **Common Patterns in Successful Attacks on RINE.** In successful attacks against RINE, we
 93 observe that the generated images often exhibit low brightness. In Fig. 17 we show some images
 94 generated by unsteered (top) and PolyJuice-steered (bottom) T2I models, using captions from the
 95 COCO validation set. Further, the unsteered and steered images share the same random initial latent
 96 (i.e. they are all generated with the random seed 0). Although RINE successfully detects the unsteered
 97 images as fake, it is deceived by the relatively *darker* PolyJuice-steered images. This suggests a
 98 vulnerability of RINE to synthetic images that appear underexposed or have low brightness levels.



Figure 18: (Top) Unsteered T2I-generated images that UFD correctly detects as fake. (Bottom) PolyJuice-steered images that successfully deceive UFD as real.

99 **Common Patterns in Successful Attacks on UFD.** In successful attacks against UFD, we observe
 100 that the generated images often exhibit warm colors. Fig. 18 shows some examples unsteered images
 101 (top) that UFD detects as fake, and the corresponding PolyJuice-steered images (bottom) that fools the
 102 detector. Even on obviously fake images, such as the third column in Fig. 18 (cartoon of wild animals,

generated by SDv3.5), PolyJuice produces an image with a warmer tone that UFD cannot detect as fake. This suggests a vulnerability of UFD to synthetic images with warm color temperatures.

C Additional Results

C.1 Spectral Fingerprint Analysis

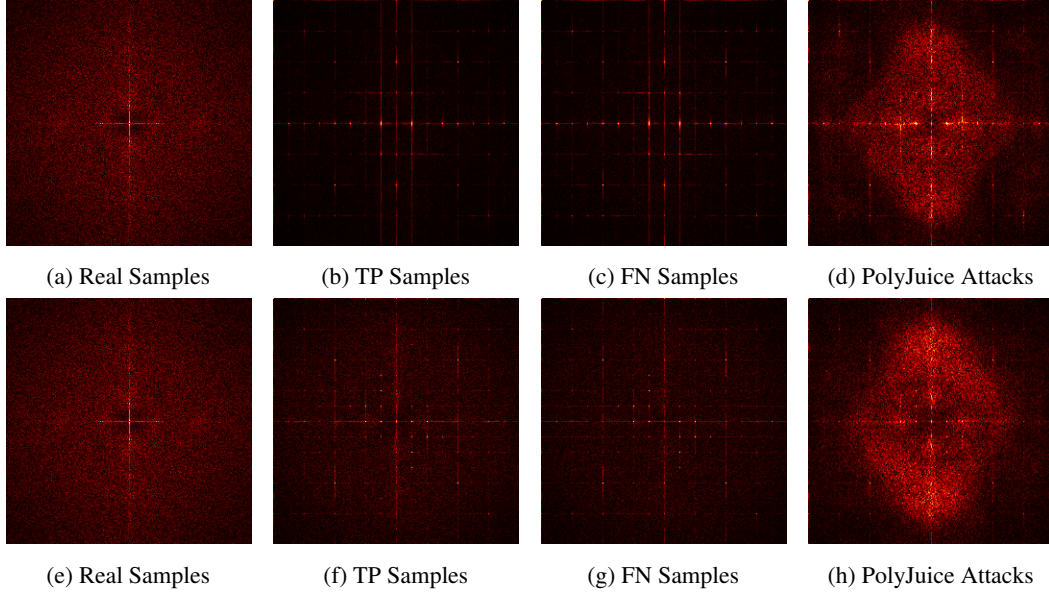


Figure 19: More results on **Average Frequency Spectra** of COCO images and generated counterparts where 1st and 2nd rows correspond to samples from FLUX_[dev] and FLUX_[sch], respectively.

Fig. 6 in the main paper illustrates the spectral fingerprints of real, unsteered, and PolyJuice-steered samples generated by SDv3.5. Here, we extend the analysis to the other two T2I models, FLUX_[dev] and FLUX_[sch]. The first and second rows of Fig. 19 show the spectral fingerprints of samples generated by FLUX_[dev] and FLUX_[sch], respectively. We observe that PolyJuice-steered attacks effectively obscure the characteristic frequency patterns of their underlying T2I models, producing spectra that more closely resemble those of real images compared to unsteered attacks.

C.2 Realness Shift in T2I Latent Space

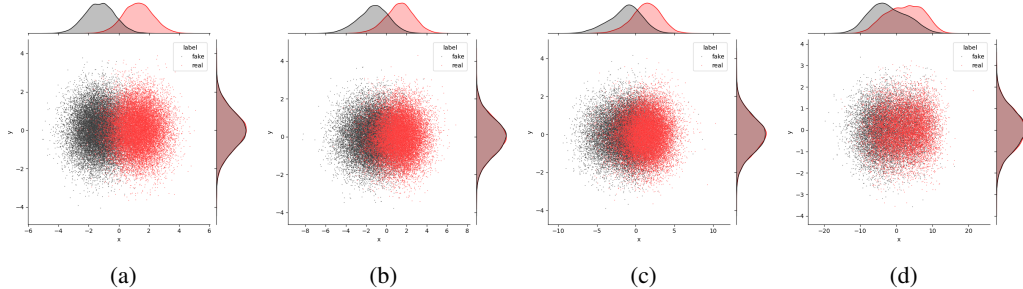


Figure 20: More visualizations of the distribution shift in the T2I model’s latent space.

In Fig. 1b of the main paper, we showed the realness shift in the latent space of T2I model for one timestep. In Fig. 20, we show the shift for four timesteps. We observe that there exists a clear shift between the predicted real and fake samples. However, the degree of linear separability of these distributions is not constant across different timesteps.

118 C.3 More Visualizations on the Effect of PolyJuice in Image Generation Process

119 In Fig. 21, we show additional image-space visualizations of the effect of PolyJuice, by estimating
 120 the clean image at various timesteps. As noted in § A.3, we only apply PolyJuice over a continuous
 121 interval of inference steps $[a, b] \subseteq [0, T)$, as can be seen from the figure. In both Figs. 21a and 21b,
 122 the original unsteered T2I image generation process (bottom rows) produces an image that is detected
 123 by the SID as fake. PolyJuice steers the T2I to generate images that deceive the detectors (top rows).



(a) Intermediate steps for "A replica of a bear and her cub in a glass case in an exhibit."



(b) Intermediate steps for "Two men sitting next to each other on a wooden bench."

Figure 21: Visualizing the effect of PolyJuice on the image generation process. (Top) Image that successfully deceives RINE. (bottom) Image that successfully deceives UFD.

124 References

- 125 [1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-
 126 generation hyperparameter optimization framework. In *The 25th ACM SIGKDD International Conference*
 127 *on Knowledge Discovery & Data Mining*, pages 2623–2631, 2019.
- 128 [2] Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping,
 129 and Tom Goldstein. Universal guidance for diffusion models. In *Proceedings of the IEEE/CVF Conference*
 130 *on Computer Vision and Pattern Recognition*, pages 843–852, 2023.
- 131 [3] Riccardo Corvi, Davide Cozzolino, Giada Zingarini, Giovanni Poggi, Koki Nagano, and Luisa Verdoliva. On
 132 the detection of synthetic images generated by diffusion models. In *ICASSP 2023-2023 IEEE International*
 133 *Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.
- 134 [4] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi,
 135 Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution
 136 image synthesis. In *Forty-first international conference on machine learning*, 2024.

- 137 [5] Felix Friedrich, Manuel Brack, Lukas Struppek, Dominik Hintersdorf, Patrick Schramowski, Sasha Luccioni,
138 and Kristian Kersting. Auditing and instructing text-to-image generation models on fairness. *AI and Ethics*,
139 2024.
- 140 [6] Huggingface. Huggingface models. <https://huggingface.co/models>, 2025.
- 141 [7] Nithin Gopalakrishnan Nair, Anoop Cherian, Suhas Lohit, Ye Wang, Toshiaki Koike-Akino, Vishal M
142 Patel, and Tim K Marks. Steered diffusion: A generalized framework for plug-and-play conditional
143 image synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages
144 20850–20860, 2023.
- 145 [8] Utkarsh Ojha, Yuheng Li, and Yong Jae Lee. Towards universal fake image detectors that generalize
146 across generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern
147 Recognition*, pages 24480–24489, 2023.
- 148 [9] Andreas Stathopoulos and Kesheng Wu. A block orthogonalization procedure with constant synchronization
149 requirements. *SIAM Journal on Scientific Computing*, 23(6):2165–2182, 2002.