Distilling Diffusion Models into Conditional GANs

-Supplementary Material-

We elaborate on the training and evaluation details for Diffusion2GAN in Supplement A and Supplement B, respectively. Following this, we provide an additional explanation of our proposed E-LatentLPIPS in Supplement C. In Supplement D, we offer a quantitative comparison with GigaGAN. Then, we discuss the noise and ODE solution pair dataset in Supplement E. We also explain the limitations of our work in Supplement F and its societal impact in Supplement G. Finally, we present additional visuals of Diffusion2GAN and also qualitatively demonstrate that Diffusion2GAN is capable of synthesizing well-aligned and diverse images using a single prompt in Supplement H.

A Training Details

A.1 Text-to-Image Synthesis

Parameterization. We distill Stable Diffusion [24, 26] into Diffusion2GAN using the PyTorch framework [23]. Throughout our experiments, we utilize the U-Net architecture employed in Stable Diffusion, initializing the U-Net weights with the pre-trained weights of Stable Diffusion. As the Stable Diffusion was originally designed to predict a denoising noise $\epsilon(\mathbf{x}_t, \mathbf{c}, t)$ given a noisy sample $\mathbf{x}_t$, we modify the noise prediction parameterization to the data prediction parameterization using the following equation, though with a slight abuse of notation:

$$G(\mathbf{x}_t, \mathbf{c}, t) = \frac{\mathbf{x}_t - \sigma_t \epsilon(\mathbf{x}_t, \mathbf{c}, t)}{\alpha_t}, \tag{1}$$

where σ_t and α_t are manually defined diffusion schedule. Since Diffusion2GAN performs noise-to-latent mapping, translating pure Gaussian noise $\mathbf{z} = \mathbf{x}_T$ to a target latent $\mathbf{x} = \mathbf{x}_0$, the data prediction parameterization mentioned above can be re-written as follows:

$$G(\mathbf{z}, \mathbf{c}) = \frac{\mathbf{z} - \sigma_T \epsilon(\mathbf{z}, \mathbf{c}, T)}{\alpha_T}. \tag{2}$$

While it is essential to employ the data prediction parameterization for the generator, as the Diffusion2GAN's objective is to predict a target latent rather than a denoising noise, we empirically discover that adopting the noise prediction parameterization for the discriminator does not lead to instability issues.

Two-stage Diffusion2GAN training. We observe enhanced stability and increased diversity in image generation when employing a two-stage training approach for Diffusion2GAN. In the initial stage, Diffusion2GAN is exclusively trained using the ODE distillation loss. Subsequently, we fine-tune the ODE-distilled one-step generator by incorporating the ODE distillation, conditional GAN, and single-sample R1 losses. Experimentally, we discover that training

Diffusion2GAN with a different conditional GAN loss weight typically results in stable convergence. Increasing the weight of the conditional GAN loss component enhances the fidelity of generated images but decreases image diversity. We speculate this occurs because the conditional GAN loss prioritizes realistic image synthesis over accurately learning the original ODE trajectory of the teacher model. Detailed hyperparameters are provided in Table A1.

A.2 Conditional Image Synthesis on CIFAR10-32px

Consistency Distillation training. In Table 6, we present FID [6] of Consistency Distillation (CD) [31] on CIFAR10 [15]. We implement a conditional version of CD and train it for 150k iterations with a batch size of 512, resulting in 307.2M $= 4 \times 150\text{k} \times 512$ number of function evaluations (NFE). Note that the official unconditional CD was trained for 800k iterations to achieve an FID of 3.55, while our conditional CD implementation achieves a nearly identical FID of 3.67 with only 400k training iterations, entailing 819.2M NFE.
ODE distillation training. We distill a pre-trained EDM [10] on CIFAR10 into a single-step generator only using ODE distillation loss. To create the noise and ODE solution pairs, we simulate the pre-trained EDM 18 times using a Heun sampler [10]. When training the ODE distilled generator, we adhere to using the original parameterization of EDM, as the EDM is originally designed to perform data prediction. The hyperparameter details are presented in Table A1.

B Evaluation Details

We evaluate our model on two widely used datasets, COCO2014 and COCO2017. We include the evaluation on COCO2017, as progressive distillation [28] and DPM solver [19] only report results on this dataset. We use FID [6] and CLIP-score [5] to assess image realism and text-to-image alignment. Following GigaGAN's protocol [9], we resize the generated images 512px to 256px, reprocess them to 299px, and then feed them into the InceptionV3 network for FID and Precision & Recall calculations [16, 27]. FID [6, 22] is computed on 40,504 real images from the COCO2014 validation dataset and 30,000 fake images generated using 30,000 randomly sampled COCO2014 validation prompts, while Precision & Recall are calculated on 10,000 images due to their heavy computation. For COCO2017 dataset, we use 5,000 image-text pairs for FID and CLIP-score calculations. Precision & Recall metrics on COCO2017 are omitted, as we heuristically find that 5,000 real samples are insufficient to yield valid measurements of image fidelity and diversity. Instead, we introduce a new diversity score, calculated using DreamSim [3], to quantify the range of variation in the generated images. Note that the resizing processes in the evaluation pipeline are performed using an antialiasing bicubic resizer, as recommended by Parmar *et al.* [22].

Table A1: Hyperparameters for Diffusion2GAN training. We denote pixel blitting and geometric transformations as bg [11] and bg with color transformations as bgc [11, 34]. For additional technical details, please refer to the original papers: LPIPS [33], Cutout [2], Non-saturation loss [4], Adam optimizer [14], RAdam optimizer [17], EDM [10], SD 1.5 [25], SDXL-Base-1.0 [24], Noise augment before D [9,32], Heun sampler [10], and DDIM sampler [30]. E-LatentLPIPS* refers to the ensemble of E-LatentLPIPS with vanilla LatentLPIPS.

Model hyperparameters	CIFAR10 32px	SD-CFG-3 64px	SD-CFG-8 64px	SDXL-CFG-7 128px
$\mathbf{z}$ dimension	3×32×32	4×64×64	4×64×64	4×128×128
$\mathbf{x}$ dimension	3×32×32	4×64×64	4×64×64	4×128×128
$\mathcal{L}^{\text{ODE}}_{\text{distill}}$ loss type	LPIPS	E-LatentLPIPS*	E-LatentLPIPS	E-LatentLPIPS
E-LatentLPIPS augmentation	-	bg + cutout	bgc + cutout	bgc + cutout
$\mathcal{L}^{\text{ODE}}_{\text{distill}}$ loss strength	1.0	1.0	1.0	1.0
$\mathcal{L}_{\text{GAN}}$ loss type	-	Non-saturation	Non-saturation	Non-saturation
$\mathcal{L}_{\text{GAN}}$ loss strength	-	0.25	0.25	1.0
Single-sample R1 strength	-	0.01	0.01	-
Single-sample R1 interval	-	16	16	-
Mix-and-match augmentation	False	True	True	True
Optimizer	RAdam	Adam	Adam	Adam
Batch size	512	256	2048→1024	1024→512
Accumulation	1	1	1	1
G learning rate	4e-4	1e-4→1e-5	1e-4→1e-5	1e-4→1e-5
G β_1 for Adam	0.9	0.9	0.9	0.9
G β_2 for Adam	0.999	0.999	0.999	0.999
D learning rate	-	1e-4→1e-5	1e-4→1e-5	1e-4→1e-5
D β_1 for Adam	0.9	0.9	0.9	0.0
D β_2 for Adam	0.999	0.999	0.999	0.99
Weight decay strength	0.0	1e-2	1e-2	1e-2
Weight decay strength on attention	0.0	1e-5	1e-5	1e-5
Dropout rate	0.1	0.0	0.0	0.0
# D updates per G update	-	1	1	1
G ema start	20k	4k	4k	4k
G ema beta	0.9999	0.9999	0.9999	0.9999
Precision	bfloat16	bfloat16	bfloat16	bfloat16
G backbone	EDM	SD 1.5	SD 1.5	SDXL-Base-1.0
D backbone	-	SD 1.5	SD 1.5	SDXL-Base-1.0
Multi-scale training	-	True	True	True
Noise augment before D	False	False	False	True
Training specifications	**CIFAR10 32px**	**SD-CFG-3 64px**	**SD-CFG-8 64px**	**SDXL-CFG-7 128px**
Diffusion generator	EDM	SD 1.5	SD 1.5	SDXL-Base-1.0
Numerical solver	Heun	DDIM	DDIM	DDIM
Denoising steps	18	50	50	50
# ODE pairs	1.0M	3.0M	12.0M	8.0M
NFE for dataset generation	35.0M	150.0M	600.0M	400.0M
Training specifications	**CIFAR10 32px**	**SD-CFG-3 64px**	**SD-CFG-8 64px**	**SDXL-CFG-7 128px**
G Model size	61.5M	859.5M	859.5M	2567.5M
D Model size	-	859.6M	859.6M	2567.7M
First stage iterations	150k	150k	50k	20k
Second stage iterations	-	10k	10k	30k
NFE for training	76.8M	51.2M	153.6M	97.3M
GPU type	A100	A100-80GB	A100-80GB	A100-80GB
# GPUs for training	8	16	64	128
GPU days	6.0	43.6	119.2	356.8
FID	3.16	9.29	13.39	25.49

C E-LatentLPIPS

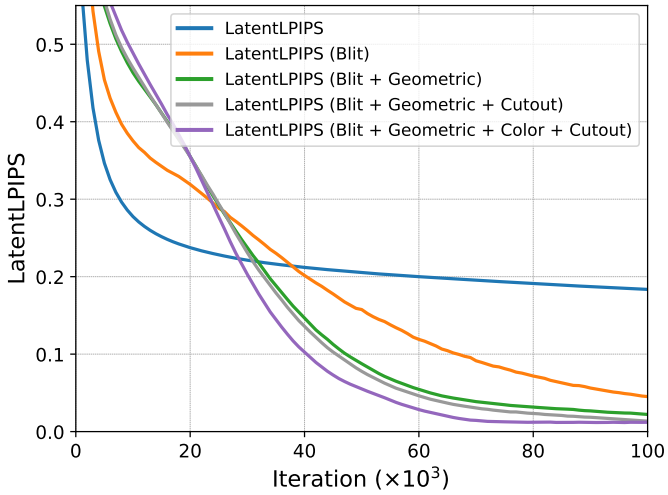

Fig. A1: Single sample overfitting experiment. LatentLPIPS fails to achieve overfitting, even in a single-sample overfitting experiment. However, by applying diverse differentiable augmentations to the inputs of LatentLPIPS, we can successfully reconstruct the target latent. Blit indicates Horizontal flip + 90-degree rotation + integer translation. Geometric indicates isotropic scaling + arbitrary rotation + anisotropic scaling + fractional translation. Color indicates random brightness + random saturation + random contrast. For technical details on the differentiable augmentations, we recommend referring to the papers [11, 34].

C.1 Toy Experiment

We conducted a single image reconstruction experiment to study how LatentLPIPS behaves. Beginning with a 512-pixel target image, denoted as $\mathbf{I}_{\text{target}} \in \mathbb{R}^{3\times512\times512}$, we utilized the VAE encoder of Stable Diffusion to obtain its latent vector, resulting in $\mathbf{x}_{\text{target}} = \text{Encode}^{1/8\times}(\mathbf{I}_{\text{target}}) \in \mathbb{R}^{4\times64\times64}$. Subsequently, we randomly initialized a trainable latent vector $\mathbf{x}_{\text{source}}$ with the same dimensions as $\mathbf{x}_{\text{target}}$. The objective of this experiment is to determine whether LatentLPIPS can achieve a latent vector $\mathbf{x}_{\text{source}}$ that precisely reconstructs $\mathbf{x}_{\text{target}}$ using the following LatentLPIPS objective and a gradient-based optimizer:

$$d_{\text{LatentLPIPS}}(\mathbf{x}_{\text{target}}, \mathbf{x}_{\text{source}}) = \ell(F(\mathbf{x}_{\text{target}}), F(\mathbf{x}_{\text{source}})), \tag{3}$$

where F is a VGG network trained in the latent space of Stable Diffusion, and $\ell(\cdot,\cdot)$ is a distance metric. While a well-designed single sample overfitting is typically considered feasible, our analysis suggests that LatentLPIPS struggles with

optimization, resulting in a high loss value, as shown in Figure A1. Moreover, we observed systematic wavy and patchy artifacts in the reconstructed image decoded by the source latent. We hypothesize that this limitation arises from a suboptimal loss landscape created by the latent version of the VGG network.

Inspired by E-LPIPS [13] and the observation that only a portion of the region has been successfully reconstructed using the source latent, we apply geometric augmentations and cutout [2] to both the source and target latents. To ensure differentiability for backpropagation, we employ off-the-shelf differentiable augmentations [11,34]. Upon introducing these augmentations, we notice improved convergence of LatentLPIPS, suggesting that the poor optimization can be alleviated by applying an appropriate combination of differentiable augmentations. Through toy experiments, we have confirmed that LatentLPIPS converges faster and better as we introduce more augmentations, including augmentations related to color (random brightness, saturation, and contrast), as shown in Figure A1.

In text-to-image experiments, we found that the combination of generic geometric transformations and cutout achieves the best FID on the SD-CFG-3 dataset, while additionally using the color-related augmentations proves beneficial for the SD-CFG-8 and SDXL-CFG-7 datasets. Furthermore, we discovered that on SD-CFG-3, Diffusion2GAN achieves better FID when E-LatentLPIPS is combined with vanilla LatentLPIPS.

C.2 Perceptual Score of LatentLPIPS vs. LPIPS

In Section 3.1, we described learning LatentLPIPS, following the procedure from LPIPS [33]. This involves training an ImageNet [1] classifier and then tuning it to perceptual scores.

In Table A2, we present ImageNet classification accuracies. The LPIPS network uses VGG16 [29] as a backbone, which achieves 71.59% accuracy. We note that a batch-norm version of the backbone achieves 73.36%. The ImageNet classification score on latent codes drops to 64.25%, while the batch-norm variant recovers some performance on 68.26%. We found the batch-norm variant trains more stably. We followed the default PyTorch training code and parameters https://github.com/pytorch/examples/blob/main/imagenet/main.py, but discovered that we had to reduce the initial learning rate for the non-batch-norm variant. We selected the batch-norm version to form the basis of LatentLPIPS. While the ImageNet classification scores are lower, they are competitive in terms of perceptual quality measurement. More importantly, as noted in the original LPIPS work, ImageNet classification scores do not necessarily correlate with perceptual quality – ImageNet classification is merely a pretext task to yield a representation with high perceptual quality.

In Table A3, we show the perceptual scores on the Berkeley-Adobe Perceptual Patch Similarity (BAPPS) dataset [33]. The dataset provides different types of perturbations, "traditional" hand-crafted perturbations, ones from CNN-generated algorithms, and outputs from real algorithms for image reconstruction tasks (colorization, video interpolation, superresolution, and video deblurring).

We followed the protocol from LPIPS [33], learning a linear calibration on 5 different intermediate layers. Across the different sets, LatentLPIPS achieves similar, sometimes higher scores, as vanilla LPIPS. This indicates that while some details that are advantageous for classification are lost during compression, the perceptually important details are preserved. This result aligns with the goal of designing the latent space [25] in the first place. In conclusion, our LatentLPIPS is able to capture a representation that aligns with human perception, at similar performance to vanilla LPIPS, while enabling faster computation. Please note that extra training for LatentLPIPS was performed to distill SDXL-Base-1.0 into Diffusion2GAN because Stable Diffusion 1.5 and SDXL-Base-1.0 do not share the same latent space.

Table A2: ImageNet classification scores. The backbone networks in * are used for LPIPS [33] & LatentLPIPS calculations. ImageNet accuracy on the Latent code is lower than on pixels, as information is lost during compression. However, ImageNet classification is merely a proxy task for achieving a strong representation to align with human perception. The perceptual scores in Table A3 are competitive, indicating perceptual information is retained.

Perceptual metric	VGG16	VGG-bn
Pixels	**71.59***	**73.36**
Latent	64.25	68.26*

Table A3: Perceptual scores. LatentLPIPS achieves similar and sometimes higher perceptual scores than vanilla LPIPS [33] on the BAPPS dataset.

Perceptual metric	Traditional	CNN	Real
LPIPS [33]	73.36	**82.20**	**63.23**
LatentLPIPS	**74.29**	81.99	63.21

D Quantitative Comparison with GigaGAN

We compare Diffusion2GAN with GigaGAN [9] using additional metrics, including Clip-score [5] and Precision & Recall [16]. We utilize the officially provided GigaGAN samples [8] to compute these metrics. As shown in Table A4, Diffusion2GAN achieves a higher recall than GigaGAN, suggesting that Diffusion2GAN suffers less from diversity collapse than GigaGAN. Despite slightly worse FID and Clip-score, Diffusion2GAN achieves almost comparable performance while using only about 1% of the compute resources.

Table A4: Comparison to text-to-image GigaGAN generator on COCO2014. While our Diffusion2GAN model shows a slightly higher FID [6] compared to GigaGAN [9], it exhibits a higher recall value [16], indicating that Diffusion2GAN can generate more diverse images than GigaGAN.

model	FID-30k (↓)	CLIP-30k (↑)	Precision (↑)	Recall (↑)	A100 days
GigaGAN [9]	**9.09**	**0.32**	**0.74**	0.60	4783.0
Diffusion2GAN	9.29	0.31	0.74	**0.64**	**43.6**

E Discussion on Noise and ODE Solution Pair Dataset

In this paper, we create noise-image (latent) pairs using a pre-trained diffusion model and a deterministic sampler. This prompts fundamental questions: should these pairs strictly adhere to a one-to-one correspondence, and can they be randomly re-paired while still maintaining this correspondence? To explore these questions, we generate noise-image pairs using a stochastic sampler. Specifically, we utilize a pre-trained EDM [10] and generate 50k noise-image pairs using an EDM's stochastic sampler. Subsequently, we train a one-step model using ODE distillation loss with LPIPS, as explained in Section 3.1. However, the one-step model with stochastic pairs cannot minimize the ODE distillation loss, resulting in an FID over 200. This phenomenon also occurs when we randomly re-wire 50k deterministic noise-image pairs without replacement. This result contradicts our earlier findings, where a model trained using ODE distillation loss achieved an FID score of 8.51 using 50k diffusion-simulated deterministic noise-image pairs, as presented in Table 6. These results suggest that for effective ODE distillation, noise-image pairs should be deterministically generated and inherit a specific relationship formed by simulating the ODE of a pre-trained diffusion model.

F Limitations

Although our method achieves faster inference while maintaining image quality, it does have several limitations. First, our current approach simulates a fixed classifier-free guidance scale, a common technique for adjusting text adherence, but does not support varying CFG values at inference time. Exploring methods like guided distillation [21] could be a promising direction. Second, as our method distills a teacher model, the performance limit of our model is bound by the quality of the original teacher's output. To enhance the quality of generated noise-image pairs, employing advanced diffusion models like EDM2 [12], which is better compatible with deterministic sampling, could be advantageous. Additionally, leveraging real text and image pairs is a potential avenue to learn a student model that outperforms the original teacher model. Third, Diffusion2GAN only supports one-step image synthesis as it was trained to translate given noise into an RGB image directly. However, extending Diffusion2GAN to multi-step

generation could result in future performance improvement. Last, while Diffusion2GAN alleviates the diversity drop by introducing ODE distillation loss and a conditional GAN framework, we have found that the diversity drop still occurs as we scale up the student and teacher models. We leave further investigation of this problem for future work.

G Societal Impact

Our work aims to develop a one-step image synthesis framework, which could significantly improve the accessibility and affordability of generative visual models. By reducing the multi-step synthesis process into a single step, our technology promises to democratize the creation of visual content, enabling a broader range of users to harness the power of generative models for creative expression and innovation. Additionally, by reducing the need for extensive computation during both training and inference stages, our framework also helps decrease electricity usage and CO2 emissions. However, as this technology becomes more accessible, it is crucial to address concerns about potential misuse, especially in areas like sexual harassment and synthetic media manipulation.

Generative visual models have the potential to facilitate the creation of highly convincing deep fake videos and enable sophisticated impersonation techniques, presenting significant challenges for the trustworthiness of online information. Moreover, they can be utilized to generate content that may incite instances of sexual harassment. While our technology boasts compelling advantages regarding efficiency, it is imperative to acknowledge and tackle the potential societal repercussions and ethical dilemmas linked with the widespread integration of generative visual models.

H More Visual Results

We provide additional visuals from Diffusion2GAN in Figures A2 and A3. Next, we present a visual comparison with Stable Diffusion 1.5 teacher and selected distillation models, including Diffusion2GAN, in Figure A4. We also present additional visual comparison between Stable Diffusion 1.5 [26], GigaGAN [9], InstaFlow-0.9B [18], and our Diffusion2GAN using COCO2014 prompts in Figures A5 and A6. Furthermore, we demonstrate that SDXL-Diffusion2GAN can generate diverse images from a single prompt while maintaining better text-to-image alignment compared to SDXL-Turbo and SDXL-Lightning in Figures A7 and A8.

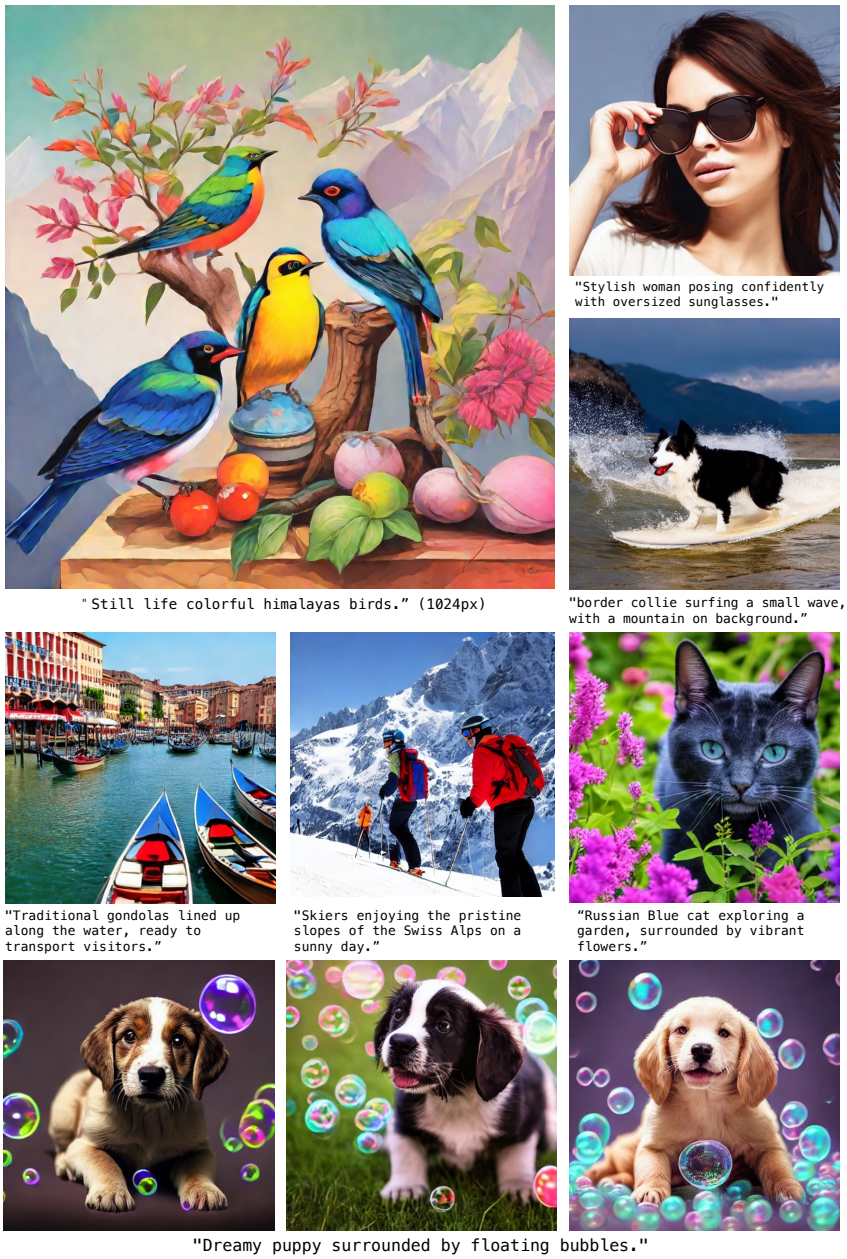

Fig. A2: High-quality generated images using our one-step Diffusion2GAN framework. Our model can synthesize a 512px/1024px image at an interactive speed of 0.09/0.16 seconds on an A100 GPU, while the teacher model, Stable Diffusion 1.5 [26]/SDXL [24], produces an image in 2.59/5.60 seconds using 50 steps of the DDIM [30].

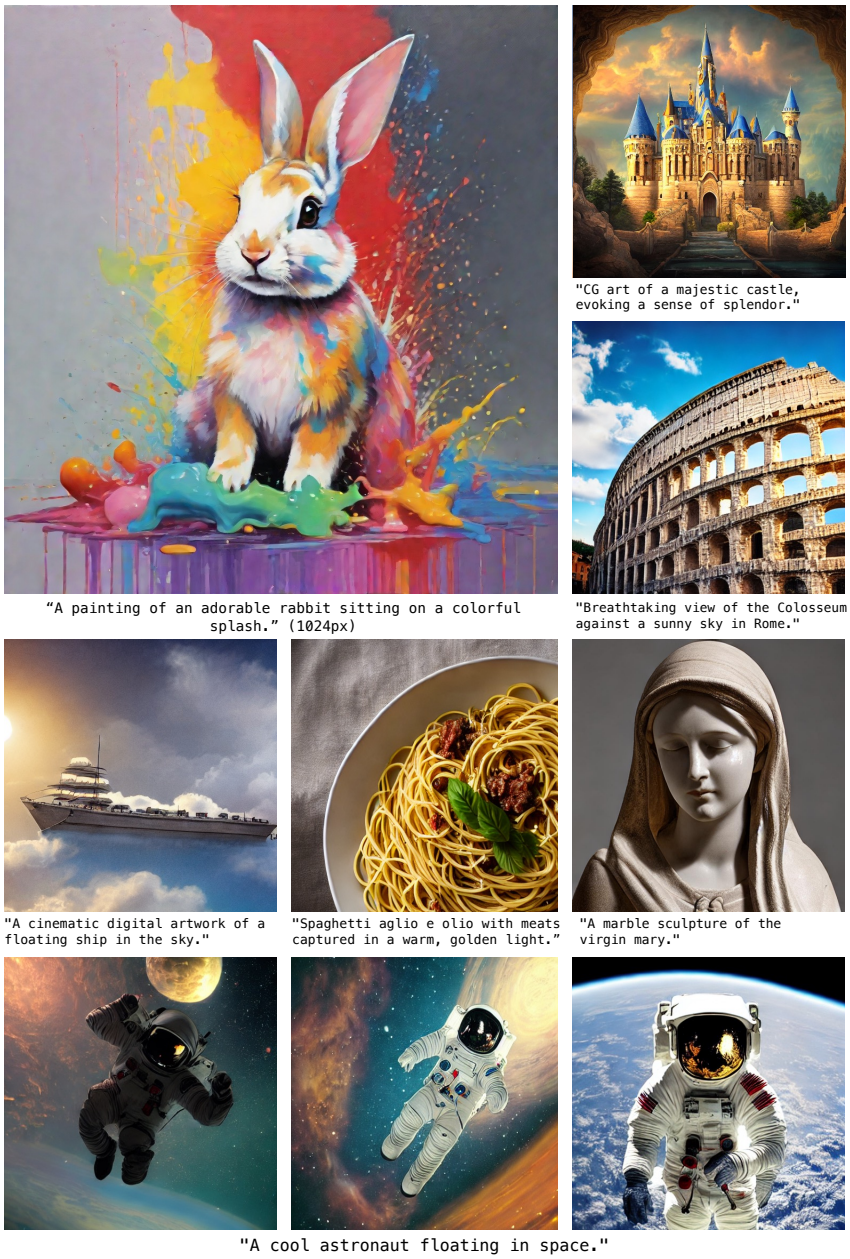

Fig. A3: High-quality generated images using our one-step Diffusion2GAN framework. Our model can synthesize a 512px/1024px image at an interactive speed of 0.09/0.16 seconds on an A100 GPU, while the teacher model, Stable Diffusion 1.5 [26]/SDXL [24], produces an image in 2.59/5.60 seconds using 50 steps of the DDIM [30].

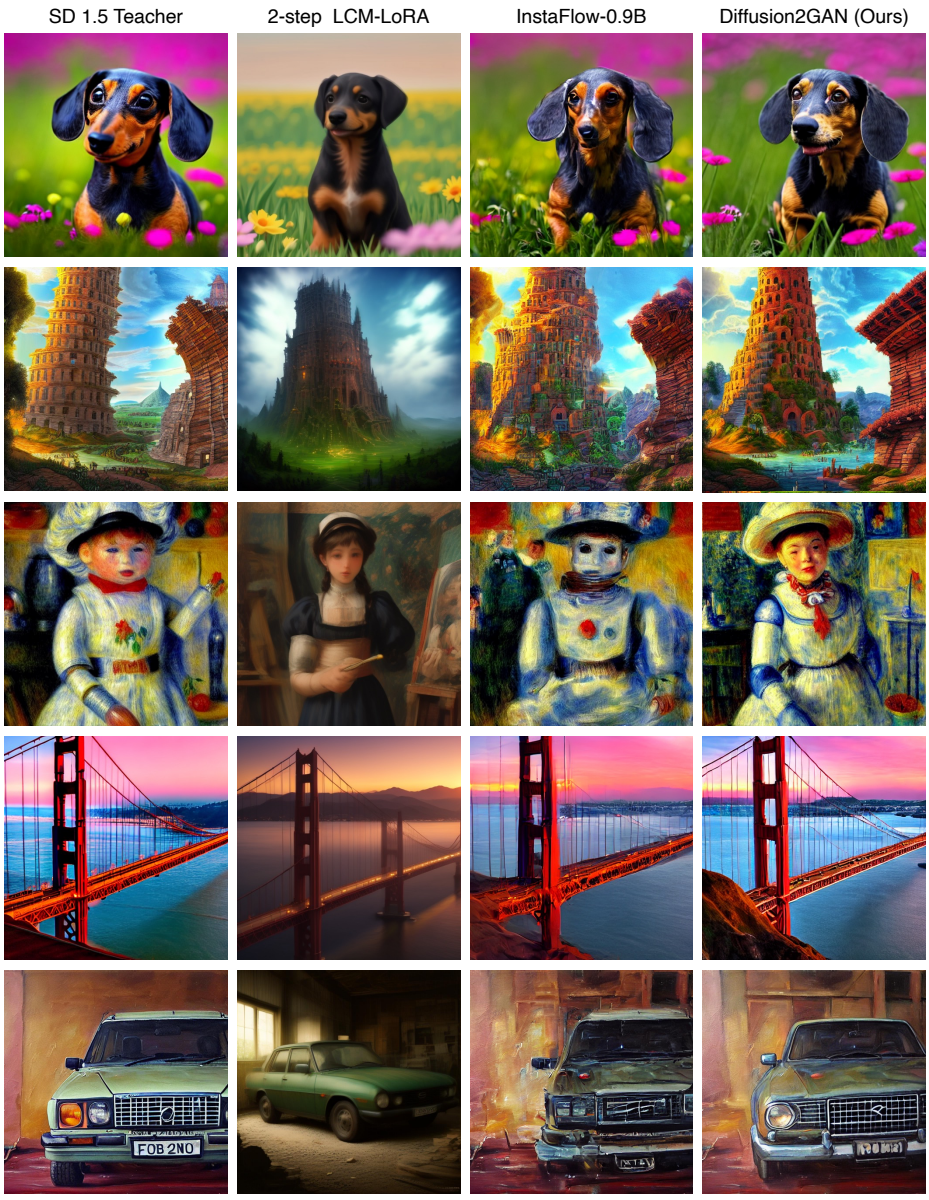

Fig. A4: Visual comparison to Stable Diffusion 1.5 teacher [26] with a classifier-free guidance scale [7] of 8 and selected distillation student models, including InstaFlow-0.9B [18], LCM-LoRA [20], and our Diffusion2GAN. The same noise input was used to generate images in the same row. Our method Diffusion2GAN achieves higher realism than the 2-step LCM-LoRA and InstaFlow-0.9B.

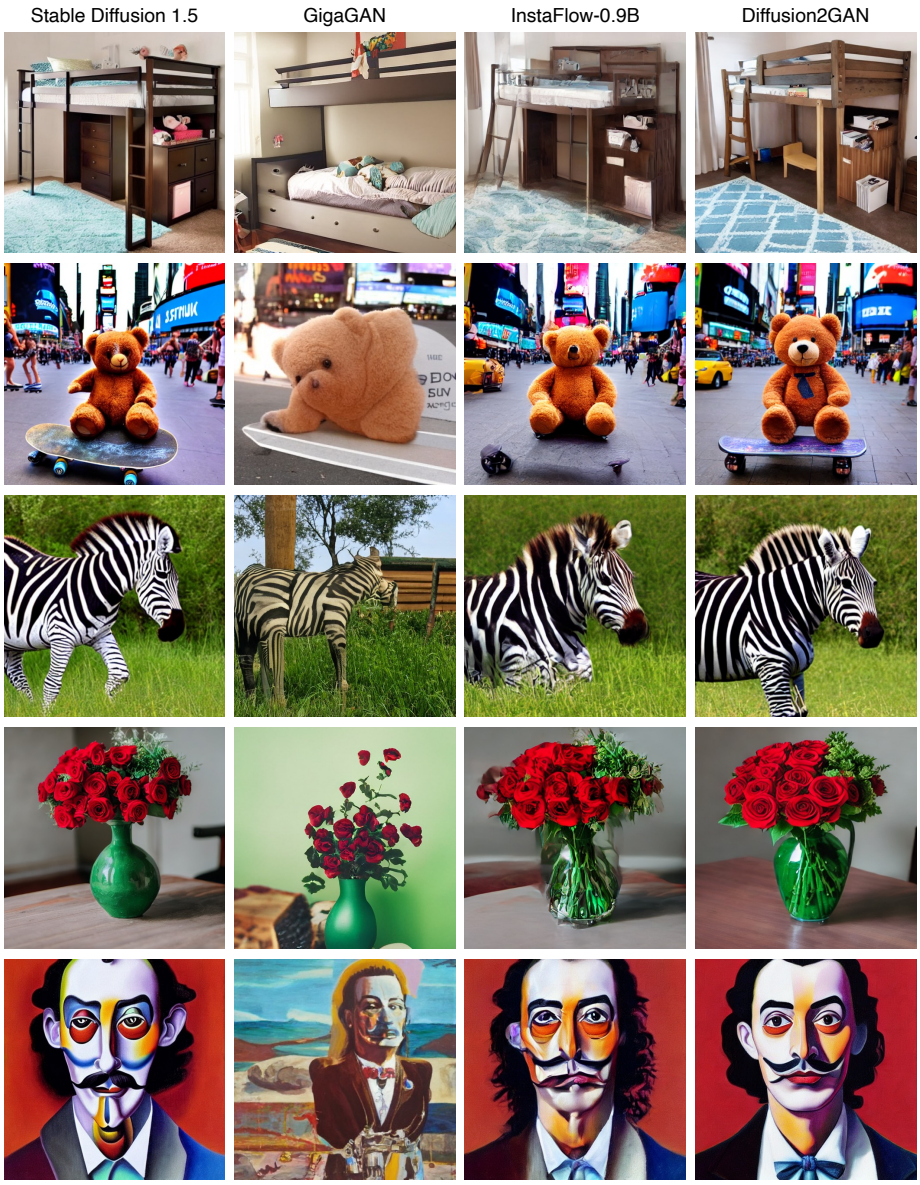

Fig. A5: Visual comparison to Stable Diffusion 1.5 [26] with a guidance scale of 8 [7] and selected one-step generators, GigaGAN [9], InstaFlow-0.9B [18], and Diffusion2GAN trained on SD-CFG-8. We observe that Diffusion2GAN produces more realistic images compared to GigaGAN and InstaFlow-0.9B, while maintaining comparable visual quality with Stable Diffusion 1.5.

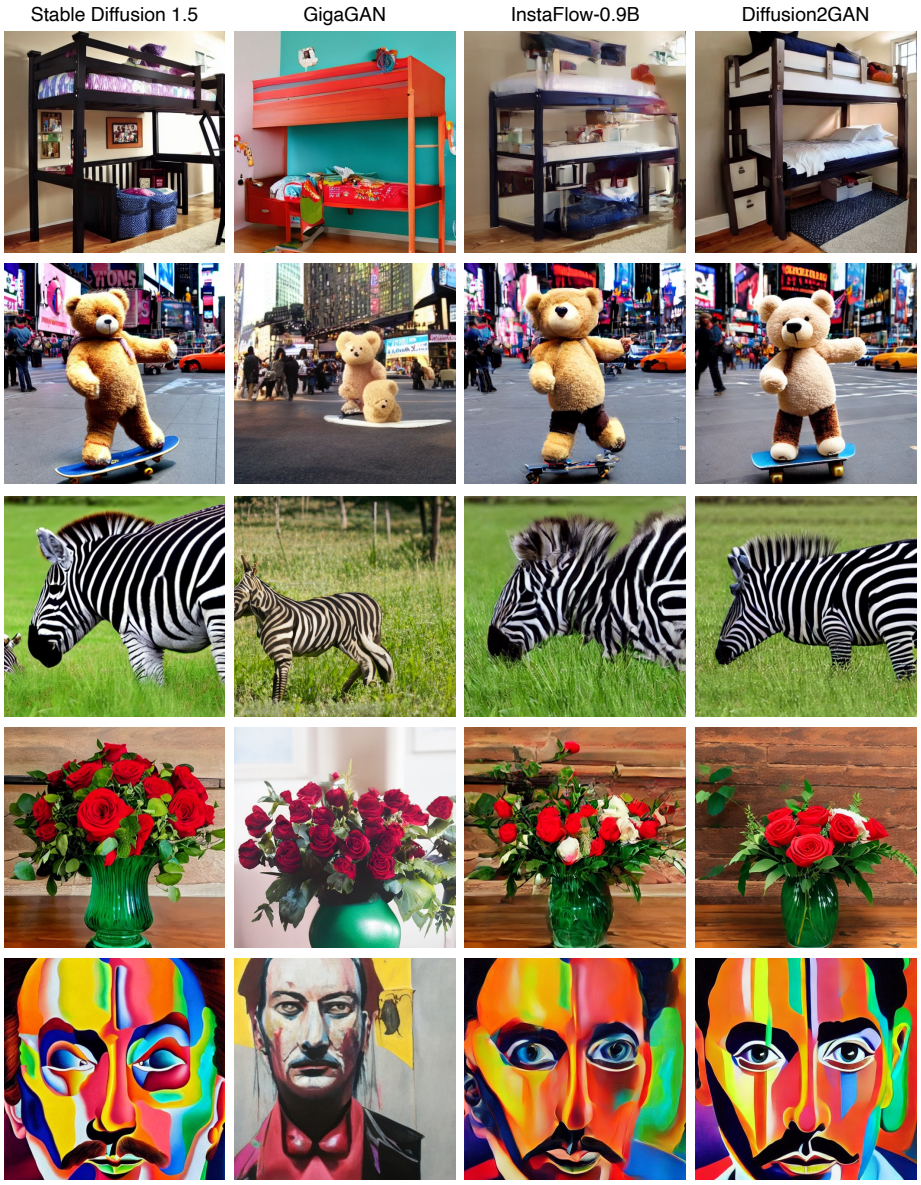

Fig. A6: Visual comparison to Stable Diffusion 1.5 [26] with a guidance scale of 8 [7] and selected one-step generators, GigaGAN [9], InstaFlow-0.9B [18], and Diffusion2GAN trained on SD-CFG-8. We observe that Diffusion2GAN produces more realistic images compared to GigaGAN and InstaFlow-0.9B, while maintaining comparable visual quality with Stable Diffusion 1.5.

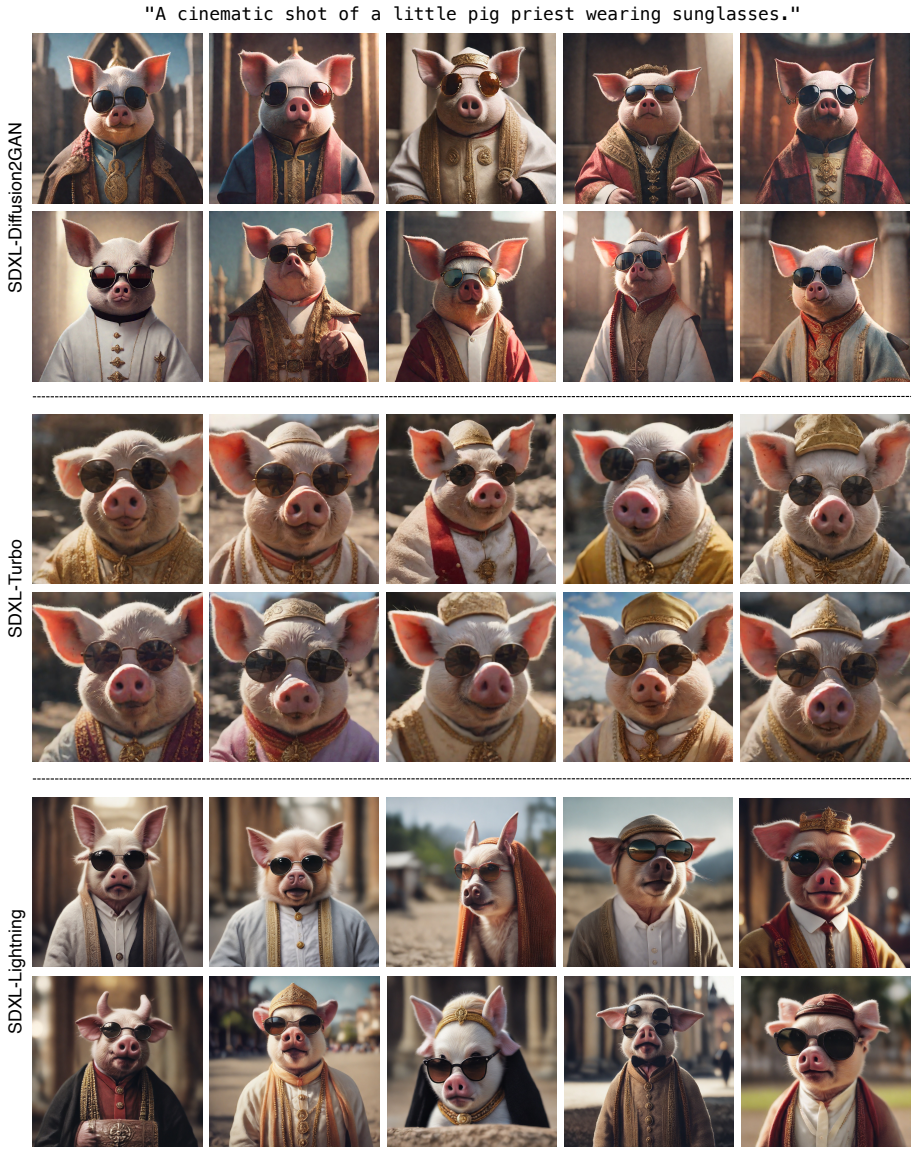

Fig. A7: Diversity of generated images from one-step diffusion distillation models. By altering the random seed used for sampling Gaussian noises, Diffusion2GAN can generate diverse images that closely align with the provided prompt.

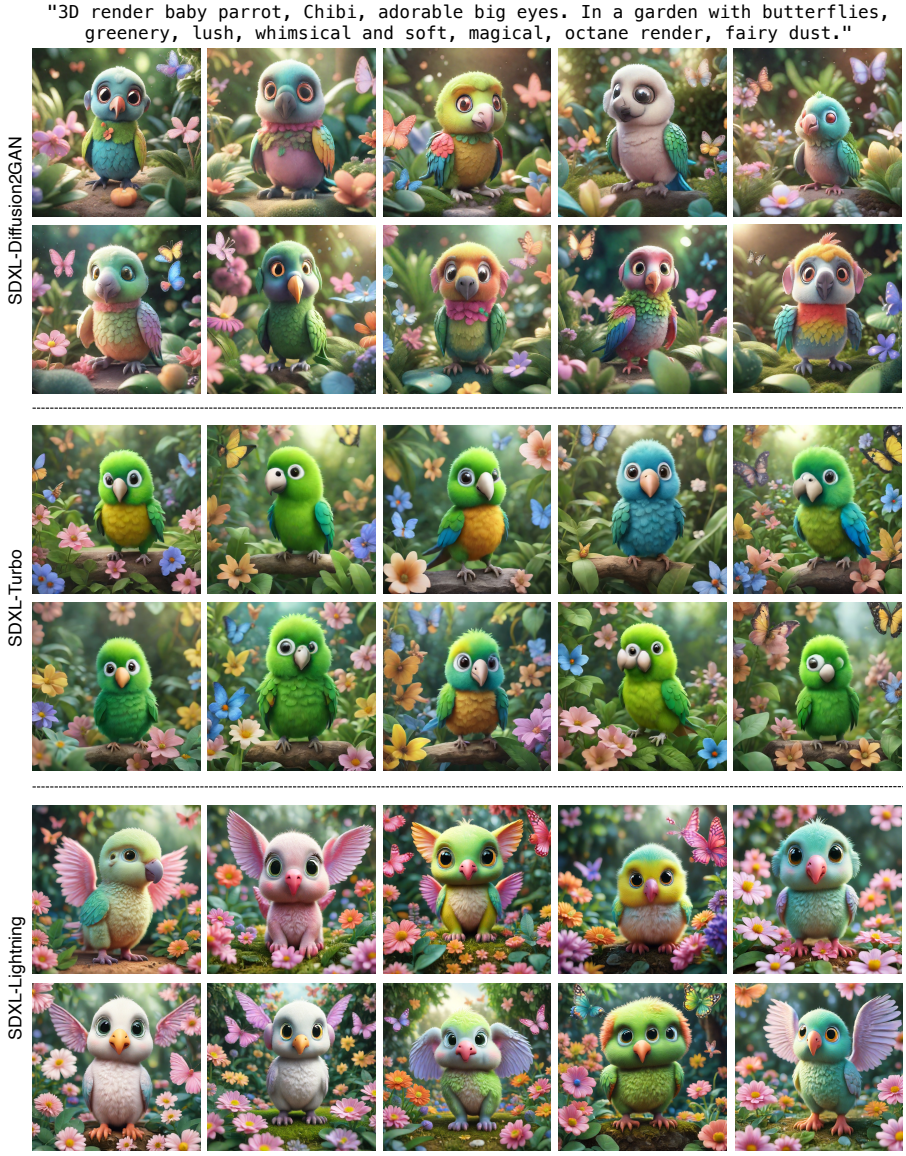

Fig. A8: Diversity of generated images from one-step diffusion distillation models. By altering the random seed used for sampling Gaussian noises, Diffusion2GAN can generate diverse images that closely align with the provided prompt.

References

1. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Li, F.F.: ImageNet: A large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2009) 5
2. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 (2017) 3, 5
3. Fu, S., Tamir, N., Sundaram, S., Chai, L., Zhang, R., Dekel, T., Isola, P.: DreamSim: Learning New Dimensions of Human Visual Similarity using Synthetic Data. In: Conference on Neural Information Processing Systems (NeurIPS) (2023) 2
4. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Nets. In: Conference on Neural Information Processing Systems (NeurIPS) (2014) 3
5. Hessel, J., Holtzman, A., Forbes, M., Bras, R.L., Choi, Y.: Clipscore: A reference-free evaluation metric for image captioning. arXiv preprint arXiv:2104.08718 (2021) 2, 6
6. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In: Conference on Neural Information Processing Systems (NeurIPS) (2017) 2, 7
7. Ho, J., Salimans, T.: Classifier-free diffusion guidance. In: Conference on Neural Information Processing Systems (NeurIPS) Workshop (2022) 11, 12, 13
8. Kang, M., Zhu, J.Y., Zhang, R., Park, J., Shechtman, E., Paris, S., Park, T.: Scaling up GANs for Text-to-Image Synthesis. https://github.com/mingukkang/GigaGAN/tree/main/evaluation 6
9. Kang, M., Zhu, J.Y., Zhang, R., Park, J., Shechtman, E., Paris, S., Park, T.: Scaling up gans for text-to-image synthesis. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2023) 2, 3, 6, 7, 8, 12, 13
10. Karras, T., Aittala, M., Aila, T., Laine, S.: Elucidating the Design Space of Diffusion-Based Generative Models. In: Conference on Neural Information Processing Systems (NeurIPS) (2022) 2, 3, 7
11. Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., Aila, T.: Training generative adversarial networks with limited data. In: Conference on Neural Information Processing Systems (NeurIPS) (2020) 3, 4, 5
12. Karras, T., Aittala, M., Lehtinen, J., Hellsten, J., Aila, T., Laine, S.: Analyzing and Improving the Training Dynamics of Diffusion Models. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2024) 7
13. Kettunen, M., Härkönen, E., Lehtinen, J.: E-lpips: robust perceptual image similarity via random transformation ensembles. arXiv preprint arXiv:1906.03973 (2019) 5
14. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. arXiv preprint arXiv 1412.6980 (2015) 3
15. Krizhevsky, A.: Learning Multiple Layers of Features from Tiny Images. Ph.D. thesis, University of Toronto (2012) 2
16. Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., Aila, T.: Improved Precision and Recall Metric for Assessing Generative Models. In: Conference on Neural Information Processing Systems (NeurIPS) (2019) 2, 6, 7
17. Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., Han, J.: On the variance of the adaptive learning rate and beyond. In: International Conference on Learning Representations (ICLR) (2020) 3

18. Liu, X., Zhang, X., Ma, J., Peng, J., qiang liu: InstaFlow: One Step is Enough for High-Quality Diffusion-Based Text-to-Image Generation. In: International Conference on Learning Representations (ICLR) (2024) 8, 11, 12, 13
19. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In: Conference on Neural Information Processing Systems (NeurIPS) (2022) 2
20. Luo, S., Tan, Y., Patil, S., Gu, D., von Platen, P., Passos, A., Huang, L., Li, J., Zhao, H.: LCM-LoRA: A Universal Stable-Diffusion Acceleration Module. arXiv preprint arXiv:2310.04378 (2023) 11
21. Meng, C., Rombach, R., Gao, R., Kingma, D., Ermon, S., Ho, J., Salimans, T.: On distillation of guided diffusion models. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2023) 7
22. Parmar, G., Zhang, R., Zhu, J.Y.: On Aliased Resizing and Surprising Subtleties in GAN Evaluation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2022) 2
23. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: Conference on Neural Information Processing Systems (NeurIPS) (2019) 1
24. Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: SDXL: Improving latent diffusion models for high-resolution image synthesis. In: International Conference on Learning Representations (ICLR) (2024) 1, 3, 9, 10
25. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2022) 3, 6
26. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: Stable Diffusion 1.5. https://github.com/runwayml/stable-diffusion, accessed: 2022-11-06 1, 8, 9, 10, 11, 12, 13
27. Sajjadi, M.S., Bachem, O., Lucic, M., Bousquet, O., Gelly, S.: Assessing generative models via precision and recall. In: Conference on Neural Information Processing Systems (NeurIPS) (2018) 2
28. Salimans, T., Ho, J.: Progressive Distillation for Fast Sampling of Diffusion Models. In: International Conference on Learning Representations (ICLR) (2022) 2
29. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014) 5
30. Song, J., Meng, C., Ermon, S.: Denoising Diffusion Implicit Models. In: International Conference on Learning Representations (ICLR) (2021) 3, 9, 10
31. Song, Y., Dhariwal, P., Chen, M., Sutskever, I.: Consistency Models. In: International Conference on Machine Learning (ICML) (2023) 2
32. Xu, Y., Zhao, Y., Xiao, Z., Hou, T.: Ufogen: You forward once large scale text-to-image generation via diffusion gans. arXiv preprint arXiv:2311.09257 (2023) 3
33. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 3, 5, 6
34. Zhao, S., Liu, Z., Lin, J., Zhu, J.Y., Han, S.: Differentiable augmentation for data-efficient gan training. In: Conference on Neural Information Processing Systems (NeurIPS) (2020) 3, 4, 5