Table A2. **Hyperparameters for GigaGAN training.** We denote Projection Discriminator [62] as PD, R1 regularization [61] as R1, Learned Perceptual Image Patch Similarity [105] as LPIPS, Adam with decoupled weight decay [56] as AdamW, and the pretrained VIT-B/32 visual encoder [71] as CLIP-ViT-B/32-V.

Task	Class-Label-to-Image		Text-to-Image		Super-Resolution
Dataset & Resolution	ImageNet 64	ImageNet 64→256	LAION&COYO 64	LAION&COYO 64→512	ImageNet 64→256
$\mathbf{z}$ dimension	64	128	128	128	128
$\mathbf{w}$ dimension	512	512	1024	512	512
Adversarial loss type	Logistic	Logistic	Logistic	Logistic	Logistic
Conditioning loss type	PD	PD	MS-I/O	MS-I/O	-
R1 strength	0.2048	0.2048	0.2048 ∼ 2.048	0.2048	0.2048
R1 interval	16	16	16	16	16
G Matching loss strength	-	-	1.0	1.0	-
D Matching loss strength	-	-	1.0	1.0	-
LPIPS strength	-	100.0	-	10.0	100.0
CLIP loss strength	-	-	0.2 ∼ 1.0	1.0	-
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
Batch size	1024	256	512∼1024	192∼320	256
G learning rate	0.0025	0.0025	0.0025	0.0025	0.0025
D learning rate	0.0025	0.0025	0.0025	0.0025	0.0025
β_1 for AdamW	0.0	0.0	0.0	0.0	0.0
β_2 for AdamW	0.99	0.99	0.99	0.99	0.99
Weight decay strength	0.00001	0.00001	0.00001	0.00001	0.00001
Weight decay strength on attention	-	-	0.01	0.01	-
# D updates per G update	1	1	1	1	1
G ema beta	0.9651	0.9912	0.9999	0.9890	0.9912
Precision	TF32	TF32	TF32	TF32	TF32
Mapping Network M layer depth	2	4	4	4	4
Text Transformer T layer depth	-	-	4	2	-
G channel base	32768	32768	16384	32768	32768
D channel base	32768	32768	16384	32768	32768
G channel max	512	512	1600	512	512
D channel max	768	512	1536	512	512
G # of filters N for adaptive kernel selection	8	4	[1, 1, 2, 4, 8]	[1, 1, 1, 1, 1, 2, 4, 8, 16, 16, 16, 16]	4
Attention type	self	self	self + cross	self + cross	self
G attention resolutions	[8, 16, 32]	[16, 32]	[8, 16, 32]	[8, 16, 32, 64]	[16, 32]
D attention resolutions	[8, 16, 32]	-	[8, 16, 32]	[8, 16]	-
G attention depth	[4, 4, 4]	[4, 2]	[2, 2, 1]	[2, 2, 2, 1]	[4, 2]
D attention depth	[1, 1, 1]	-	[2, 2, 1]	[2, 2]	-
Attention dimension multiplier	1.0	1.4	1.0	1.0	1.4
MLP dimension multiplier of attention	4.0	4.0	4.0	4.0	4.0
# synthesis block per resolution	1	5	[3, 3, 3, 2, 2]	[4, 4, 4, 4, 4, 4, 3]	5
# discriminator block per resolution	1	1	[1, 2, 2, 2, 2]	1	-
Residual gain	1.0	0.4	0.4	0.4	0.4
Residual gain on attention	1.0	0.3	0.3	0.5	0.3
MinibatchStdLayer	True	True	False	True	True
D epilogue mbstd group size	8	4	-	2	4
Multi-scale training	False	False	True	True	False
Multi-scale loss ratio (high to low res)	-	-	[0.33, 0.17, 0.17, 0.17, 0.17]		-
D intermediate layer adv loss weight	-	-	0.01	[0.2, 0.2, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]	-
D intermediate layer matching loss weight	-	-	0.05	-	-
Vision-aided discriminator backbone	-	-	CLIP-ViT-B/32-V	-	-
G Model size	209.5M	359.8M	652.5M	359.1M	359.0M
D Model size	76.7M	30.7M	381.4M	130.1M	28.9M
Iterations	300k	620k	1350k	915k	160k
# A100 GPUs for training	64	64	96∼128	64	32

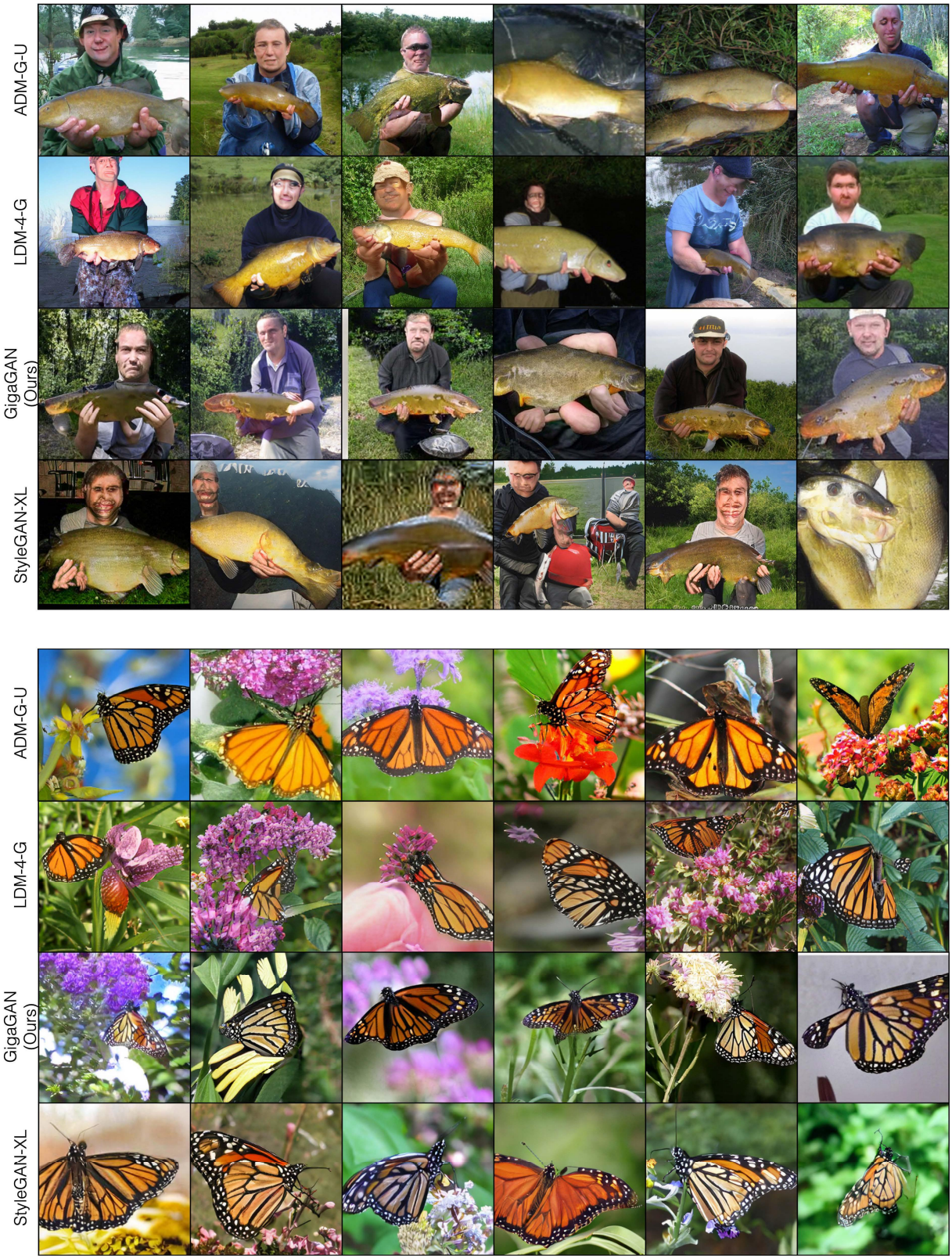


Figure A1. Uncurated images (above: Tench and below: Monarch) from ADM-G-U [15], LDM-4-G [79], GigaGAN (ours), and StyleGAN-XL [86]. FID values of each generative model are 4.01, 3.60, 3.45, and 2.32, respectively.

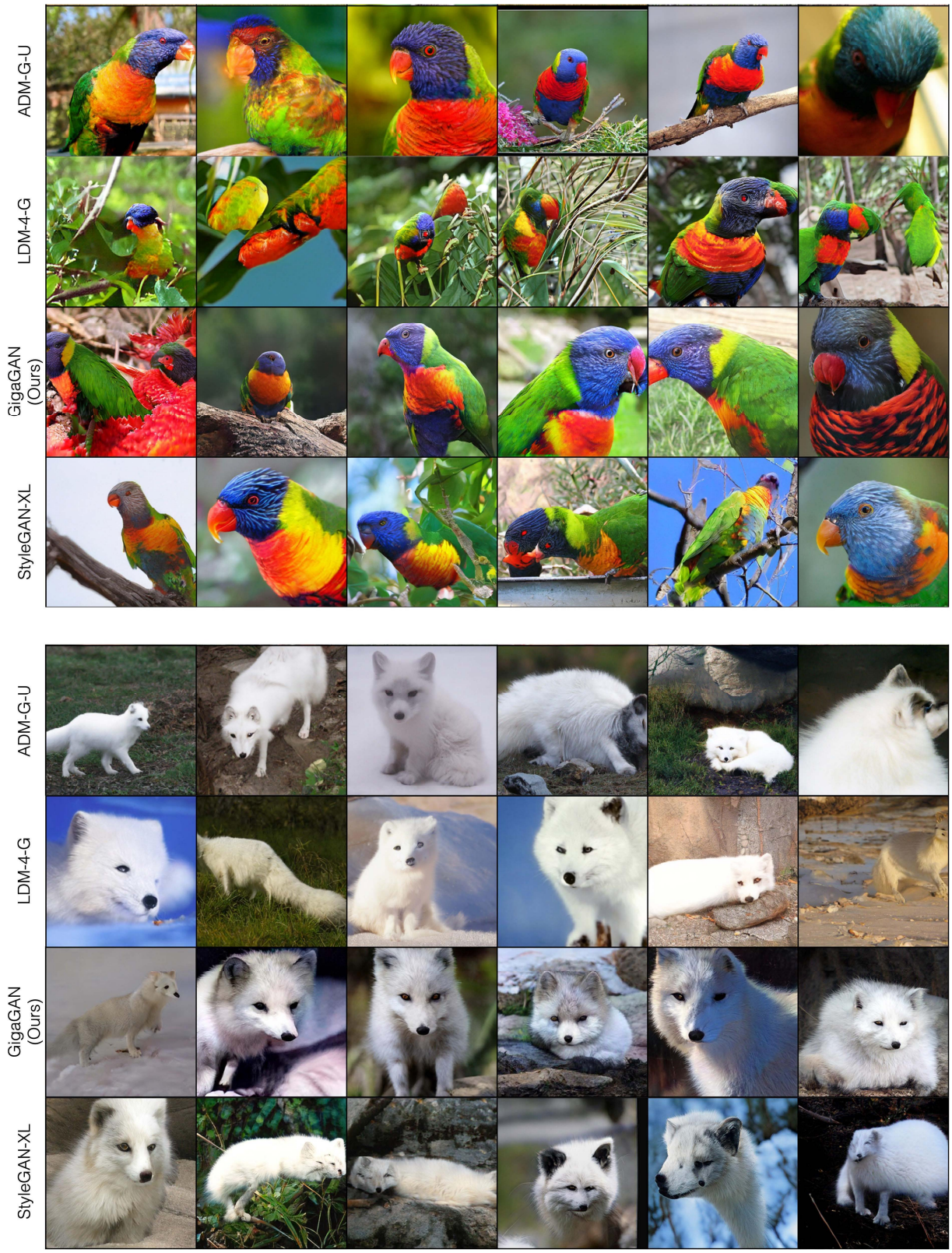


Figure A2. Uncurated images (above: Lorikeet and below: Arctic fox) from ADM-G-U [15], LDM-4-G [79], GigaGAN (ours), and StyleGAN-XL [86]. FID values of each generative model are 4.01, 3.60, 3.45, and 2.32, respectively.

C. Text-to-image synthesis results

C.1. Truncation trick at inference

Similar to the classifier guidance [15] and classifier-free guidance [28] used in diffusion models such as LDM, our GAN model can leverage the truncation trick [6, 41] at inference time.

$$\mathbf{w}_{\text{trunc}} = \text{lerp}(\mathbf{w}_{\text{mean}}, \mathbf{w}, \psi), \tag{9}$$

where $\mathbf{w}_{\text{mean}}$ is the mean of $\mathbf{w}$ of the entire dataset, which can be precomputed. In essence, the truncation trick lets us trade diversity for fidelity by interpolating the latent vector to the mean of the distribution and thereby making the outputs more typical. When $\psi = 1.0$, $\mathbf{w}_{\text{mean}}$ is not used, and there is no truncation. When $\psi = 0.0$, $\mathbf{w}$ collapses to the mean, losing diversity.

While it is straightforward to apply the truncation trick for the unconditional case, it is less clear how to achieve this for text-conditional image generation. We find that interpolating the latent vector toward both the mean of the entire distribution as well as the mean of $\mathbf{w}$ conditioned on the text prompt produces desirable results.

$$\mathbf{w}_{\text{trunc}} = \text{lerp}(\mathbf{w}_{\text{mean},\mathbf{c}}, \text{lerp}(\mathbf{w}_{\text{mean}}, \mathbf{w}, \psi), \psi), \tag{10}$$

where $\mathbf{w}_{\text{mean},\mathbf{c}}$ can be computed at inference time by sampling $\mathbf{w} = M(\mathbf{z}, \mathbf{c})$ 16 times with the same $\mathbf{c}$, and taking the average. This operation's overhead is negligible, as the mapping network M is computationally light compared to the synthesis network. At $\psi = 1.0$, $\mathbf{w}_{\text{trunc}}$ becomes $\mathbf{w}_{\text{trunc}} = \mathbf{w}$, meaning no truncation. Figure A4 demonstrates the effect of our text-conditioned truncation trick.

Quantitatively, the effect of truncation is similar to the guidance technique of diffusion models. As shown in Figure A3, the CLIP score increases with more truncation, where the FID increases due to reduced diversity.

C.2. Comparison to diffusion models

Finally, we show randomly sampled results of our model and compare them with publicly available diffusion models, LDM [79], Stable Diffusion [78], and DALL·E 2 [74].

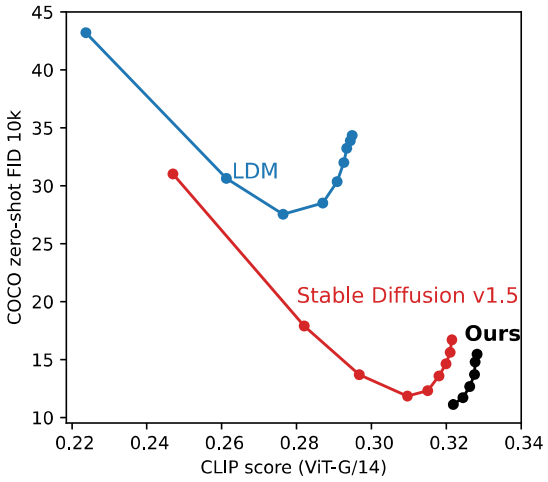


Figure A3. We investigate how our FID and CLIP score changes over different truncation values [1.0, 0.9, 0.8, 0.7, 0.6, 0.5], by visualizing them along with the FID-CLIP score curve of two publicly available large scale diffusion models: LDM and Stable Diffusion. It is seen that the CLIP score increases with more truncation, at the cost of reduced diversity indicated by higher FID. The guidance values of the diffusion models are [1.0, 1.25, 1.5, 1.75, 2, 4, 6, 8, 10].

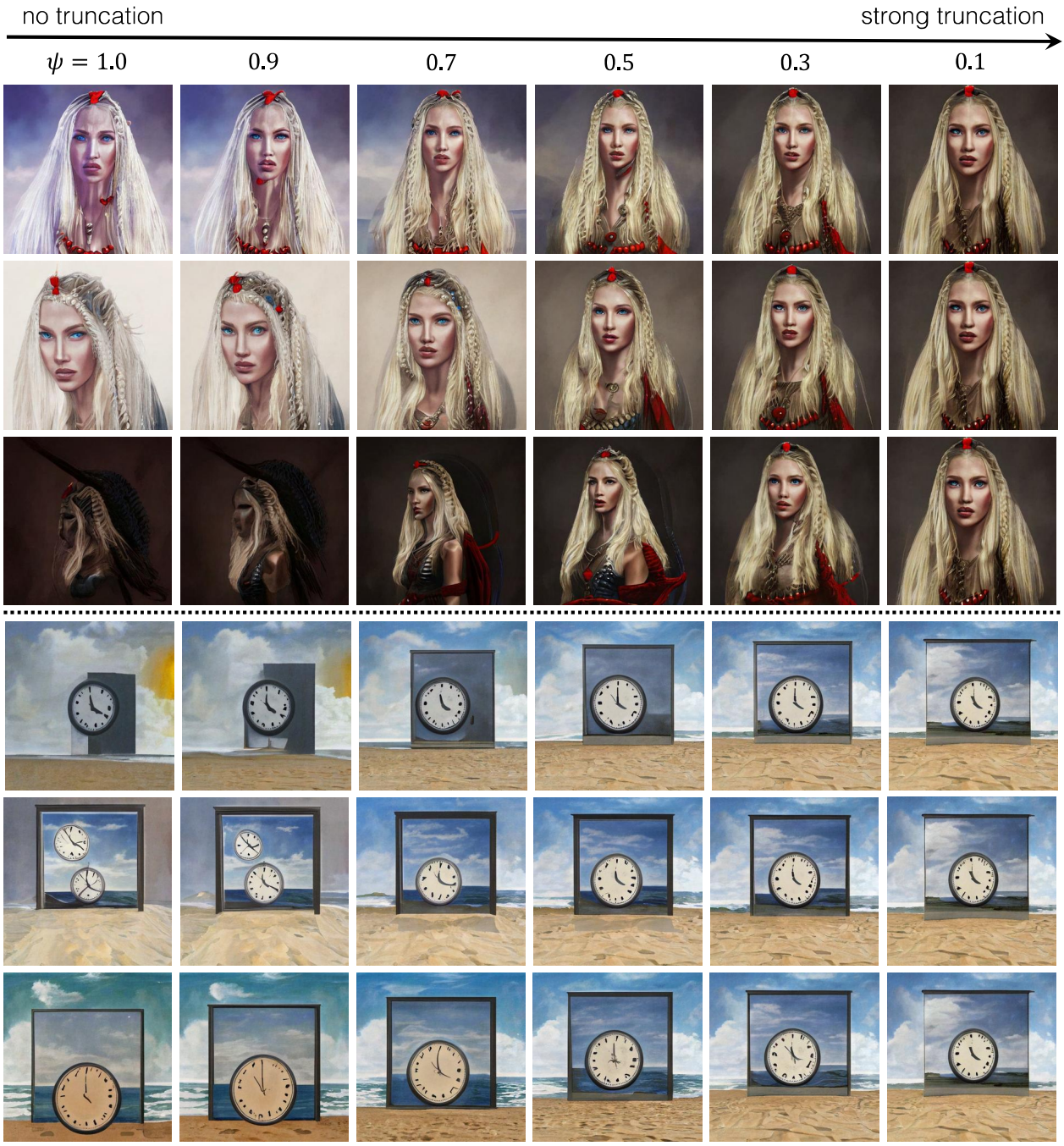


Figure A4. **The visual effect of our truncation trick**. We demonstrate the effect of truncation by decreasing the truncation value ψ from 1.0. We show six example outputs with the text prompt "digital painting of a confident and severe looking northern war goddess, extremely long blond braided hair, beautiful blue eyes and red lips." and "Magritte painting of a clock on a beach.". At 1.0 (no truncation), the diversity is high, but the alignment is not satisfactory. As the truncation increases, text-image alignment improves, at the cost of diversity. We find that a truncation value between 0.8 and 0.7 produces the best result.

Figure A5. **Style mixing.** GigaGAN maintains a disentangled latent space, allowing us to blend the coarse style of one sample with the fine style of another. The corresponding latent codes are spliced together to produce a style-swapping grid. The outputs are generated from the same prompt but with different latent codes.

Figure A6. **Prompt interpolation**. GigaGAN enables smooth interpolation between prompts, as shown in the interpolation grid. The four corners are generated from the same latent but with different text prompts. The corresponding text embeddings and style vectors are interpolated to create a smooth transition. The same results in similar layouts.

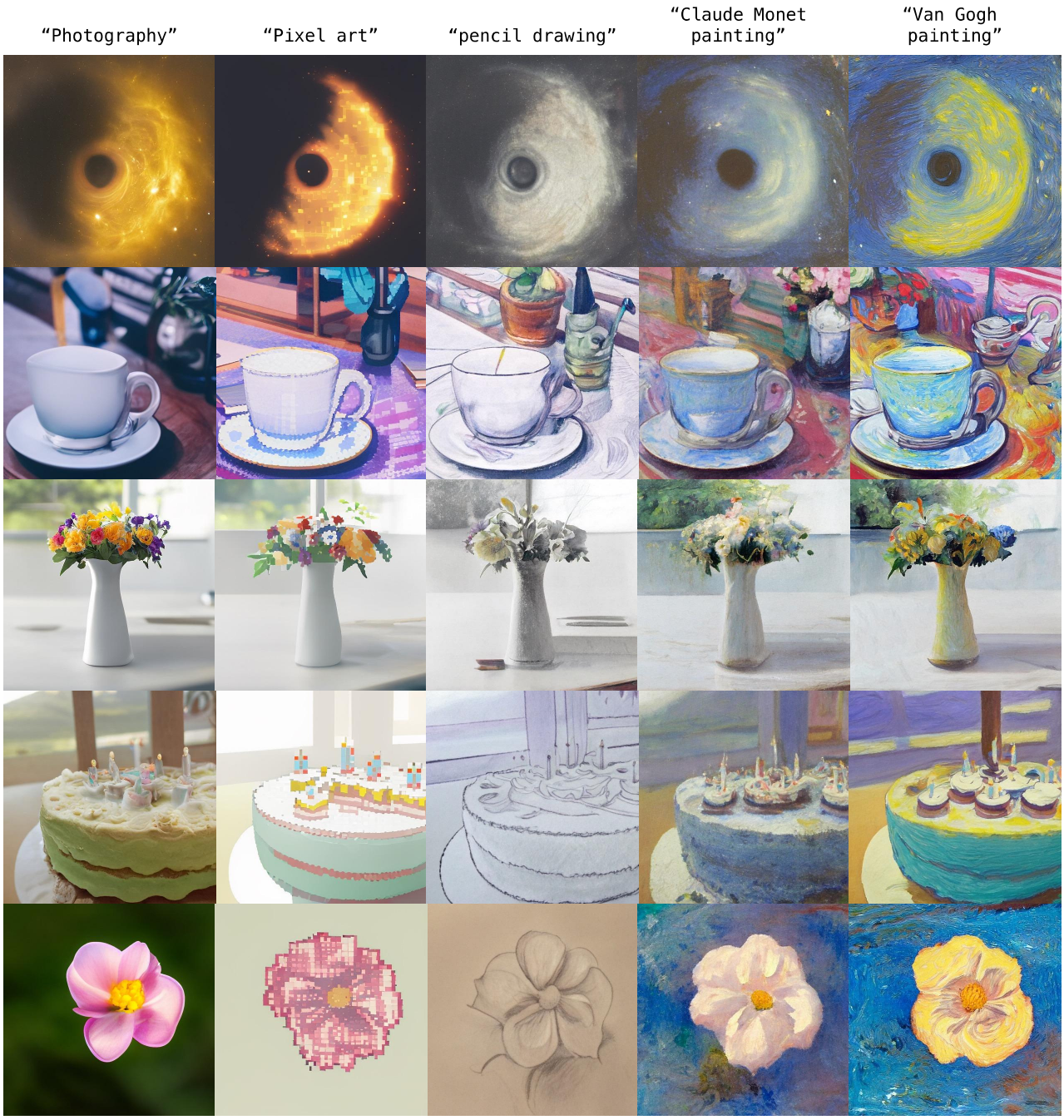


Figure A7. **Prompt mixing**. GigaGAN can directly control the style with text prompts. Here we generate five outputs using the prompts “Photography of X”, shown in the “Photography” column. Then we re-compute the text embeddings $\mathbf{t}$ and the style codes $\mathbf{w}$ using the new prompts “Y of X”, such as “Van Gogh painting of the black hole in the space”, and apply them to the second half layers of the generator, achieving layout-preserving style control. Cross-attention mechanism automatically localizes the style to the object of interest. We use the following prompts in order from the row above. (1) the black hole in the space. (2) a teacup on the desk. (3) a table top with a vase of flowers on it. (4) a birthday cake. (5) a beautiful flower. We discover that GigaGAN’s prompt-based style transfer is only possible for images of a single and simple object.

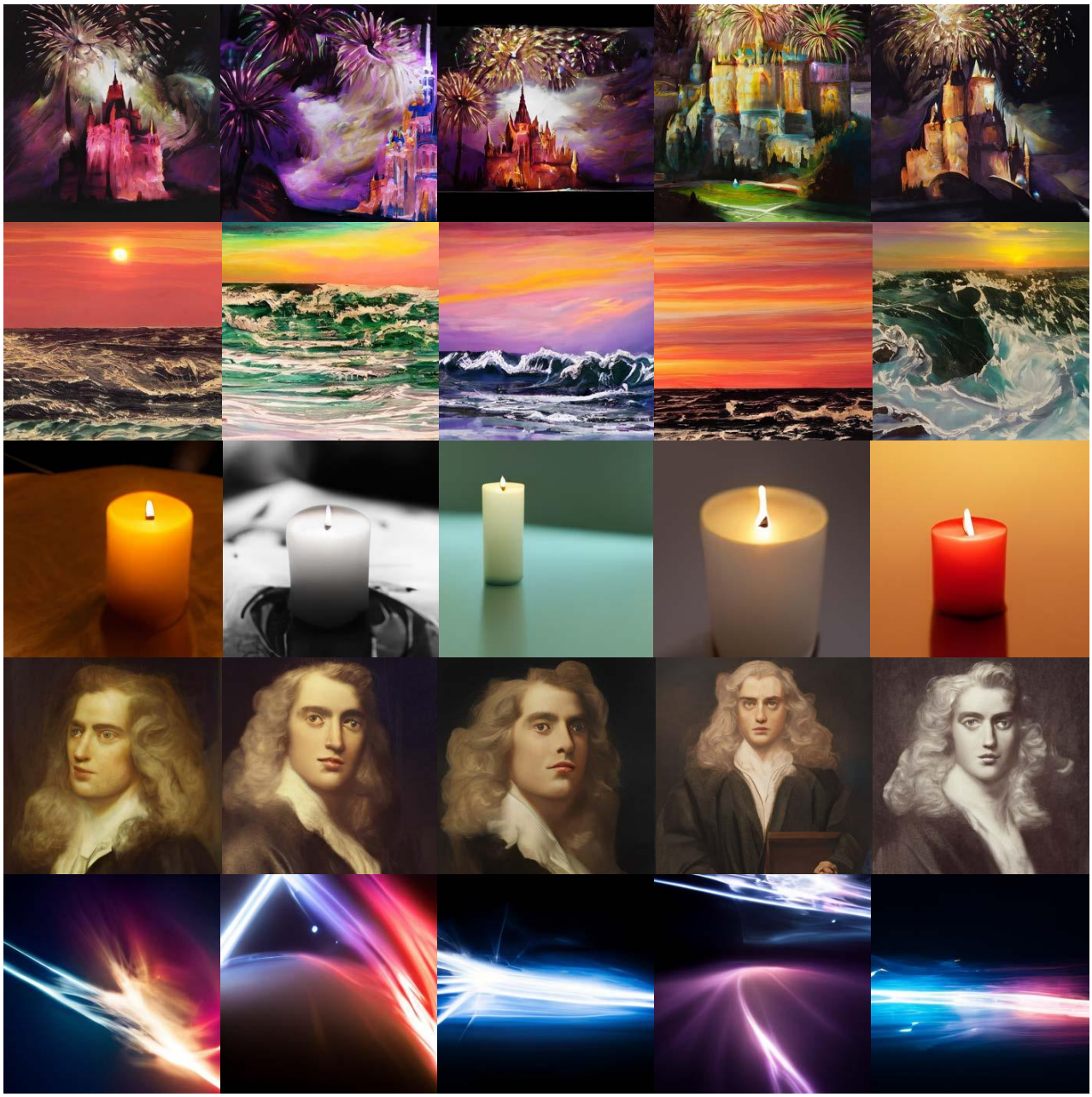


Figure A8. **Diversity of synthesized images using GigaGAN**. GigaGAN can synthesize diverse images for a given prompt. We use the following prompts in order from the row above. (1) Majestic castle and fireworks, art work, oil painting, 2k. (2) Oil-painting depicting a sunset over the sea with waves. (3) A burning candle with tho wicks, detailed photo, studio lighting. (4) Portrait of Isaac Newton, long hair. (5) An abstract representation of the speed of light.

“A loft bed with a dresser underneath it.”

Ours (512px, 0.13s / img)

Ours (512px, 0.14s / img, truncation $\psi = 0.8$)

LDM (256px, 9.4s / img, 250 steps, guidance=6.0)

Stable Diffusion v1.5 (512px, 2.9s / img, 50 steps, guidance=7.5)

DALL·E 2 (1024px)

Figure A9. Random outputs of our model, Latent Diffusion Model [79], Stable Diffusion [78], and DALL·E 2 [74], using prompt “A loft bed with a dresser underneath it”. We show two versions of our model, one without truncation and the other with truncation. Our model enjoys faster speed than the diffusion models. Still, we observe our model falls behind in structural coherency, such as the number of legs of the bed frames. For LDM and Stable Diffusion, we use 250 and 50 sampling steps with DDIM / PLMS [55], respectively. For DALL·E 2, we generate images using the official DALL·E service [64].

“A green vase filed with red roses sitting on top of table.”

Ours (512px, 0.13s / img)

Ours (512px, 0.14s / img, truncation $\psi = 0.8$)

LDM (256px, 9.4s / img, 250 steps, guidance=6.0)

Stable Diffusion v1.5 (512px, 2.9s / img, 50 steps, guidance=7.5)

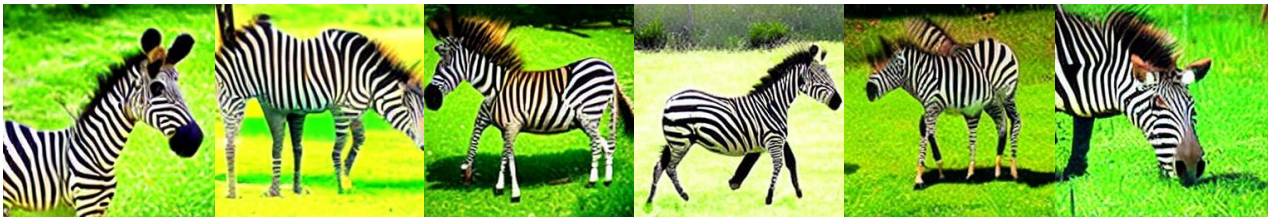
DALL·E 2 (1024px)

Figure A10. Random outputs of our model, Latent Diffusion Model [79], Stable Diffusion [78], and DALL·E 2 [74], using prompt “A green vase filed with red roses sitting on top of table”. We show two versions of our model, one without truncation and the other with truncation. Our model enjoys faster speed than the diffusion models in both cases. Still, we observe our model falls behind in structural coherency like the symmetry of the vases. For LDM and Stable Diffusion, we use 250 and 50 sampling steps with DDIM / PLMS [55], respectively. For DALL·E 2, we generate images using the official DALL·E service [64].

“A zebra in the grass who is cleaning himself.”

Ours (512px, 0.13s / img)

Ours (512px, 0.14s / img, truncation $\psi = 0.8$)



LDM (256px, 9.4s / img, 250 steps, guidance=6.0)

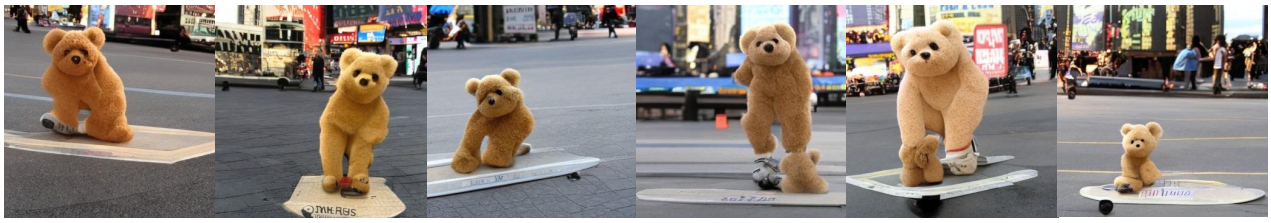
Stable Diffusion v1.5 (512px, 2.9s / img, 50 steps, guidance=7.5)

DALL·E 2 (1024px)

Figure A11. Random outputs of our model, Latent Diffusion Model [79], Stable Diffusion [78], and DALL·E 2 [74], using prompt “A zebra in the grass who is cleaning himself”. We show two versions of our model, one without truncation and the other with truncation. Our model enjoys faster speed than the diffusion models in both cases. Still, we observe our model falls behind in details, such as the precise stripe pattern of the positioning of eyes. For LDM and Stable Diffusion, we use 250 and 50 sampling steps with DDIM / PLMS [55], respectively. For DALL·E 2, we generate images using the official DALL·E service [64].

“A teddy bear on a skateboard in times square.”

Ours (512px, 0.13s / img)



Ours (512px, 0.14s / img, truncation $\psi = 0.8$)

LDM (256px, 9.4s / img, 250 steps, guidance=6.0)

Stable Diffusion v1.5 (512px, 2.9s / img, 50 steps, guidance=7.5)

DALL·E 2 (1024px)

Figure A12. Random outputs of our model, Latent Diffusion Model [79], Stable Diffusion [78], and DALL·E 2 [74], using prompt “A teddy bear on a skateboard in times square”. We show two versions of our model, one without truncation and the other with truncation. Our model enjoys faster speed than the diffusion models in both cases. Still, we observe our model falls behind in details, like the exact shape of skateboards. For LDM and Stable Diffusion, we use 250 and 50 sampling steps with DDIM / PLMS [55], respectively. For DALL·E 2, we generate images using the official DALL·E service [64].

“Vibrant portrait painting of Salvador Dalí with a robotic half face.”

Ours (512px, 0.13s / img)

Ours (512px, 0.14s / img, truncation $\psi = 0.8$)

LDM (256px, 9.4s / img, 250 steps, guidance=6.0)

Stable Diffusion v1.5 (512px, 2.9s / img, 50 steps, guidance=7.5)

DALL·E 2 (1024px)

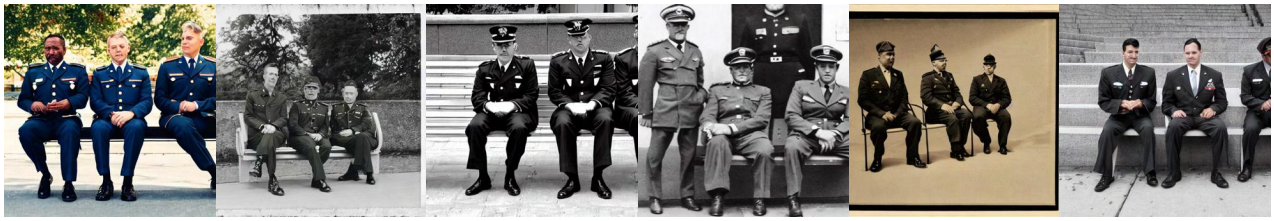
Figure A13. Random outputs of our model, Latent Diffusion Model [79], Stable Diffusion [78], and DALL·E 2 [74], using prompt “Vibrant portrait painting of Salvador Dalí with a robotic half face”. We show two versions of our model, one without truncation and the other with truncation. Our model enjoys faster speed than the diffusion models in both cases. Still, we observe our model falls behind in structural details like in the detailed shape of eyes. For LDM and Stable Diffusion, we use 250 and 50 sampling steps with DDIM / PLMS [55], respectively. For DALL·E 2, we generate images using the official DALL·E service [64].

“Three men in military suits are sitting on a bench.”

Ours (512px, 0.13s / img)

Ours (512px, 0.14s / img, truncation $\psi = 0.8$)

LDM (256px, 9.4s / img, 250 steps, guidance=6.0)



Stable Diffusion v1.5 (512px, 2.9s / img, 50 steps, guidance=7.5)

DALL·E 2 (1024px)

Figure A14. Random outputs of our model, Latent Diffusion Model [79], Stable Diffusion [78], and DALL·E 2 [74], using prompt “Three men in military suits are sitting on a bench”. We show two versions of our model, one without truncation and the other with truncation. Our model enjoys faster speed than the diffusion models in both cases. Still, we observe our model falls behind in details in facial expression and attire. For LDM and Stable Diffusion, we use 250 and 50 sampling steps with DDIM / PLMS [55], respectively. For DALL·E 2, we generate images using the official DALL·E service [64].

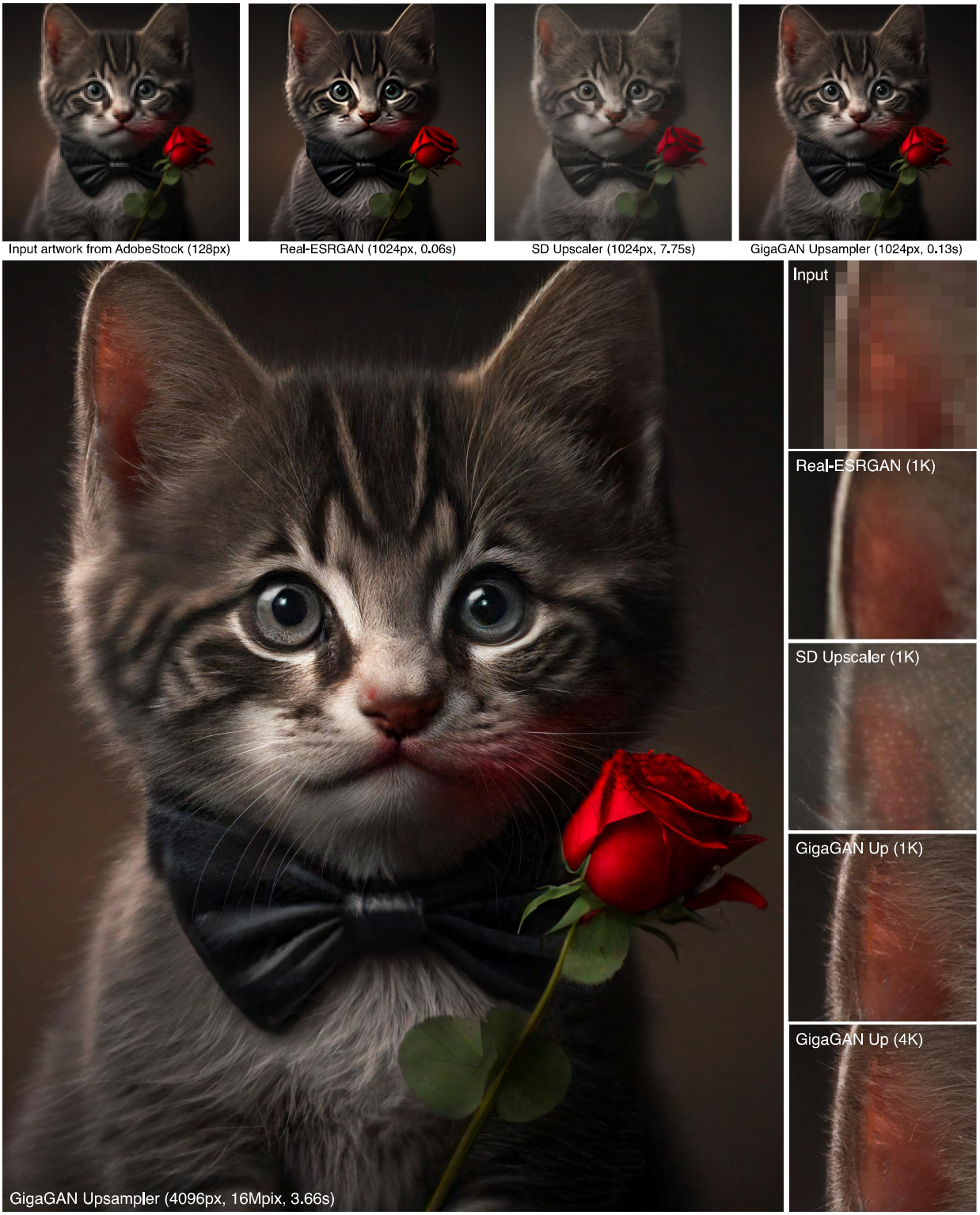


Figure A15. **Our GAN-based upsampler** can serve as the upsampler for many text-to-image models that generate initial outputs at low resolutions like 64px or 128px. We simulate such usage by applying our 8× superresolution model on a low-res 128px artwork to obtain the 1K output, using "Portrait of a kitten dressed in a bow tie. Red Rose. Valentine's day.". Then our model can be re-applied to go beyond 4K. We compare our model with the text-conditioned upscaler of Stable Diffusion [78] and unconditional Real-ESRGAN [33]. Zooming in is recommended for comparison between 1K and 4K outputs.

Figure A16. **Our GAN-based upsampler** can serve as the upsampler for many text-to-image models that generate initial outputs at low resolutions like 64px or 128px. We simulate such usage by applying our 8× superresolution model on a low-res 128px artwork to obtain the 1K output, using "Heart shaped pancakes with honey and strawberry for Valentine's Day". Then our model can be re-applied to go beyond 4K. We compare our model with the text-conditioned upscaler of Stable Diffusion [78] and unconditional Real-ESRGAN [33]. Zooming in is recommended for comparison between 1K and 4K outputs.

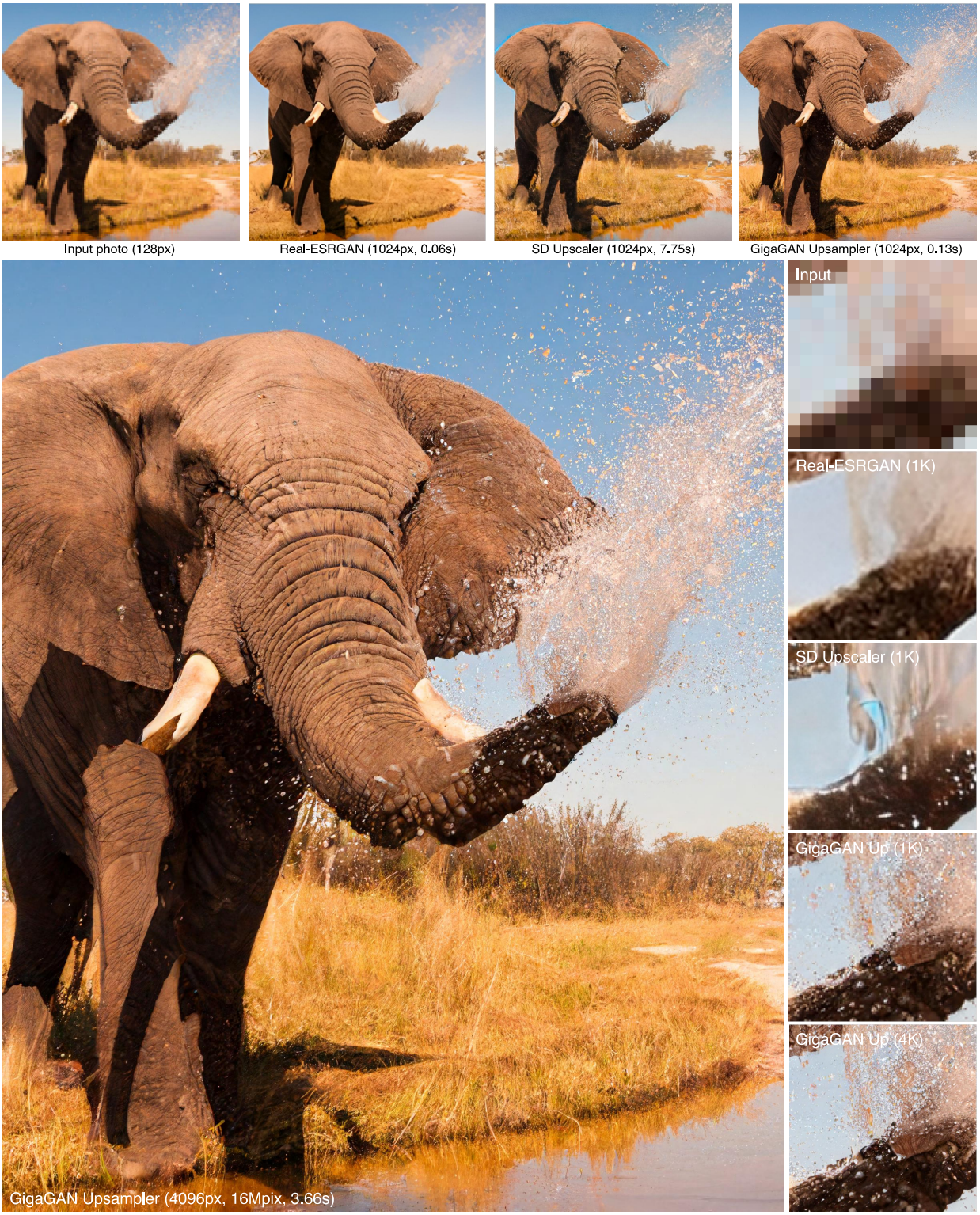


Figure A17. **Our GAN-based upsampler** can also be used as an off-the-shelf superresolution model for real images with a large scaling factor by providing an appropriate description of the image. We apply our text-conditioned 8× superresolution model on a low-res 128px photo to obtain the 1K output, using "An elephant spraying water with its trunk". Then our model can be re-applied to go beyond 4K. We compare our model with the text-conditioned upscaler of Stable Diffusion [78] and unconditional Real-ESRGAN [33]. Zooming in is recommended for comparison between 1K and 4K outputs.