Scaling up GANs for Text-to-Image Synthesis

Minguk Kang[1,3] Jun-Yan Zhu[2] Richard Zhang[3]
Jaesik Park[1] Eli Shechtman[3] Sylvain Paris[3] Taesung Park[3]

[1]POSTECH [2]Carnegie Mellon University [3]Adobe Research

Abstract

The recent success of text-to-image synthesis has taken the world by storm and captured the general public's imagination. From a technical standpoint, it also marked a drastic change in the favored architecture to design generative image models. GANs used to be the de facto choice, with techniques like StyleGAN. With DALL·E 2, autoregressive and diffusion models became the new standard for large-scale generative models overnight. This rapid shift raises a fundamental question: can we scale up GANs to benefit from large datasets like LAION? We find that naïvely increasing the capacity of the StyleGAN architecture quickly becomes unstable. We introduce GigaGAN, a new GAN architecture that far exceeds this limit, demonstrating GANs as a viable option for text-to-image synthesis. GigaGAN offers three major advantages. First, it is orders of magnitude faster at inference time, taking only 0.13 seconds to synthesize a 512px image. Second, it can synthesize high-resolution images, for example, 16-megapixel images in 3.66 seconds. Finally, GigaGAN supports various latent space editing applications such as latent interpolation, style mixing, and vector arithmetic operations.

1. Introduction

Recently released models, such as DALL·E 2 [74], Imagen [80], Parti [101], and Stable Diffusion [79], have ushered in a new era of image generation, achieving unprecedented levels of image quality and model flexibility. The now-dominant paradigms, diffusion models and autoregressive models, both rely on iterative inference. This is a double-edged sword, as iterative methods enable stable training with simple objectives but incur a high computational cost during inference.

Contrast this with Generative Adversarial Networks (GANs) [6,21,41,72], which generate images through a single forward pass and thus inherently efficient. While such models dominated the previous "era" of generative modeling, scaling them requires careful tuning of the network architectures and training considerations due to instabilities in the training procedure. As such, GANs have excelled at modeling single or multiple object classes, but scaling to complex datasets, much less an open world, has remained challenging. As a result, ultra-large models, data, and compute resources are now dedicated to diffusion and autoregressive models. In this work, we ask – *can GANs continue to be scaled up and potentially benefit from such resources, or have they plateaued? What prevents them from further scaling, and can we overcome these barriers?*

We first experiment with StyleGAN2 [42] and observe that simply scaling the backbone causes unstable training. We identify several key issues and propose techniques to stabilize the training while increasing the model capacity. First, we effectively scale the generator's capacity by retaining a bank of filters and taking a sample-specific linear combination. We also adapt several techniques commonly used in the diffusion context and confirm that they bring similar benefits to GANs. For instance, interleaving both self-attention (image-only) and cross-attention (image-text) with the convolutional layers improves performance.

Furthermore, we reintroduce multi-scale training, finding a new scheme that improves image-text alignment and low-frequency details of generated outputs. Multi-scale training allows the GAN-based generator to use parameters in low-resolution blocks more effectively, leading to better image-text alignment and image quality. After careful tuning, we achieve stable and scalable training of a one-billion-parameter GAN (GigaGAN) on large-scale datasets, such as LAION2B-en [88]. Our results are shown in Figure 1.

In addition, our method uses a multi-stage approach [14, 104]. We first generate at 64×64 and then upsample to 512×512. These two networks are modular and robust enough to be used in a plug-and-play fashion. We show that our text-conditioned GAN-based upsampling network can be used as an efficient, higher-quality upsampler for a base diffusion model such as DALL·E 2, despite never having seen diffusion images at training time (Figures 2).

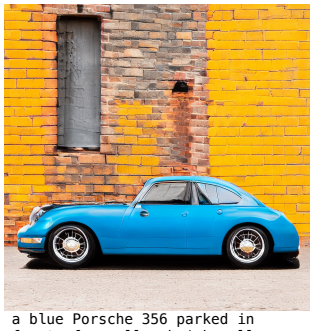
Together, these advances enable our GigaGAN to go far beyond previous GANs: $36\times$ larger than Style-

A portrait of a human growing colorful flowers from her hair. Hyperrealistic oil painting. Intricate details.

A golden luxury motorcycle parked at the King's palace. 35mm f/4.5.

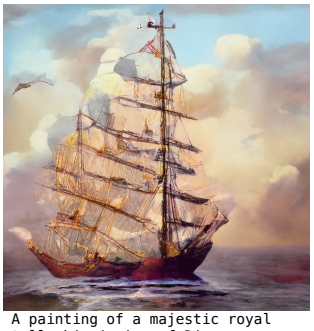
a cute magical flying maltipoo at light speed, fantasy concept art, bokeh, wide sky

A living room with a fireplace at a wood cabin. Interior design.



a blue Porsche 356 parked in front of a yellow brick wall.

Eiffel Tower, landscape photography



A painting of a majestic royal tall ship in Age of Discovery.

Isometric underwater Atlantis city with a Greek temple in a bubble.

A hot air balloon in shape of a heart. Grand Canyon

low poly bunny with cute eyes

A cube made of denim on a wooden table

Figure 1. Our model, GigaGAN, shows GAN frameworks can also be scaled up for general text-to-image synthesis tasks, generating a 512px output at an interactive speed of 0.13s, and 4096px at 3.7s. Selected examples at 2K or 4K resolutions are shown. Please zoom in for more details. See Appendix C and our website for more uncurated comparisons.

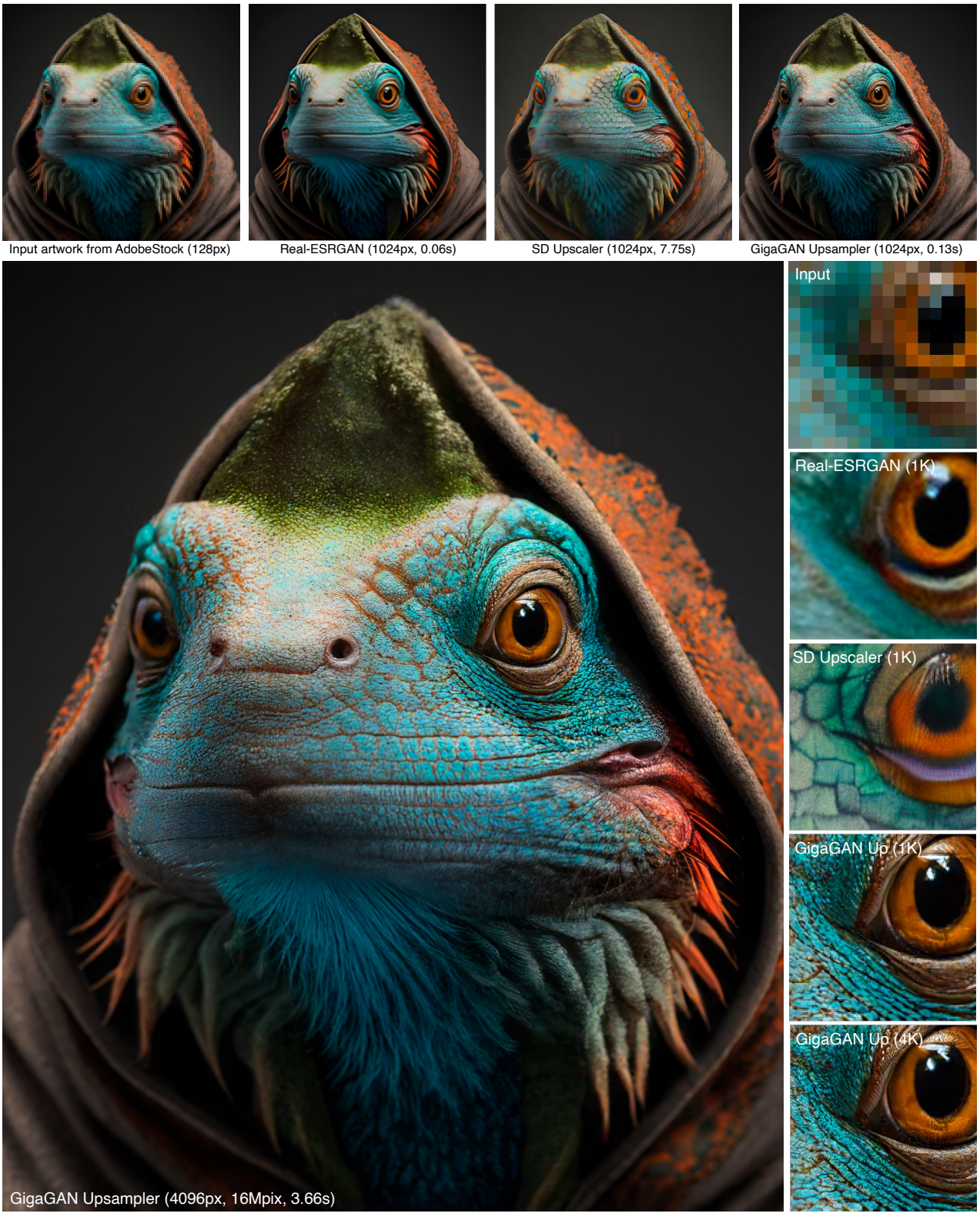


Figure 2. **Our GAN-based upsampler** can serve in the upsampling pipeline of many text-to-image models that often generate initial outputs at low resolutions like 64px or 128px. We simulate such usage by applying our text-conditioned 8× superresolution model on a low-res 128px artwork to obtain the 1K output, using "Portrait of a colored iguana dressed in a hoodie". 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.

Figure 3. **Our GAN-based upsampler**, similar to Figure 2, 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 "A dog sitting in front of a mini tipi tent". 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.

GAN2 [42] and 6× larger than StyleGAN-XL [86] and XMC-GAN [103]. While our 1B parameter count is still lower than the largest recent synthesis models, such as Imagen (3.0B), DALL·E 2 (5.5B), and Parti (20B), we have not yet observed a quality saturation regarding the model size. GigaGAN achieves a zero-shot FID of 9.09 on COCO2014 dataset, lower than the FID of DALL·E 2, Parti-750M, and Stable Diffusion.

Furthermore, GigaGAN has three major practical advantages compared to diffusion and autoregressive models. First, it is orders of magnitude faster, generating a 512px image in 0.13 seconds (Figure 1). Second, it can synthesize ultra high-res images at 4k resolution in 3.66 seconds. Third, it is endowed with a controllable, latent vector space that lends itself to well-studied controllable image synthesis applications, such as style mixing (Figure 6), prompt interpolation (Figure 7), and prompt mixing (Figure 8).

In summary, our model is the first GAN-based method that successfully trains a billion-scale model on billions of real-world complex Internet images. This suggests that GANs are still a viable option for text-to-image synthesis and should be considered for future aggressive scaling. Please visit our website for additional results.

2. Related Works

Text-to-image synthesis. Generating a realistic image given a text description, as first explored by Mansimov et al. [58], is a challenging task. Earlier works adopted text-conditional GANs [76, 77, 93, 99, 104, 111] on specific domains [96] and datasets with a closed-world assumption [54]. With the development of diffusion models [15, 26], autoregressive (AR) transformers [12], and large-scale language encoders [71, 73], text-to-image synthesis has shown remarkable improvement on an open-world of arbitrary text descriptions. GLIDE [63], DALL·E 2 [74], and Imagen [80] are representative diffusion models that show photorealistic outputs with the aid of a pretrained language encoder [71, 73]. AR models such as DALL·E [75], Make-A-Scene [20], CogView [16, 17], and Parti [101] also achieve amazing results. While these models exhibit unprecedented image synthesis ability, they require time-consuming iterative processes to achieve high-quality image sampling.

To accelerate the sampling, several methods propose to reduce the sampling steps [57, 59, 83, 89] or reuse pre-computed features [51]. Latent Diffusion Model (LDM) [79] performs the reverse processes in low-dimensional latent space instead of pixel space. However, consecutive reverse processes are still computationally expensive, limiting the usage of large-scale text-to-image models for interactive applications.

GAN-based image synthesis. GANs [21] have been one of the primary families of generative models for natural image synthesis. As the sampling quality and diversity of GANs improve [39–42, 44, 72, 84], GANs have been deployed to various computer vision and graphics applications, such as text-to-image synthesis [76], image-to-image translation [29, 34, 49, 65, 66, 110], and image editing [1, 7, 69, 109]. Notably, StyleGAN-family models [40, 42] have shown impressive ability in image synthesis tasks for single-category domains [1, 31, 69, 98, 112]. Other works have explored class-conditional GANs [6, 36, 86, 102, 107] on datasets with a fixed set of object categories.

In this paper, we change the data regimes from single- or multi-categories datasets to extremely data-rich situations. We make the first expedition toward training a large-scale GAN for text-to-image generation on a vast amount of web-crawled text and image pairs, such as LAION2B-en [88] and COYO-700M [8]. Existing GAN-based text-to-image synthesis models [52, 76, 93, 99, 103, 104, 111] are trained on relatively small datasets, such as CUB-200 (12k training pairs), MSCOCO (82k) and LN-OpenImages (507k). Also, those models are evaluated on associated validation datasets, which have not been validated to perform large-scale text-image synthesis like diffusion or AR models.

Concurrent with our method, StyleGAN-T [85] and GALIP [92] share similar goals to ours. However, GigaGAN and the aforementioned techniques were developed independently with distinct technical contributions. We hope these methods can complement each other and collectively address the limitations of GANs.

Super-resolution for large-scale text-to-image models. Large-scale models require prohibitive computational costs for both training and inference. To reduce the memory and running time, cutting-edge text-to-image models [63, 74, 80, 101] have adopted cascaded generation processes where images are first generated at 64×64 resolution and upsampled to 256×256 and 1024×1024 sequentially. However, the super-resolution networks are primarily based on diffusion models, which require many iterations. In contrast, our low-res image generators and upsamplers are based on GANs, reducing the computational costs for both stages. Unlike traditional super-resolution techniques [2, 18, 47, 95] that aim to faithfully reproduce low-resolution inputs or handle image degradation like compression artifacts, our upsamplers for large-scale models serve a different purpose. They need to perform larger upsampling factors while potentially leveraging the input text prompt.

3. Method

We train a generator $G(\mathbf{z}, \mathbf{c})$ to predict an image $\mathbf{x} \in \mathbb{R}^{H \times W \times 3}$ given a latent code $\mathbf{z} \sim \mathcal{N}(0, 1) \in \mathbb{R}^{128}$ and text-conditioning signal $\mathbf{c}$. We use a discriminator $D(\mathbf{x}, \mathbf{c})$

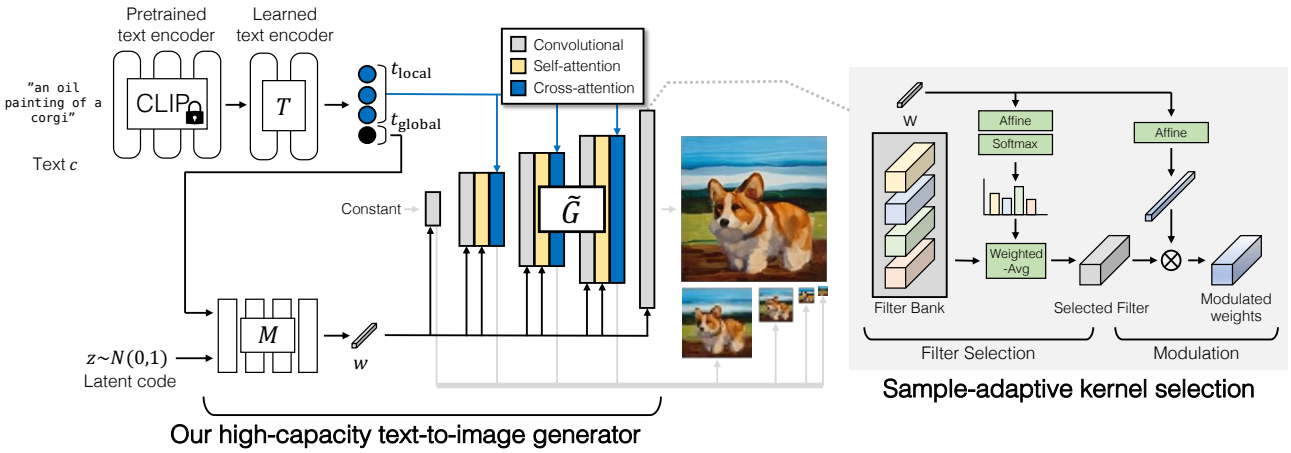


Figure 4. **Our GigaGAN high-capacity text-to-image generator.** First, we extract text embeddings using a pretrained CLIP model and a learned encoder T. The local text descriptors are fed to the generator using cross-attention. The global text descriptor, along with a latent code $\mathbf{z}$, is fed to a style mapping network M to produce style code $\mathbf{w}$. The style code modulates the main generator using our style-adaptive kernel selection, shown on the right. The generator outputs an image pyramid by converting the intermediate features into RGB images. To achieve higher capacity, we use multiple attention and convolution layers at each scale (Appendix A2). We also use a separate upsampler model, which is not shown in this diagram.

to judge the realism of the generated image, as compared to a sample from the training database $\mathcal{D}$, which contains image-text pairs.

Although GANs [6, 39, 41] can successfully generate realistic images on single- and multi-category datasets [13, 41, 100], open-ended text-conditioned synthesis on Internet images remains challenging. We hypothesize that the current limitation stems from its reliance on convolutional layers. That is, the same convolution filters are challenged to model the general image synthesis function for all text conditioning across all locations of the image. In this light, we seek to inject more expressivity into our parameterization by dynamically selecting convolution filters based on the input conditioning and by capturing long-range dependence via the attention mechanism.

Below, we discuss our key contributions to making ConvNets more expressive (Section 3.1), followed by our designs for the generator (Section 3.2) and discriminator (Section 3.3). Lastly, we introduce a new, fast GAN-based upsampler model that can improve the inference quality and speed of our method and diffusion models such as Imagen [80] and DALL·E 2 [74].

3.1. Modeling complex contextual interaction

Baseline StyleGAN generator. We base our architecture off the conditional version of StyleGAN2 [42], comprised of two networks $G = \widetilde{G} \circ M$. The mapping network $\mathbf{w} = M(\mathbf{z}, \mathbf{c})$ maps the inputs into a "style" vector $\mathbf{w}$, which modulates a series of upsampling convolutional layers in the synthesis network $\widetilde{G}(\mathbf{w})$ to map a learned constant tensor to an output image $\mathbf{x}$. Convolution is the main engine to generate all output pixels, with the $\mathbf{w}$ vector as the only source of information to model conditioning.

Sample-adaptive kernel selection. To handle the highly diverse distribution of internet images, we aim to increase the capacity of convolution kernels. However, increasing the width of the convolution layers becomes too demanding, as the same operation is repeated across all locations.

We propose an efficient way to enhance the expressivity of convolutional kernels by creating them on-the-fly based on the text conditioning, as illustrated in Figure 4 (right). In this scheme, we instantiate a bank of N filters $\{\mathbf{K}_i \in \mathbb{R}^{C_{\text{in}} \times C_{\text{out}} \times K \times K}\}_{i=1}^{N}$, instead of one, that takes a feature $\mathbf{f} \in \mathbb{R}^{C_{\text{in}}}$ at each layer. The style vector $\mathbf{w} \in \mathbb{R}^d$ then goes through an affine layer $[W_{\text{filter}}, b_{\text{filter}}] \in \mathbb{R}^{(d+1) \times N}$ to predict a set of weights to average across the filters, to produce an aggregated filter $\mathbf{K} \in \mathbb{R}^{C_{\text{in}} \times C_{\text{out}} \times K \times K}$.

$$\mathbf{K} = \sum_{i=1}^{N} \mathbf{K}_i \cdot \text{softmax}\left(W_{\text{filter}}^{\top} \mathbf{w} + b_{\text{filter}}\right)_i \qquad (1)$$

The filter is then used in the regular convolution pipeline of StyleGAN2, with the second affine layer $[W_{\text{mod}}, b_{\text{mod}}] \in \mathbb{R}^{(d+1) \times C_{\text{in}}}$ for weight (de-)modulation [42].

$$g_{\text{adaconv}}(\mathbf{f}, \mathbf{w}) = \left((W_{\text{mod}}^{\top} \mathbf{w} + b_{\text{mod}}) \otimes \mathbf{K}\right) * \mathbf{f}, \qquad (2)$$

where $\otimes$ and $*$ represent (de-)modulation and convolution.

At a high level, the softmax-based weighting can be viewed as a differentiable filter selection process based on input conditioning. Furthermore, since the filter selection process is performed only once at each layer, the selection process is much faster than the actual convolution, decoupling compute complexity from the resolution. Our method

shares a spirit with dynamic convolutions [23, 35, 91, 97] in that the convolution filters dynamically change per sample, but differs in that we explicitly instantiate a larger filter bank and select weights based on a separate pathway conditional on the $\mathbf{w}$-space of StyleGAN.

Interleaving attention with convolution. Since the convolutional filter operates within its receptive field, it cannot contextualize itself in relationship to distant parts of the images. One way to incorporate such long-range relationships is using attention layers $g_{\text{attention}}$. While recent diffusion-based models [15, 27, 79] have commonly adopted attention mechanisms, StyleGAN architectures are predominantly convolutional with the notable exceptions such as BigGAN [6], GANformer [30], and ViTGAN [50].

We aim to improve the performance of StyleGAN by integrating attention layers with the convolutional backbone. However, simply adding attention layers to StyleGAN often results in training collapse, possibly because the dot-product self-attention is not Lipschitz, as pointed out by Kim et al. [43]. As the Lipschitz continuity of discriminators has played a critical role in stable training [3, 22, 60], we use the L2-distance instead of the dot product as the attention logits to promote Lipschitz continuity [43], similar to ViTGAN [50].

To further improve performance, we find it crucial to match the architectural details of StyleGAN, such as equalized learning rate [39] and weight initialization from a unit normal distribution. We scale down the L2 distance logits to roughly match the unit normal distribution at initialization and reduce the residual gain from the attention layers. We further improve stability by tying the key and query matrix [50], and applying weight decay.

In the synthesis network $\widetilde{G}$, the attention layers are interleaved with each convolutional block, leveraging the style vector $\mathbf{w}$ as an additional token. At each attention block, we add a separate cross-attention mechanism $g_{\text{cross-attention}}$ to attend to individual word embeddings [4]. We use each input feature tensor as the query, and the text embeddings as the key and value of the attention mechanism.

3.2. Generator design

Text and latent-code conditioning. First, we extract the text embedding from the prompt. Previous works [75, 80] have shown that leveraging a strong language model is essential for producing strong results. To do so, we tokenize the input prompt (after padding it to $C = 77$ words, following best practices [75, 80]) to produce conditioning vector $\mathbf{c} \in \mathbb{R}^{C\times 1024}$, and take the features from the penultimate layer [80] of a frozen CLIP feature extractor [71]. To allow for additional flexibility, we apply additional attention layers T on top to process the word embeddings before passing them to the MLP-based mapping network. This results in text embedding $\mathbf{t} = T(\mathcal{E}_{\text{txt}}(\mathbf{c})) \in \mathbb{R}^{C\times 1024}$. Each component $\mathbf{t}_i$ of $\mathbf{t}$ captures the embedding of the i^{th} word in the sentence. We refer to them as $\mathbf{t}_{\text{local}} = \mathbf{t}_{\{1:C\}\setminus\text{EOT}} \in \mathbb{R}^{(C-1)\times 1024}$. The EOT ("end of text") component of $\mathbf{t}$ aggregates global information, and is called $\mathbf{t}_{\text{global}} \in \mathbb{R}^{1024}$. We process this global text descriptor, along with the latent code $\mathbf{z} \sim \mathcal{N}(0, 1)$, via an MLP mapping network to extract the style $\mathbf{w} = M(\mathbf{z}, \mathbf{t}_{\text{global}})$.

$$\begin{aligned}(\mathbf{t}_{\text{local}}, \mathbf{t}_{\text{global}}) &= T(\mathcal{E}_{\text{txt}}(\mathbf{c})),\\ \mathbf{w} &= M(\mathbf{z}, \mathbf{t}_{\text{global}}).\end{aligned} \tag{3}$$

Different from the original StyleGAN, we use both the text-based style code $\mathbf{w}$ to modulate the synthesis network $\widetilde{G}$ and the word embeddings $\mathbf{t}_{\text{local}}$ as features for cross-attention.

$$\mathbf{x} = \widetilde{G}(\mathbf{w}, \mathbf{t}_{\text{local}}). \tag{4}$$

Similar to earlier works [58, 74, 80], the text-image alignment visually improves with cross-attention.

Synthesis network. Our synthesis network consists of a series of upsampling convolutional layers, with each layer enhanced with the adaptive kernel selection (Equation 1) and followed by our attention layers.

$$\mathbf{f}_{\ell+1} = g^{\ell}_{\text{xa}}(g^{\ell}_{\text{attn}}(g^{\ell}_{\text{adaconv}}(\mathbf{f}_{\ell}, \mathbf{w}), \mathbf{w}), \mathbf{t}_{\text{local}}), \tag{5}$$

where g^{ℓ}_{xa}, g^{ℓ}_{attn}, and $g^{\ell}_{\text{adaconv}}$ denote the l-th layer of cross-attention, self-attention, and weight (de-)modulation layers. We find it beneficial to increase the depth of the network by adding more blocks at each layer. In addition, our generator outputs a multi-scale image pyramid with $L = 5$ levels, instead of a single image at the highest resolution, similar to MSG-GAN [38] and AnycostGAN [53]. We refer to the pyramid as $\{\mathbf{x}_i\}_{i=0}^{L-1} = \{\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_4\}$, with spatial resolutions $\{S_i\}_{i=0}^{L-1} = \{64, 32, 16, 8, 4\}$, respectively. The base level $\mathbf{x}_0$ is the output image $\mathbf{x}$. Each image of the pyramid is independently used to compute the GAN loss, as discussed in Section 3.3. We follow the findings of StyleGAN-XL [86] and turn off the style mixing and path length regularization [42]. We include more training details in Appendix A.1.

3.3. Discriminator design

As shown in Figure 5, our discriminator consists of separate branches for processing text with the function $\mathbf{t}_D$ and images with function ϕ. The prediction of real vs. fake is made by comparing the features from the two branches using function ψ. We introduce a new way of making predictions on multiple scales. Finally, we use additional CLIP and Vision-Aided GAN losses [44] to improve stability.

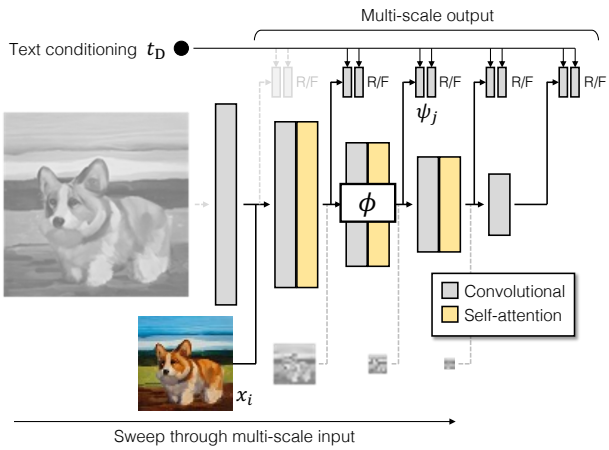


Figure 5. **Our discriminator** consists of two branches for processing the image and the text conditioning t_D. The text branch processes the text similar to the generator (Figure 4). The image branch receives an image pyramid and makes independent predictions for each image scale. Moreover, the predictions are made at all subsequent scales of the downsampling layers, making it a *multi-scale input, multi-scale output* (MS-I/O) discriminator.

Text conditioning. First, to incorporate conditioning into discriminators, we extract text descriptor $\mathbf{t}_\text{D}$ from text $\mathbf{c}$. Similar to the generator, we apply a pretrained text encoder, such as CLIP [71], followed by a few learnable attention layers. In this case, we only use the global descriptor.

Multiscale image processing. We observe that the early, low-resolution layers of the generator become inactive, using small dynamic ranges irrespective of the provided prompts. StyleGAN2 [42] also observes this phenomenon, concluding that the network relies on the high-resolution layers, as the model size increases. As recovering performance in low frequencies, which contains complex structure information, is crucial, we redesign the model architecture to provide training signals across multiple scales.

Recall the generator produces a pyramid $\{\mathbf{x}_i\}_{i=0}^{L-1}$, with the full image $\mathbf{x}_0$ at the pyramid base. MSG-GAN [38] improves performance by making a prediction on the entire pyramid at once, enforcing consistency across scales. However, in our large-scale setting, this harms stability, as this limits the generator from making adjustments to its initial low-res output.

Instead, we process each level of the pyramid *independently*. As shown in Figure 5, each level $\mathbf{x}_i$ makes real/fake a prediction at multiple scales $i < j \leq L$. For example, the full $\mathbf{x}_0$ makes predictions at $L = 5$ scales, the next level $\mathbf{x}_1$ makes predictions at 4 scales, and so on. In total, our discriminator produces $\frac{L(L-1)}{2}$ predictions, supervising multi-scale generations at multiple scales.

To extract features at different scales, we define feature extractor $\phi_{i\to j} : \mathbb{R}^{X_i \times X_i \times 3} \to \mathbb{R}^{X_j^D \times X_j^D \times C_j}$. Practically, each sub-network $\phi_{i\to j}$ is a subset of full $\phi \triangleq \phi_{0\to L}$, with $i > 0$ indicating late entry and $j < L$ indicating early exit. Each layer in ϕ is composed of self-attention, followed by convolution with stride 2. The final layer flattens the spatial extent into a 1×1 tensor. This produces output resolutions at $\{X_j^D\} = \{32, 16, 8, 4, 1\}$. This allows us to inject lower-resolution images on the pyramid into intermediate layers [39]. As we use a shared feature extractor across different levels and most of the added predictions are made at low resolutions, the increased computation overhead is manageable.

Multi-scale input, multi-scale output adversarial loss. In total, our training objective consists of discriminator losses, along with our proposed matching loss, to encourage the discriminator to take into account the conditioning:

$$\mathcal{V}_\text{MS-I/O}(G, D) = \sum_{i=0}^{L-1} \sum_{j=1}^{L} \mathcal{V}_\text{GAN}(G_i, D_{ij}) + \mathcal{V}_\text{match}(G_i, D_{ij}), \tag{6}$$

where $\mathcal{V}_\text{GAN}$ is the standard, non-saturating GAN loss [21]. To compute the discriminator output, we train predictor ψ, which uses text feature $\mathbf{t}_\text{D}$ to modulate image features $\phi(\mathbf{x})$:

$$D_{ij}(\mathbf{x}, \mathbf{c}) = \psi_j(\phi_{i\to j}(\mathbf{x}_i), \mathbf{t}_D) + \text{Conv}_{1\times 1}(\phi_{i\to j}(\mathbf{x}_i)), \tag{7}$$

where ψ_j is implemented as a 4-layer 1×1 modulated convolution, and $\text{Conv}_{1\times 1}$ is added as a skip connection to explicitly maintain an unconditional prediction branch [62].

Matching-aware loss. The previous GAN terms measure how closely the image $\mathbf{x}$ matches the conditioning $\mathbf{c}$, as well as how realistic $\mathbf{x}$ looks, irrespective of conditioning. However, during early training, when artifacts are obvious, the discriminator heavily relies on making a decision independent of conditioning and hesitates to account for the conditioning later.

To enforce the discriminator to incorporate conditioning, we match $\mathbf{x}$ with a random, independently sampled condition $\hat{\mathbf{c}}$, and present them as a fake pair:

$$\begin{aligned}\mathcal{V}_\text{match} = \mathbb{E}_{\mathbf{x},\mathbf{c},\hat{\mathbf{c}}} \big[&\log(1 + \exp(D(\mathbf{x}, \hat{\mathbf{c}}))) \\ &+ \log(1 + \exp(D(G(\mathbf{c}), \hat{\mathbf{c}}))) \big],\end{aligned} \tag{8}$$

where $(\mathbf{x}, c)$ and $\hat{\mathbf{c}}$ are separately sampled from p_data. This loss has previously been explored in text-to-image GAN works [76, 104], except we find that enforcing the Matching-aware loss on generated images from G, as well real images $\mathbf{x}$, leads to clear gains in performance (Table 1).

CLIP contrastive loss. We further leverage off-the-shelf pretrained models as a loss function [44, 84, 90]. In particular, we enforce the generator to produce outputs that are identifiable by the pre-trained CLIP image and text encoders [71], $\mathcal{E}_{\text{img}}$ and $\mathcal{E}_{\text{txt}}$, in the contrastive cross-entropy loss that was used to train them originally.

$$\mathcal{L}_{\text{CLIP}} = \mathbb{E}_{\{\mathbf{c}_n\}}\Big[-\log \frac{\exp(\mathcal{E}_{\text{img}}(G(\mathbf{c}_0))^\top \mathcal{E}_{\text{txt}}(\mathbf{c}_0))}{\sum_n \exp(\mathcal{E}_{\text{img}}(G(\mathbf{c}_0))^\top \mathcal{E}_{\text{txt}}(\mathbf{c}_n)}\Big), \tag{9}$$

where $\{\mathbf{c}_n\} = \{\mathbf{c}_0, \dots\}$ are sampled captions from the training data.

Vision-aided adversarial loss. Lastly, we build an additional discriminator that uses the CLIP model as a backbone, known as Vision-Aided GAN [44]. We freeze the CLIP image encoder, extract features from the intermediate layers, and process them through a simple network with 3×3 conv layers to make real/fake predictions. We also incorporate conditioning through modulation, as in Equation 7. To stabilize training, we also add a fixed random projection layer, as proposed by Projected GAN [84]. We refer to this as $\mathcal{L}_{\text{Vision}}(G)$ (omitting the learnable discriminator parameters for clarity).

Our final objective is $\mathcal{V}(G, D) = \mathcal{V}_{\text{MS-I/O}}(G, D) + \mathcal{L}_{\text{CLIP}}(G) + \mathcal{L}_{\text{Vision}}(G)$, with weighting between the terms specified in Table A2.

3.4. GAN-based upsampler

Furthermore, GigaGAN framework can be easily extended to train a text-conditioned superresolution model, capable of upsampling the outputs of the base GigaGAN generator to obtain high-resolution images at 512px or 2k resolution. By training our pipeline in two separate stages, we can afford a higher capacity 64px base model within the same computational resources.

In the upsampler, the synthesis network is rearranged to an asymmetric U-Net architecture, which processes the 64px input through 3 downsampling residual blocks, followed by 6 upsampling residual blocks with attention layers to produce the 512px image. There exist skip connections at the same resolution, similar to CoModGAN [106]. The model is trained with the same losses as the base model, as well as the LPIPS Perceptual Loss [105] with respect to the ground truth high-resolution image. Vision-aided GAN is not used for the upsampler. During training and inference time, we apply moderate Gaussian noise augmentation to reduce the gap between real and GAN-generated images. Please refer to Appendix A.3 for more details.

Our GigaGAN framework becomes particularly effective for the superresolution task compared to the diffusion-based models, which cannot afford as many sampling steps as the base model at high resolution. The LPIPS regression loss also provides a stable learning signal. We believe that our GAN upsampler can serve as a drop-in replacement for the superresolution stage of other generative models.

4. Experiments

Systematic, controlled evaluation of large-scale text-to-image synthesis tasks is difficult, as most existing models are not publicly available. Training a new model from scratch would be prohibitively costly, even if the training code were available. Still, we compare our model to recent text-to-image models, such as Imagen [80], Latent Diffusion Models (LDM) [79], Stable Diffusion [78], and Parti [101], based on the available information, while acknowledging considerable differences in the training dataset, number of iterations, batch size, and model size. In addition to text-to-image results, we evaluate our model on ImageNet class-conditional generation in Appendix B, for an apples-to-apples comparison with other methods at a more controlled setting.

For quantitative evaluation, we mainly use the Fréchet Inception Distance (FID) [25] for measuring the realism of the output distribution and the CLIP score for evaluating the image-text alignment.

We conduct five different experiments. First, we show the effectiveness of our method by gradually incorporating each technical component one by one (Section 4.2). Second, our text-to-image synthesis results demonstrate that GigaGAN exhibits comparable FID with Stable Diffusion (SD-v1.5) [79] while generating results hundreds of times faster than diffusion or autoregressive models (Section 4.3). Third, we compare GigaGAN with a distillation-based diffusion model [59] and show that GigaGAN can synthesize higher-quality images faster than the distillation-based diffusion model. Fourth, we verify the advantage of GigaGAN's upsampler over other upsamplers in both conditional and unconditional super-resolution tasks. Lastly, we show our large-scale GANs still enjoy the continuous and disentangled latent space manipulation of GANs, enabling new image editing modes (Section 4.6).

4.1. Training and evaluation details

We implement GigaGAN based on the StudioGAN PyTorch library [37], following the standard FID evaluation protocol with the anti-aliasing bicubic resize function [67], unless otherwise noted. For text-to-image synthesis, we train our models on the union of LAION2B-en [88] and COYO-700M [8] datasets, with the exception of the 128-to-1024 upsampler model trained on Adobe's internal Stock images. The image-text pairs are preprocessed based on CLIP score [24], image resolution, and aesthetic score [87], similar to prior work [78]. We use CLIP ViT-L/14 [71] for the pre-trained text encoder and OpenCLIP ViT-G/14 [32] for CLIP score calculation [24] except for Table 1. All our

Figure 6. **Style mixing**. Our GAN-based architecture retains a disentangled latent space, enabling us to blend the coarse style of one sample with the fine style of another. All outputs are generated with the prompt "A Toy sport sedan, CG art." The corresponding latent codes are spliced together to produce a style-swapping grid.

Figure 7. **Prompt interpolation**. GigaGAN enables smooth interpolation between prompts, as shown in the interpolation grid. The four corners are generated from the same latent $\mathbf{z}$ but with different text prompts. The corresponding text embeddings $\mathbf{t}$ and style vectors $\mathbf{w}$ are interpolated to create a smooth transition. The same $\mathbf{z}$ results in similar layouts. See Figure 8 for more precise control.

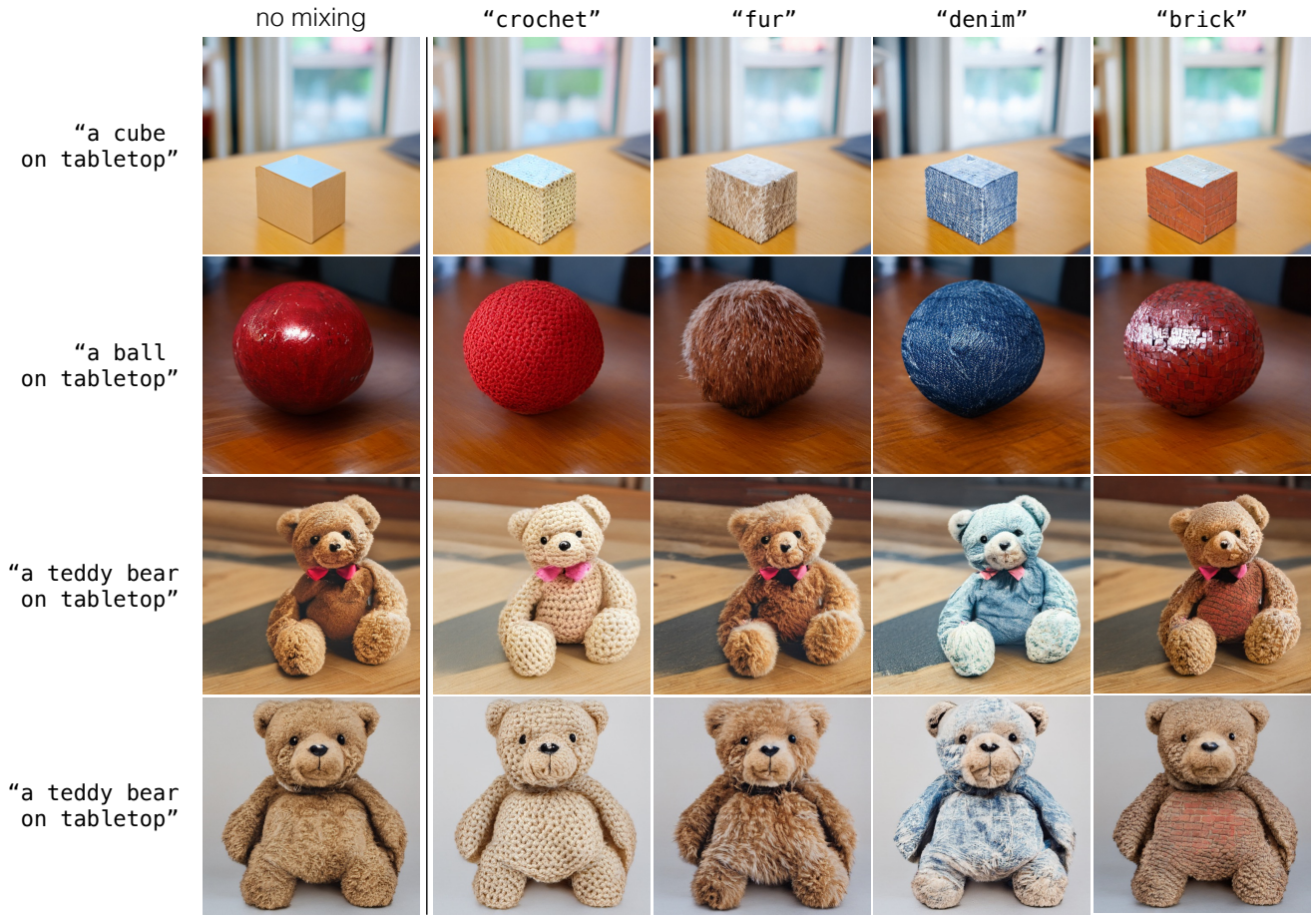


Figure 8. **Prompt mixing**. GigaGAN retains a disentangled latent space, enabling us to combine the coarse style of one sample with the fine style of another. Moreover, GigaGAN can directly control the style with text prompts. Here we generate four outputs using the prompts “a X on tabletop”, shown in the “no mixing” column. Then we re-compute the text embeddings $\mathbf{t}$ and the style codes $\mathbf{w}$ using the new prompts “a X with the texture of Y on tabletop”, such as “a cube with the texture of crochet on tabletop”, and apply them to the second half layers of the generator, achieving layout-preserving fine style control. Cross-attention mechanism automatically localizes the style to the object of interest.

models are trained and evaluated on A100 GPUs. We include more training and evaluation details in Appendix A.

4.2. Effectiveness of proposed components

First, we show the effectiveness of our formulation via ablation study in Table 1. We set up a baseline by adding text-conditioning to StyleGAN2 and tuning the configuration based on the findings of StyleGAN-XL. We first directly increase the model size of this baseline, but we find that this does not improve the FID and CLIP scores. Then, we add our components one by one and observe that they consistently improve performance. In particular, our model is more scalable, as the higher-capacity version of the final formulation achieves better performance.

4.3. Text-to-Image synthesis

We proceed to train a larger model by increasing the capacity of the base generator and upsampler to 652.5M and 359.1M, respectively. This results in an unprecedented size of GAN model, with a total parameter count of 1.0B. Table 2 compares the performance of our end-to-end pipeline to various text-to-image generative models [5, 10, 63, 74, 75, 78–80, 101, 108]. Note that there exist differences in the training dataset, the pretrained text encoders, and even image resolutions. For example, GigaGAN initially synthesizes 512px images, which are resized to 256px before evaluation.

Table 2 shows that GigaGAN exhibits a lower FID than DALL·E 2 [74], Stable Diffusion [78], and Parti-750M [101]. While our model can be optimized to better match the feature distribution of real images than existing

Table 1. **Ablation study on 64px text-to-image synthesis.** To evaluate the effectiveness of our components, we start with a modified version of StyleGAN for text conditioning. While increasing the network width does not show satisfactory improvement, each addition of our contributions keeps improving metrics. Finally, we increase the network width and scale up training to reach our final model. All ablated models are trained for 100k iterations at a batch size of 256 except for the Scale-up row (1350k iterations with a larger batch size). CLIP Score is computed using CLIP ViT-B/32 [71].

Model	FID-10k ↓	CLIP Score ↑	# Param.
StyleGAN2	29.91	0.222	27.8M
+ Larger (5.7×)	34.07	0.223	158.9M
+ Tuned	28.11	0.228	26.2M
+ Attention	23.87	0.235	59.0M
+ Matching-aware D	27.29	0.250	59.0M
+ Matching-aware G and D	21.66	0.254	59.0M
+ Adaptive convolution	19.97	0.261	80.2M
+ Deeper	19.18	0.263	161.9M
+ CLIP loss	14.88	0.280	161.9M
+ Multi-scale training	14.92	0.300	164.0M
+ Vision-aided GAN	13.67	0.287	164.0M
+ Scale-up (**GigaGAN**)	9.18	0.307	652.5M

models, the quality of the generated images is not necessarily better (see Appendix C for more samples). We acknowledge that this may represent a corner case of zero-shot FID on COCO2014 dataset and suggest that further research on a better evaluation metric is necessary to improve text-to-image models. Nonetheless, we emphasize that GigaGAN is the first GAN model capable of synthesizing promising images from arbitrary text prompts and exhibits competitive zero-shot FID with other text-to-image models.

Table 2. **Comparison to recent text-to-image models**. Model size, GPU days, total images seen during training, COCO FID-30k, and inference speed of text-image models. $*$ denotes that the model has been evaluated by us. GigaGAN achieves a lower FID than DALL·E 2 [74], Stable Diffusion [78], and Parti-750M [101], while being much faster compared to recent competitive methods.

	Model	Type	# Param.	# Images	FID-30k ↓	Inf. time
256	DALL·E [75]	Diff	12.0B	1.54B	27.50	-
	GLIDE [63]	Diff	5.0B	5.94B	12.24	15.0s
	LDM [79]	Diff	1.5B	0.27B	12.63	9.4s
	DALL·E 2 [74]	Diff	5.5B	5.63B	10.39	-
	Imagen [80]	Diff	3.0B	15.36B	7.27	9.1s
	eDiff-I [5]	Diff	9.1B	11.47B	6.95	32.0s
	Parti-750M [101]	AR	750M	3.69B	10.71	-
	Parti-3B [101]	AR	3.0B	3.69B	8.10	6.4s
	Parti-20B [101]	AR	20.0B	3.69B	7.23	-
	LAFITE [108]	GAN	75M	-	26.94	0.02s
512	SD-v1.5* [78]	Diff	0.9B	3.16B	9.62	2.9s
	Muse-3B [10]	AR	3.0B	0.51B	7.88	1.3s
	GigaGAN	GAN	1.0B	0.98B	9.09	0.13s

Table 3. **Comparison to distilled diffusion models** shows that GigaGAN achieves better FID and CLIP scores compared to the progressively distilled diffusion models [59] for fast inference. As GigaGAN generates outputs in a single feedforward pass, the inference speed is still faster. The evaluation setup is different from Table 2 to match SD-distilled's protocol [59].

Model	Steps	FID-5k ↓	CLIP ↑	Inf. time
SD-distilled-2 [59]	2	37.3	0.27	0.23s
SD-distilled-4 [59]	4	26.0	0.30	0.33s
SD-distilled-8 [59]	8	26.9	0.30	0.52s
SD-distilled-16 [59]	16	28.8	0.30	0.88s
GigaGAN	1	21.1	0.32	0.13s

Table 4. **Text-conditioned 128→1024 super-resolution** on random 10K LAION samples, compared against unconditional Real-ESRGAN [33] and Stable Diffusion Upscaler [78]. GigaGAN enjoys the fast speed of a GAN-based model while achieving better FID, patch-FID [9], CLIP score, and LPIPS [105].

Model	# Param.	Inf. time	FID-10k ↓	pFID ↓	CLIP ↑	LPIPS↓
Real-ESRGAN [33]	17M	0.06s	8.60	22.8	0.314	0.363
SD Upscaler [78]	846M	7.75s	9.39	41.3	0.316	0.523
GigaGAN	693M	0.13s	1.54	8.90	0.322	0.274

Table 5. **Unconditional 64→256 super-resolution** on ImageNet. We compare to a simple U-Net trained with a pixel regression loss (U-Net regression), and diffusion-based methods (SR3 [81] and LDM [79]. Our method achieves higher realism scores represented by the Inception Score (IS) and FID.

Model	# Param.	Steps	IS ↑	FID-50k ↓	PSNR ↑	SSIM ↑
U-Net regression [81]	625M	1	121.1	15.2	27.9	0.80
SR3 [81]	625M	100	180.1	5.2	26.4	0.76
LDM-4 [79]	169M	100	166.3	2.8	24.4	0.69
emphLDM-4 [79]	552M	100	174.9	2.4	24.7	0.71
LDM-4-G [79]	183M	50	153.7	4.4	25.8	0.74
GigaGAN	359M	1	191.5	1.2	24.3	0.71

4.4. Comparison with distilled diffusion models

While GigaGAN is at least 20 times faster than the above diffusion models, there have been efforts to improve the inference speed of diffusion models. We compare GigaGAN with progressively distilled Stable Diffusion (SD-distilled) [59]. Table 3 demonstrates that GigaGAN remains faster than the distilled Stable Diffusion while showing better FID and CLIP scores of 21.1 and 0.32, respectively. We follow the evaluation protocol of SD-distilled [59] and report FID and CLIP scores on COCO2017 dataset [54], where images are resized to 512px.

4.5. Super-resolution for large-scale image synthesis

We separately evaluate the performance of the GigaGAN upsampler. Our evaluation consists of two parts. First, we compare GigaGAN with several commonly-used upsamplers. For the text-conditioned upsampling task, we combine the Stable Diffusion [78] 4x Upscaler and 2x Latent Upscaler to establish an 8x upscaling model (SD Up-

Figure 9. **Failure cases.** Our outputs with the same prompts as DALL·E 2. Each column conditions on "a teddy bear on a skateboard in Times Square", "a Vibrant portrait painting of Salvador Dali with a robotic half face", and "A close up of a handpalm with leaves growing from it". Compared to production-grade models such as DALL·E 2, our model exhibits limitations in realism and compositionality. See Appendix C for uncurated comparisons.

scaler). We also use the unconditional Real-ESRGAN [33] as another baseline. Table 4 measures the performance of the upsampler on random 10K images from the LAION dataset and shows that our GigaGAN upsampler significantly outperforms the other upsamplers in realism scores (FID and patch-FID [9]), text alignment (CLIP score) and closeness to the ground truth (LPIPS [105]). In addition, for more controlled comparison, we train our model on the ImageNet *unconditional* superresolution task and compare performance with the diffusion-based models, including SR3 [81] and LDM [79]. As shown in Table 5, GigaGAN achieves the best IS and FID scores with a single feedforward pass.

4.6. Controllable image synthesis

StyleGANs are known to possess a linear latent space useful for image manipulation, called the $\mathcal{W}$-space. Likewise, we perform coarse and fine-grained style swapping using style vectors $\mathbf{w}$. Similar to the $\mathcal{W}$-space of StyleGAN, Figure 6 illustrates that GigaGAN maintains a disentangled $\mathcal{W}$-space, suggesting existing latent manipulation techniques of StyleGAN can transfer to GigaGAN. Furthermore, our model possesses another latent space of text embedding $\mathbf{t} = [\mathbf{t}_{\text{local}}, \mathbf{t}_{\text{global}}]$ prior to $\mathcal{W}$, and we explore its potential for image synthesis. In Figure 8, we show that the disentangled style manipulation can be controlled via text inputs. In detail, we can compute the text embedding $\mathbf{t}$ and style code $\mathbf{w}$ using different prompts and apply them to different layers of the generator. This way, we gain not only the coarse and fine style disentanglement but also an intuitive prompt-based maneuver in the style space.

5. Discussion and Limitations

Our experiments provide a conclusive answer about the scalability of GANs: our new architecture can scale up to model sizes that enable text-to-image synthesis. However, the visual quality of our results is not yet comparable to production-grade models like DALL·E 2. Figure 9 shows several instances where our method fails to produce high-quality results when compared to DALL·E 2, in terms of photorealism and text-to-image alignment for the same input prompts used in their paper.

Nevertheless, we have tested capacities well beyond what is possible with a naïve approach and achieved competitive visual quality with autoregressive and diffusion models trained with similar resources while being orders of magnitude faster and enabling latent interpolation and stylization. Our GigaGAN architecture opens up a whole new design space for large-scale generative models and brings back key editing capabilities that became challenging with the transition to autoregressive and diffusion models. We expect our performance to improve with larger models, as seen in Table 1.

Acknowledgments. We thank Simon Niklaus, Alexandru Chiculita, and Markus Woodson for building the distributed training pipeline. We thank Nupur Kumari, Gaurav Parmar, Bill Peebles, Phillip Isola, Alyosha Efros, and Joonghyuk Shin for their helpful comments. We also want to thank Chenlin Meng, Chitwan Saharia, and Jiahui Yu for answering many questions about their fantastic work. We thank Kevin Duarte for discussions regarding upsampling beyond 4K. Part of this work was done while Minguk Kang was an intern at Adobe Research. Minguk Kang and Jaesik Park were supported by IITP grant funded by the government of South Korea (MSIT) (POSTECH GSAI: 2019-0-01906 and Image restoration: 2021-0-00537).

References

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 5

[2] Saeed Anwar and Nick Barnes. Densely residual laplacian super-resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 5

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein Generative Adversarial Networks. In *International Conference on Machine Learning (ICML)*, 2017. 7

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations (ICLR)*, 2015. 7

[5] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, et al. ediffi: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022. 11, 12

[6] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *International Conference on Learning Representations (ICLR)*, 2019. 1, 5, 6, 7, 19, 23

[7] Andrew Brock, Theodore Lim, James M. Ritchie, and Nick Weston. Neural Photo Editing with Introspective Adversarial Networks. In *International Conference on Learning Representations (ICLR)*, 2017. 5

[8] Minwoo Byeon, Beomhee Park, Haecheon Kim, Sungjun Lee, Woonhyuk Baek, and Saehoon Kim. COYO-700M: Image-Text Pair Dataset. https://github.com/kakaobrain/coyo-dataset, 2022. 5, 9, 19

[9] Lucy Chai, Michael Gharbi, Eli Shechtman, Phillip Isola, and Richard Zhang. Any-resolution training for high-resolution image synthesis. In *European Conference on Computer Vision (ECCV)*, 2022. 12, 13

[10] Huiwen Chang, Han Zhang, Jarred Barber, AJ Maschinot, Jose Lezama, Lu Jiang, Ming-Hsuan Yang, Kevin Murphy, William T Freeman, Michael Rubinstein, et al. Muse: Text-to-image generation via masked generative transformers. *arXiv preprint arXiv:2301.00704*, 2023. 11, 12

[11] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11315–11325, 2022. 19

[12] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International Conference on Machine Learning (ICML)*. PMLR, 2020. 5

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 6, 19

[14] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. *Conference on Neural Information Processing Systems (NeurIPS)*, 28, 2015. 1

[15] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021. 5, 7, 19, 21, 22, 23

[16] Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, et al. Cogview: Mastering text-to-image generation via transformers. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021. 5

[17] Ming Ding, Wendi Zheng, Wenyi Hong, and Jie Tang. Cogview2: Faster and better text-to-image generation via hierarchical transformers. *arXiv preprint arXiv:2204.14217*, 2022. 5

[18] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2015. 5

[19] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12873–12883, 2021. 19

[20] Oran Gafni, Adam Polyak, Oron Ashual, Shelly Sheynin, Devi Parikh, and Yaniv Taigman. Make-A-Scene: Scene-Based Text-to-Image Generation with Human Priors. In *European Conference on Computer Vision (ECCV)*, 2022. 5

[21] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 2672–2680, 2014. 1, 5, 8

[22] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2017. 7

[23] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. In *International Conference on Learning Representations (ICLR)*, 2017. 6

[24] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021. 9, 19

[25] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 6626–6637, 2017. 9, 19

[26] Jonathan Ho, Ajay Jain, and P. Abbeel. Denoising Diffusion Probabilistic Models. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 5

[27] Jonathan Ho, Chitwan Saharia, William Chan, David J. Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded Diffusion Models for High Fidelity Image Generation. *Journal of Machine Learning Research*, pages 47:1–47:33, 2022. 7, 19

[28] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *Conference on Neural Information Processing Systems (NeurIPS) Workshop*, 2022. 23

[29] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *European Conference on Computer Vision (ECCV)*, 2018. 5

[30] Drew A Hudson and Larry Zitnick. Generative adversarial transformers. In *International Conference on Machine Learning (ICML)*, 2021. 7

[31] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. GANSpace: Discovering Interpretable GAN Controls. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 5

[32] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip. *https://doi.org/10.5281/zenodo.5143773*, 2021. 9

[33] intao Wang and Liangbin Xie and Chao Dong and Ying Shan. Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In *IEEE International Conference on Computer Vision (ICCV) Workshop*, 2021. 3, 4, 12, 33, 34, 35

[34] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5

[35] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. *Conference on Neural Information Processing Systems (NeurIPS)*, 29, 2016. 6

[36] Minguk Kang, Woohyeon Shim, Minsu Cho, and Jaesik Park. Rebooting ACGAN: Auxiliary Classifier GANs with Stable Training. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021. 5

[37] Minguk Kang, Joonghyuk Shin, and Jaesik Park. StudioGAN: A Taxonomy and Benchmark of GANs for Image Synthesis. *arXiv preprint arXiv:2206.09479*, 2022. 9, 19

[38] Animesh Karnewar and Oliver Wang. Msg-gan: Multi-scale gradients for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7799–7808, 2020. 7, 8

[39] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning Representations (ICLR)*, 2018. 5, 6, 7, 8

[40] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021. 5

[41] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4401–4410, 2019. 1, 5, 6, 19, 23

[42] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8110–8119, 2020. 1, 5, 6, 7, 8

[43] Hyunjik Kim, George Papamakarios, and Andriy Mnih. The lipschitz constant of self-attention. In *International Conference on Machine Learning (ICML)*, 2021. 7

[44] Nupur Kumari, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Ensembling off-the-shelf models for gan training. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 5, 7, 8, 9

[45] Tuomas Kynkäänniemi, Tero Karras, Miika Aittala, Timo Aila, and Jaakko Lehtinen. The Role of ImageNet Classes in Fr\'echet Inception Distance. *arXiv preprint arXiv:2203.06026*, 2022. 19

[46] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved Precision and Recall Metric for Assessing Generative Models. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 19

[47] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5

[48] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive Image Generation using Residual Quantization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11523–11532, 2022. 19

[49] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *European Conference on Computer Vision (ECCV)*, 2018. 5

[50] Kwonjoon Lee, Huiwen Chang, Lu Jiang, Han Zhang, Zhuowen Tu, and Ce Liu. ViTGAN: Training GANs with vision transformers. In *International Conference on Learning Representations (ICLR)*, 2022. 7

[51] Muyang Li, Ji Lin, Chenlin Meng, Stefano Ermon, Song Han, and Jun-Yan Zhu. Efficient spatially sparse inference for conditional gans and diffusion models. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022. 5

[52] Jiadong Liang, Wenjie Pei, and Feng Lu. Cpgan: Content-parsing generative adversarial networks for text-to-image synthesis. In *European Conference on Computer Vision (ECCV)*, 2020. 5

[53] Ji Lin, Richard Zhang, Frieder Ganz, Song Han, and Jun-Yan Zhu. Anycost gans for interactive image synthesis and editing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14986–14996, 2021. 7

[54] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014. 5, 12, 19

[55] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo Numerical Methods for Diffusion Models on Manifolds. In *International Conference on Learning Representations (ICLR)*, 2022. 27, 28, 29, 30, 31, 32

[56] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations (ICLR)*, 2019. 20

[57] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *arXiv preprint arXiv:2206.00927*, 2022. 5

[58] Elman Mansimov, Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Generating Images from Captions with Attention. In *International Conference on Learning Representations (ICLR)*, 2016. 5, 7

[59] Chenlin Meng, Ruiqi Gao, Diederik P Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Conference on Neural Information Processing Systems (NeurIPS) Workshop*, 2022. 5, 9, 12

[60] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International Conference on Machine Learning (ICML)*, 2018. 7

[61] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Which Training Methods for GANs do actually Converge? In *International Conference on Machine Learning (ICML)*, 2018. 20

[62] Takeru Miyato and Masanori Koyama. cGANs with Projection Discriminator. In *International Conference on Learning Representations (ICLR)*, 2018. 8, 20

[63] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. In *International Conference on Machine Learning (ICML)*, 2022. 5, 11, 12

[64] OpenAI. DALL·E API. https://openai.com/product/dall-e-2, 2022. 27, 28, 29, 30, 31, 32

[65] Taesung Park, Alexei A. Efros, Richard Zhang, and Jun-Yan Zhu. Contrastive Learning for Unpaired Image-to-Image Translation. In *European Conference on Computer Vision (ECCV)*, 2020. 5

[66] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 5

[67] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On Aliased Resizing and Surprising Subtleties in GAN Evaluation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 9, 19

[68] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 8024–8035, 2019. 19

[69] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 5

[70] William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. *arXiv preprint arXiv:2212.09748*, 2022. 19

[71] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, 2021. 5, 7, 8, 9, 12, 20

[72] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 1, 5

[73] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 2020. 5

[74] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 1, 5, 6, 7, 11, 12, 19, 23, 27, 28, 29, 30, 31, 32

[75] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning (ICML)*, 2021. 5, 7, 11, 12

[76] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International Conference on Machine Learning (ICML)*, 2016. 5, 8

[77] Scott E Reed, Zeynep Akata, Santosh Mohan, Samuel Tenka, Bernt Schiele, and Honglak Lee. Learning what and where to draw. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2016. 5

[78] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. Stable Diffusion. https://github.com/CompVis/stable-diffusion. Accessed: 2022-11-06. 3, 4, 9, 11, 12, 19, 23, 27, 28, 29, 30, 31, 32, 33, 34, 35

[79] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 5, 7, 9, 11, 12, 13, 19, 21, 22, 23, 27, 28, 29, 30, 31, 32

[80] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. *arXiv preprint arXiv:2205.11487*, 2022. 1, 5, 6, 7, 9, 11, 12, 19

[81] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2022. 12, 13, 19

[82] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved Techniques for Training GANs. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 2234–2242, 2016. 19

[83] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations (ICLR)*, 2022. 5

[84] Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected GANs Converge Faster. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021. 5, 8, 9

[85] Axel Sauer, Tero Karras, Samuli Laine, Andreas Geiger, and Timo Aila. StyleGAN-T: Unlocking the Power of GANs for Fast Large-Scale Text-to-Image Synthesis. *arXiv preprint arXiv:2301.09515*, 2023. 5

[86] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–10, 2022. 5, 7, 19, 21, 22

[87] Christoph Schuhmann. CLIP+MLP Aesthetic Score Predictor. https://github.com/christophschuhmann/improved-aesthetic-predictor. 9, 19

[88] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. LAION-5B: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022. 1, 5, 9, 19

[89] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations (ICLR)*, 2021. 5

[90] Diana Sungatullina, Egor Zakharov, Dmitry Ulyanov, and Victor Lempitsky. Image manipulation with perceptual discriminators. In *European Conference on Computer Vision (ECCV)*, 2018. 8

[91] Md Mehrab Tanjim. DynamicRec: a dynamic convolutional network for next item recommendation. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM)*, 2020. 6

[92] Ming Tao, Bing-Kun Bao, Hao Tang, and Changsheng Xu. GALIP: Generative Adversarial CLIPs for Text-to-Image Synthesis. *arXiv preprint arXiv:2301.12959*, 2023. 5

[93] Ming Tao, Hao Tang, Fei Wu, Xiao-Yuan Jing, Bing-Kun Bao, and Changsheng Xu. DF-GAN: A Simple and Effective Baseline for Text-to-Image Synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 5

[94] Ken Turkowski. Filters for common resampling tasks. *Graphics gems*, pages 147–165, 1990. 19

[95] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *European Conference on Computer Vision (ECCV) Workshop*, 2018. 5

[96] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical report, California Institute of Technology, 2010. 5

[97] Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. Pay Less Attention with Lightweight and Dynamic Convolutions. In *International Conference on Learning Representations (ICLR)*, 2018. 6

[98] Jonas Wulff and Antonio Torralba. Improving inversion and generation diversity in stylegan using a gaussianized latent space. *arXiv preprint arXiv:2009.06529*, 2020. 5

[99] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 5

[100] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 6

[101] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2022. 1, 5, 9, 11, 12

[102] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-Attention Generative Adversarial Networks. In *International Conference on Machine Learning (ICML)*, pages 7354–7363, 2019. 5

[103] Han Zhang, Jing Yu Koh, Jason Baldridge, Honglak Lee, and Yinfei Yang. Cross-Modal Contrastive Learning for Text-to-Image Generation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 5

[104] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 1, 5, 8

[105] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 9, 12, 13, 20

[106] Shengyu Zhao, Jonathan Cui, Yilun Sheng, Yue Dong, Xiao Liang, Eric I Chang, and Yan Xu. Large Scale Image Completion via Co-Modulated Generative Adversarial Networks. In *International Conference on Learning Representations (ICLR)*, 2021. 9

[107] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. *arXiv preprint arXiv 2006.10738*, 2020. 5

[108] Yufan Zhou, Ruiyi Zhang, Changyou Chen, Chunyuan Li, Chris Tensmeyer, Tong Yu, Jiuxiang Gu, Jinhui Xu, and

Tong Sun. Lafite: Towards language-free training for text-to-image generation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 11, 12

[109] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision (ECCV)*, 2016. 5

[110] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2223–2232, 2017. 5

[111] Minfeng Zhu, Pingbo Pan, Wei Chen, and Yi Yang. Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 5

[112] Peihao Zhu, Rameen Abdal, Yipeng Qin, John Femiani, and Peter Wonka. Improved stylegan embedding: Where are the good latents? *arXiv preprint arXiv:2012.09036*, 2020. 5

Appendices

We first provide training and evaluation details in Appendix A. Then, we share results on ImageNet, with visual comparison to existing methods in Appendix B. Lastly in Appendix C, we show more visuals on our text-to-image synthesis results and compare them with LDM [79], Stable Diffusion [78], and DALL·E 2 [74].

A. Training and evaluation details

A.1. Text-to-image synthesis

We train GigaGAN on a combined dataset of LAION2B-en [88] and COYO-700M [8] in PyTorch framework [68]. For training, we apply center cropping, which results in a square image whose length is the same as the shorter side of the original image. Then, we resize the image to the resolution 64×64 using PIL.LANCZOS [94] resizer, which supports anti-aliasing [67]. We filter the training image–text pairs based on image resolution (≥ 512), CLIP score (> 0.3) [24], aesthetics score (> 5.0) [87], and remove watermarked images. We train our GigaGAN based on the configurations denoted in the fourth and fifth columns of Table A2.

For evaluation, we use 40,504 and 30,000 real and generated images from COCO2014 [54] validation dataset as described in Imagen [80]. We apply the center cropping and resize the real and generated images to 299×299 resolution using PIL.BICUBIC, suggested by clean-fid [67]. We use the clean-fid library [67] for FID calculation.

A.2. Conditional image synthesis on ImageNet

We follow the training and evaluation protocol proposed by Kang *et al*. [37] to make a fair comparison against other cutting-edge generative models. We use the same cropping strategy to process images for training and evaluation as in our text-to-image experiments. Then, we resize the image to the target resolution (64×64 for the base generator or 256×256 for the super-resolution stack) using PIL.LANCZOS [94] resizer, which supports anti-aliasing [67]. Using the pre-processed training images, we train GigaGAN based on the configurations denoted in the second and third columns of Table A2.

For evaluation, we upsample the real and generated images to 299×299 resolution using the PIL.BILINEAR resizer. To compute FID, we generate 50k images without truncation tricks [6, 41] and compare those images with the entire training dataset. We use the pre-calculated features of real images provided by StudioGAN [37] and 50k generated images for Precision & Recall [46] calculation.

A.3. Super-resolution results

For model training, we preprocess ImageNet in the same way as in Section A.2 and use the configuration in the last column of Table A2. To compare our model with SR3 [81] and LDM fairly, we follow the evaluation procedure described in SR3 and LDM papers.

B. ImageNet experiments

B.1. Qualitative results

We train a class-conditional GAN on the ImageNet dataset [13], for which apples-to-apples comparison is possible using the same dataset and evaluation pipeline. Our GAN achieves comparable generation quality to the cutting-edge generative models without a pretrained ImageNet classifier, which acts favorably toward automated metrics [45]. We apply L2 self-attention, style-adaptive convolution kernel, and matching-aware loss to our model and use a wider synthesis network to train the base 64px model with a batch size of 1024. Additionally, we train a separate 256px class-conditional upsampler model and combine them with an end-to-end finetuning stage. Table A1 shows that our method generates high-fidelity images.

Table A1. **Class-conditional synthesis on ImageNet 256px.** Our method performs competitively against large diffusion and transformer models. Shaded methods leverage a pretrained ImageNet classifier at training or inference time, which could act favorably toward the automated metrics [45]. † indicates IS [82] and FID [25] are borrowed from the original DiT paper [70].

	Model	IS [82]	FID [25]	Precision/Recall [46]	Size
GAN	BigGAN-Deep [6]	224.46	6.95	0.89/0.38	112M
	StyleGAN-XL [86]	297.62	2.32	0.82/0.61	166M
Diffusion	ADM-G [15]	207.86	4.48	0.84/0.62	608M
	ADM-G-U [15]	240.24	4.01	0.85/0.62	726M
	CDM [27]	158.71	4.88	- / -	-
	LDM-8-G [79]	209.52	7.76	- / -	506M
	LDM-4-G [79]	247.67	3.60	- / -	400M
	DiT-XL/2† [70]	278.24	2.27	- / -	675M
xformer	Mask-GIT [11]	216.38	5.40	0.87/0.60	227M
	VQ-GAN [19]	314.61	5.20	0.81/0.57	1.4B
	RQ-Transformer [48]	339.41	3.83	0.85/0.60	3.8B
	GigaGAN	225.52	3.45	0.84/0.61	569M

B.2. Quantitative results

We provide visual results from ADM-G-U, LDM, StyleGAN-XL [86], and GigaGAN in Figures A1 and A2. Although StyleGAN-XL has the lowest FID, its visual quality appears worse than ADM and GigaGAN. StyleGAN-XL struggles to synthesize the overall image structure, leading to less realistic images. In contrast, GigaGAN appears to synthesize the overall structure better than StyleGAN-XL and faithfully captures fine-grained details, such as the wing patterns of a monarch and the white fur of an arctic fox. Compared to GigaGAN, ADM-G-U synthesizes the image structure more rationally but lacks in reflecting the aforementioned fine-grained details.

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

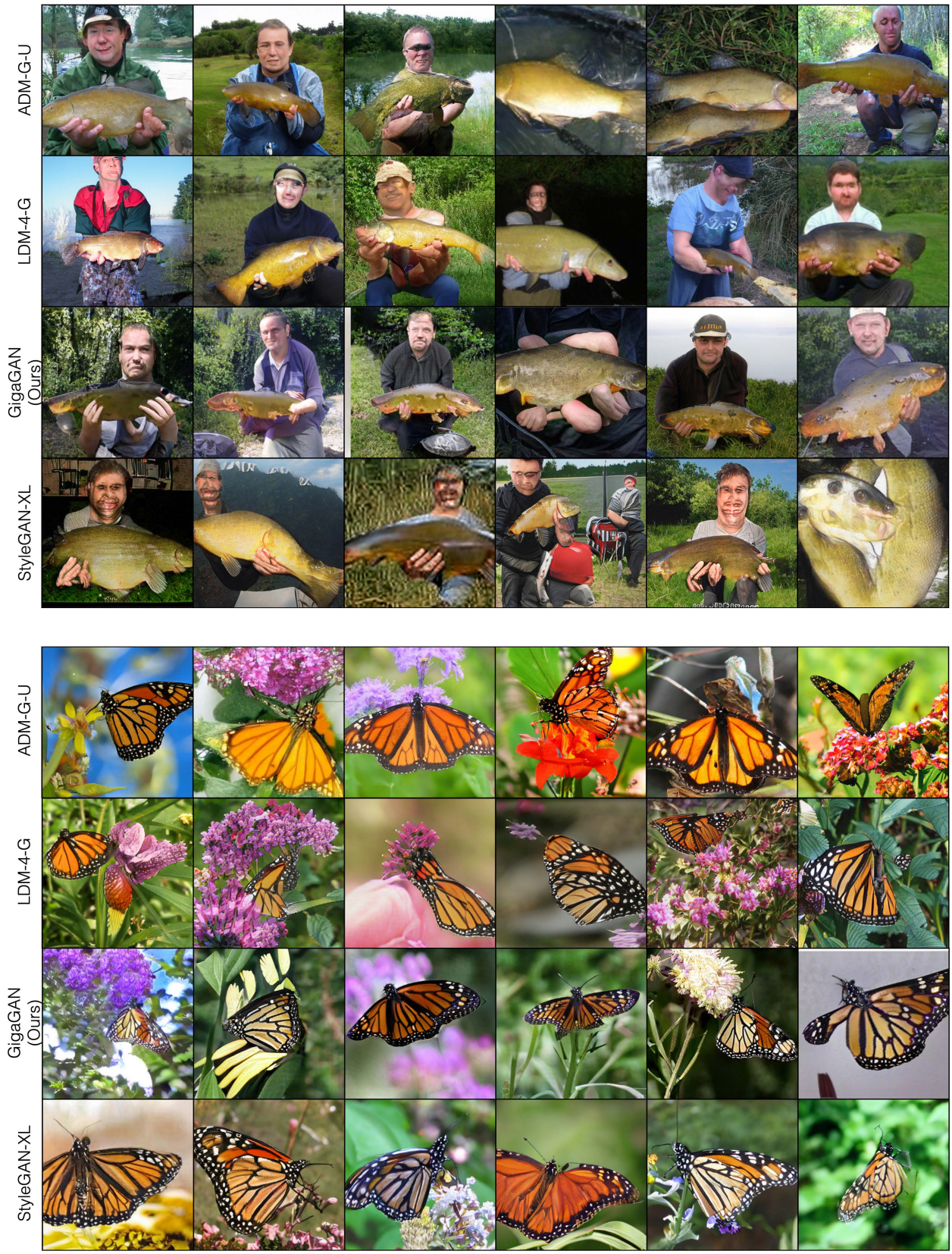


Figure A1. Uncurated images (above: Tench and below: Monarch) from ADM-G-U [15], LDM-4-G [79], GigaGAN (ours), and StyleGAN-XL [86]. FID values of each generative model are 4.01, 3.60, 3.45, and 2.32, respectively.

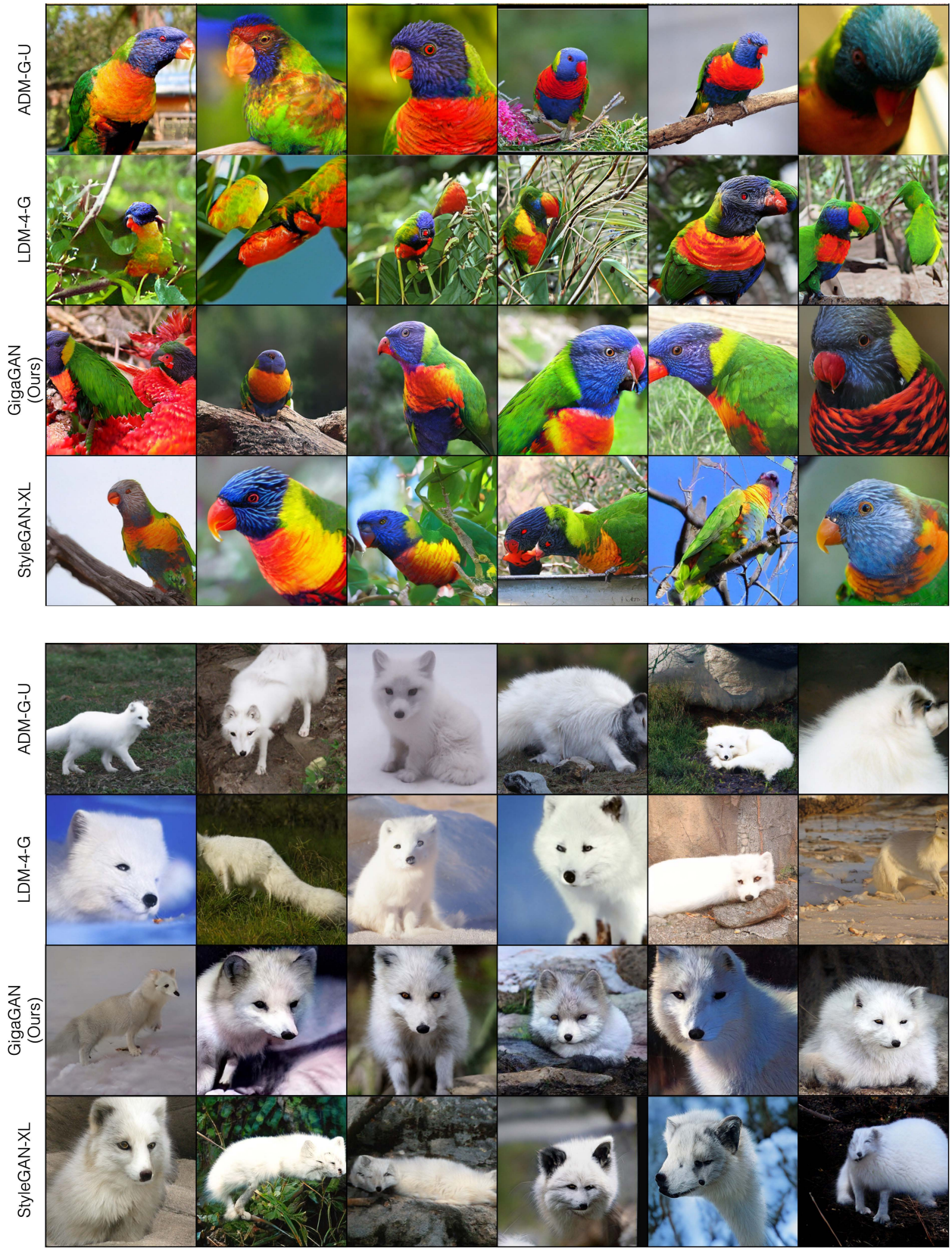


Figure A2. Uncurated images (above: Lorikeet and below: Arctic fox) from ADM-G-U [15], LDM-4-G [79], GigaGAN (ours), and StyleGAN-XL [86]. FID values of each generative model are 4.01, 3.60, 3.45, and 2.32, respectively.

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].

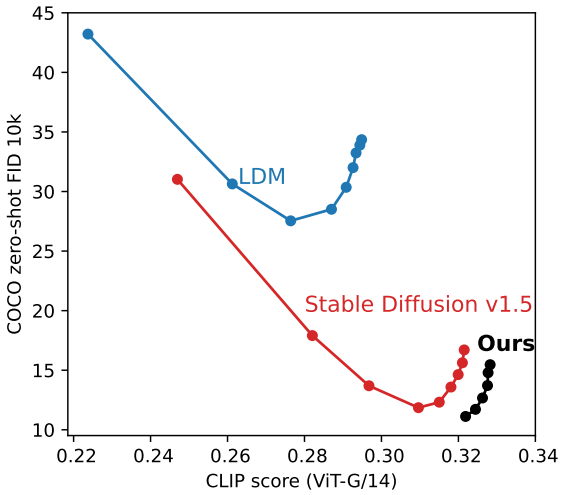


Figure A3. We investigate how our FID and CLIP score changes over different truncation values [1.0, 0.9, 0.8, 0.7, 0.6, 0.5], by visualizing them along with the FID-CLIP score curve of two publicly available large scale diffusion models: LDM and Stable Diffusion. It is seen that the CLIP score increases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

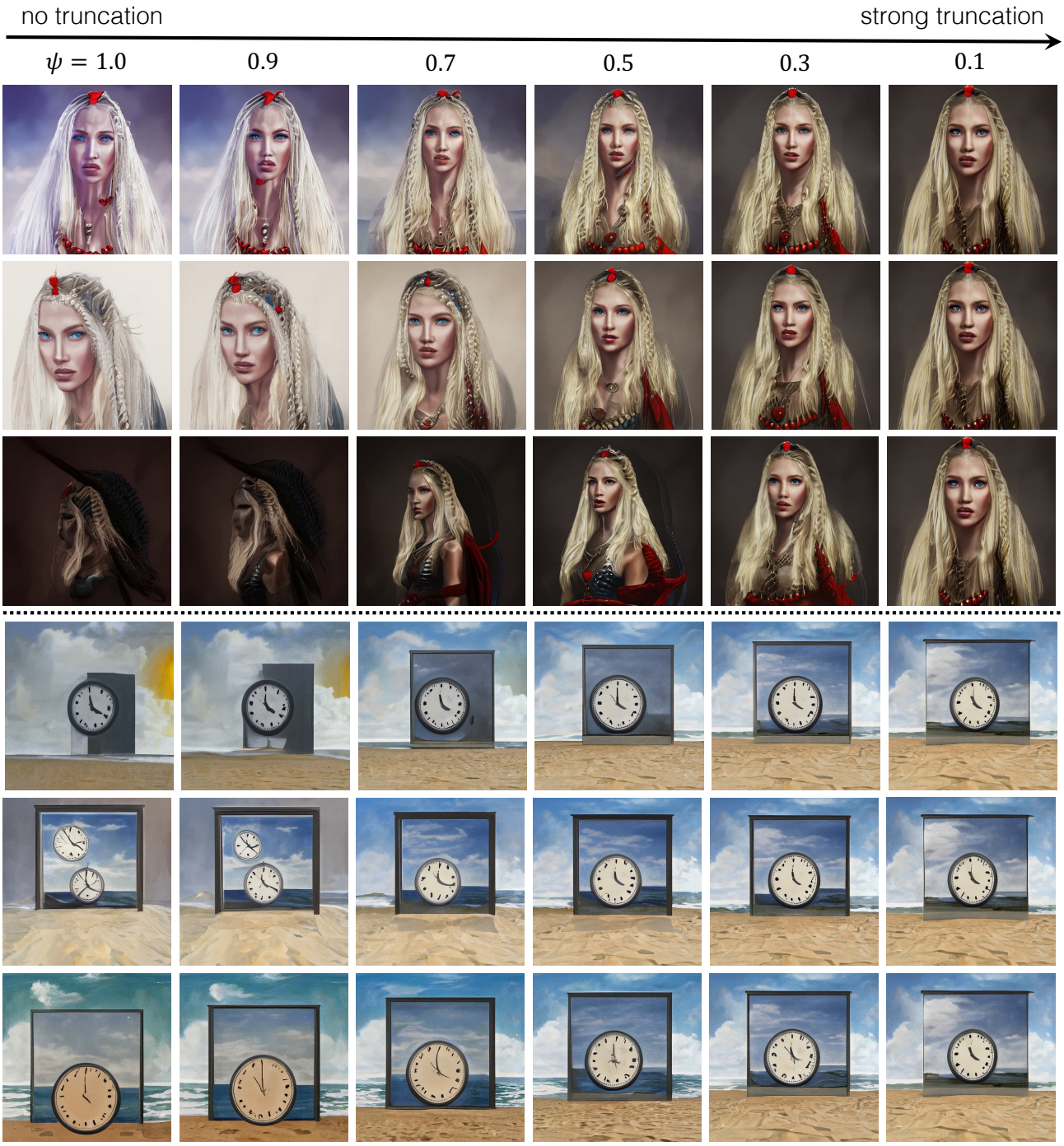


Figure A4. **The visual effect of our truncation trick**. We demonstrate the effect of truncation by decreasing the truncation value ψ from 1.0. We show six example outputs with the text prompt "digital painting of a confident and severe looking northern war goddess, extremely long blond braided hair, 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.

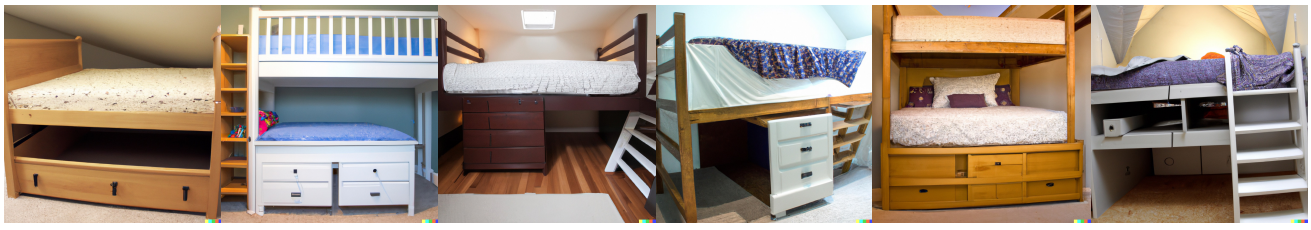
“A loft bed with a dresser underneath it.”

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 A7. 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)

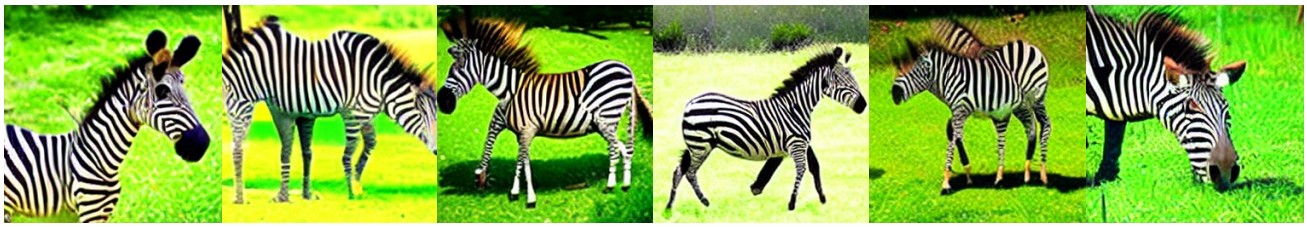
DALL·E 2 (1024px)

Figure A8. 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.”

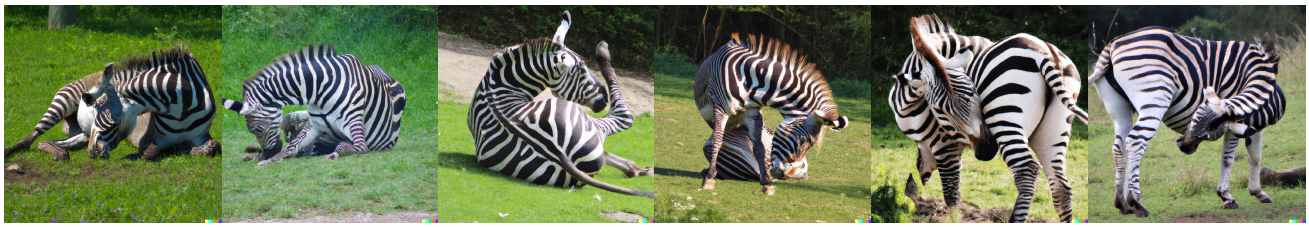
Ours (512px, 0.13s / img)

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)

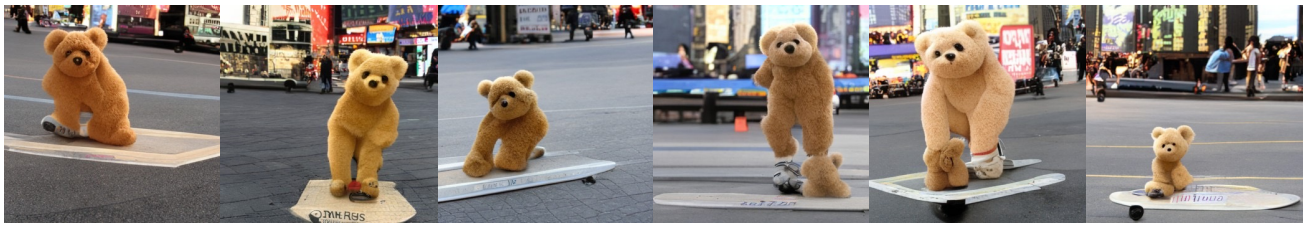


DALL·E 2 (1024px)

Figure A9. 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 A10. 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)

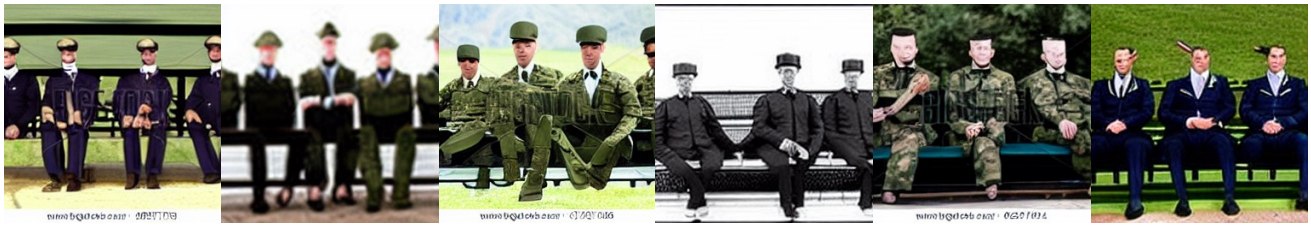
DALL·E 2 (1024px)

Figure A11. 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].

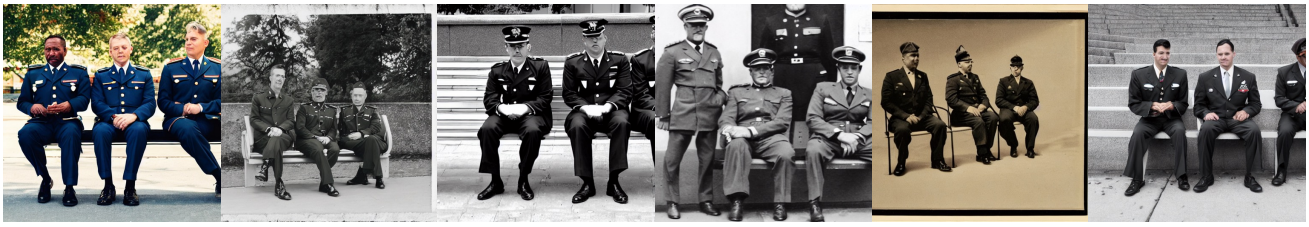
“Three men in military suits are sitting on a bench.”

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 A12. 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].

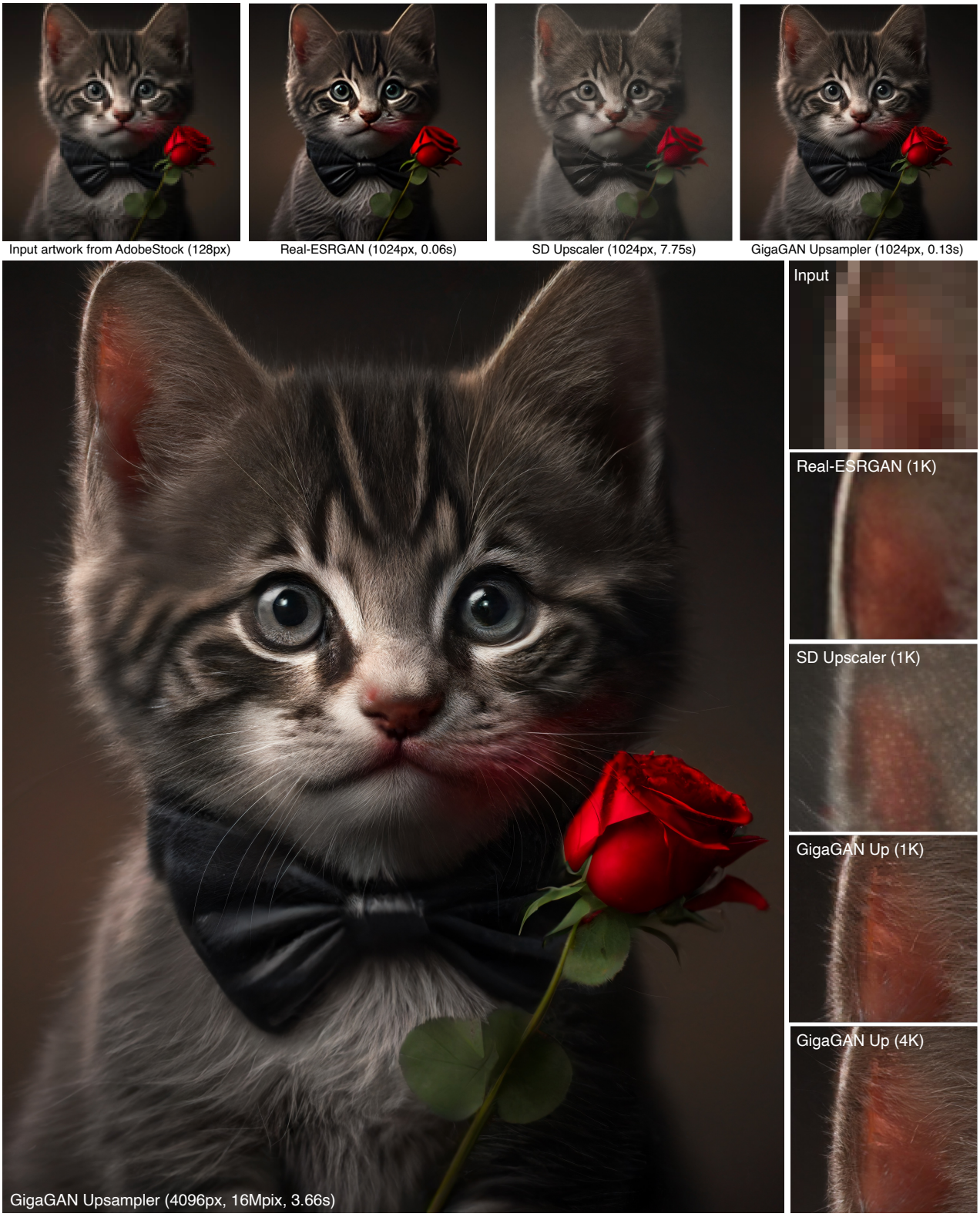


Figure A13. **Our GAN-based upsampler** can serve as the upsampler for many text-to-image models that generate initial outputs at low resolutions like 64px or 128px. We simulate such usage by applying our 8× superresolution model on a low-res 128px artwork to obtain the 1K output, using "Portrait of 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 A14. **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 A15. **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.