Lecture 13
Generative Models: GANs, Diffusion, and Beyond (part 2)

Olexandr Isayev

Department of Chemistry, CMU

olexandr@cmu.edu

In Class Project Presentations

Wednesday, April 26

~8-9 min/project ~5-7 slides

Peer grading

Please upload your code, writeup (~2-3 pages) and slides by May 5, for final grading

Class project presentation

- Background & Scientific Problem
- Dataset
- Data curation and processing (if any)
- ML experiments
- Lessons Learned

- Please spend most time with last 3 points

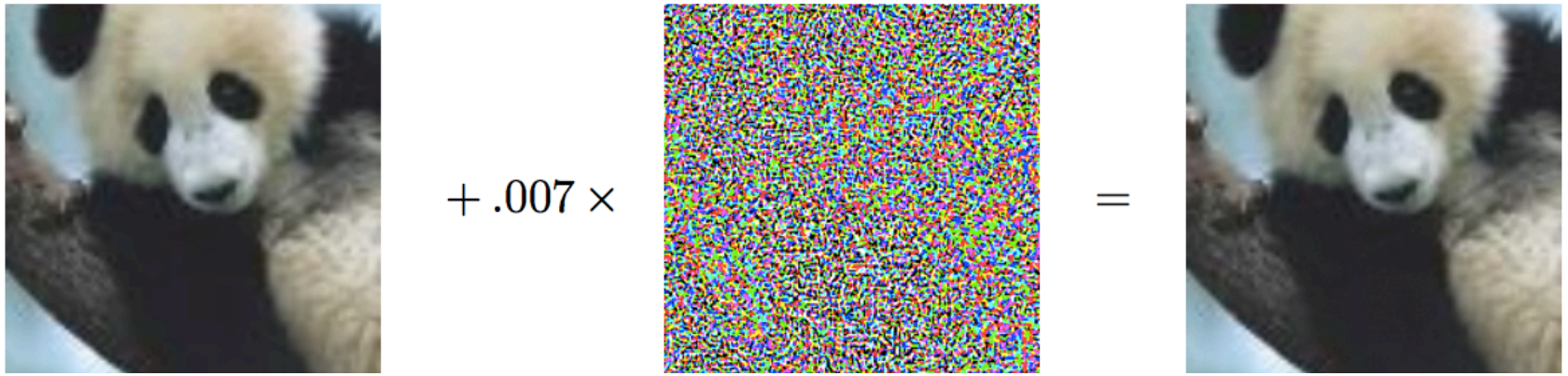
Class Feedback

Please fill out course evaluation forms.

If you have any private feedback, suggestions to improve the course & MS-DAS program please reach me out:

olexandr@cmu.edu

GANs: Fooling neural networks



"panda"
57.7% confidence

"gibbon"
99.3 % confidence

Properties of adversarial examples: Summary

- For any input image, it is usually easy to generate a very similar image that gets misclassified by the same network
- To obtain an adversarial example, one does not need to do precise gradient ascent
- Adversarial images can (somewhat) survive transformations like being printed and photographed
- It is possible to attack many images with the same perturbation
- Adversarial examples that can fool one network have a high chance of fooling a network with different parameters and even architecture

Generative adversarial networks (GANs)

Train two networks with opposing objectives:

- **Generator:** learns to generate samples
- **Discriminator:** learns to distinguish between generated and real samples

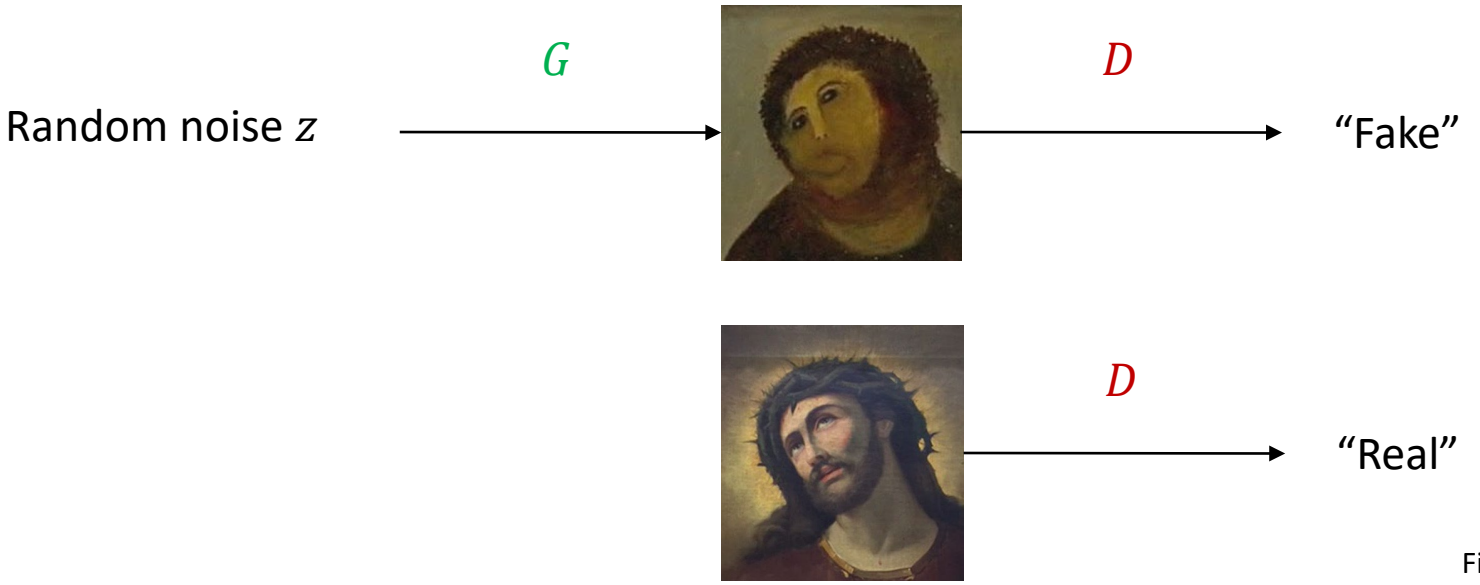
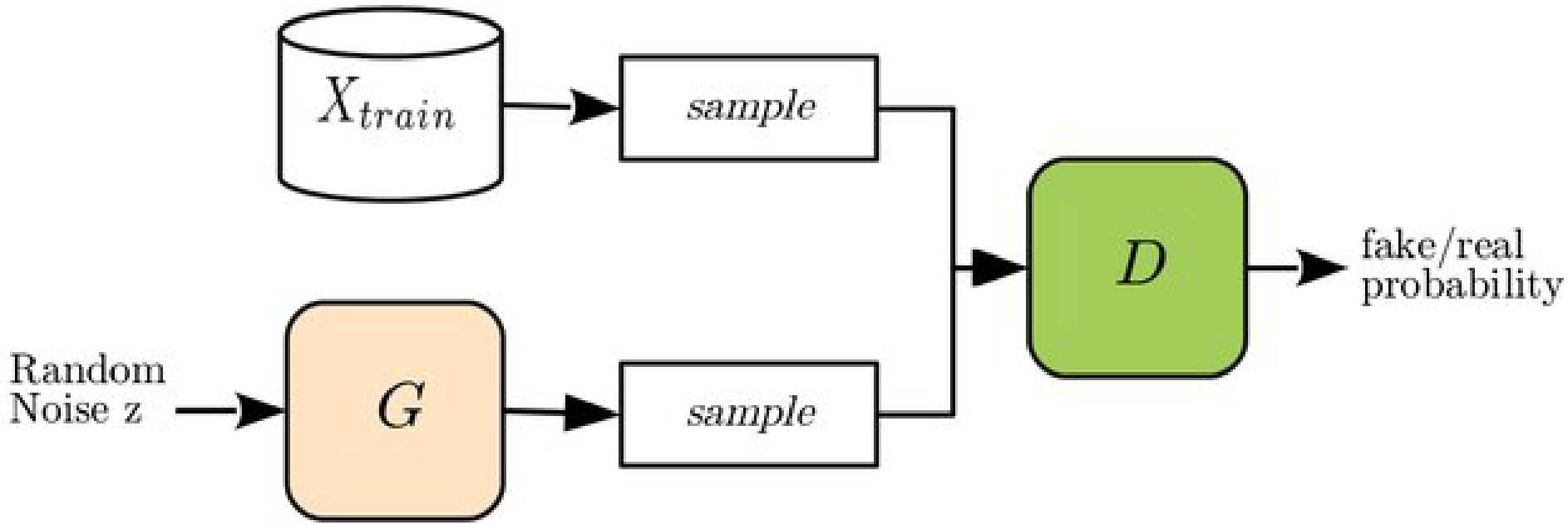


Figure adapted from F. Fleuret

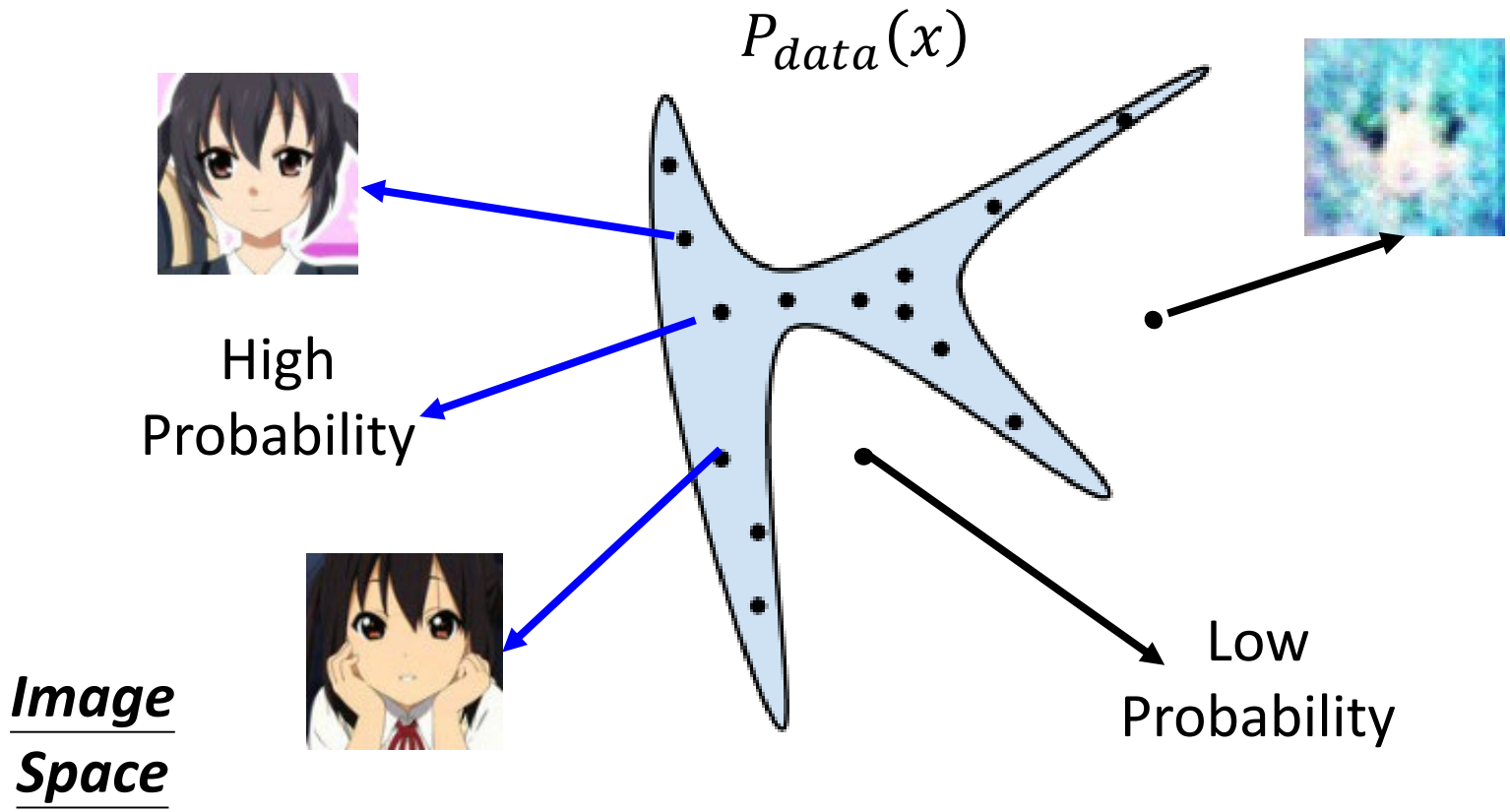
I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, NIPS 2014

Generative adversarial networks (GANs)



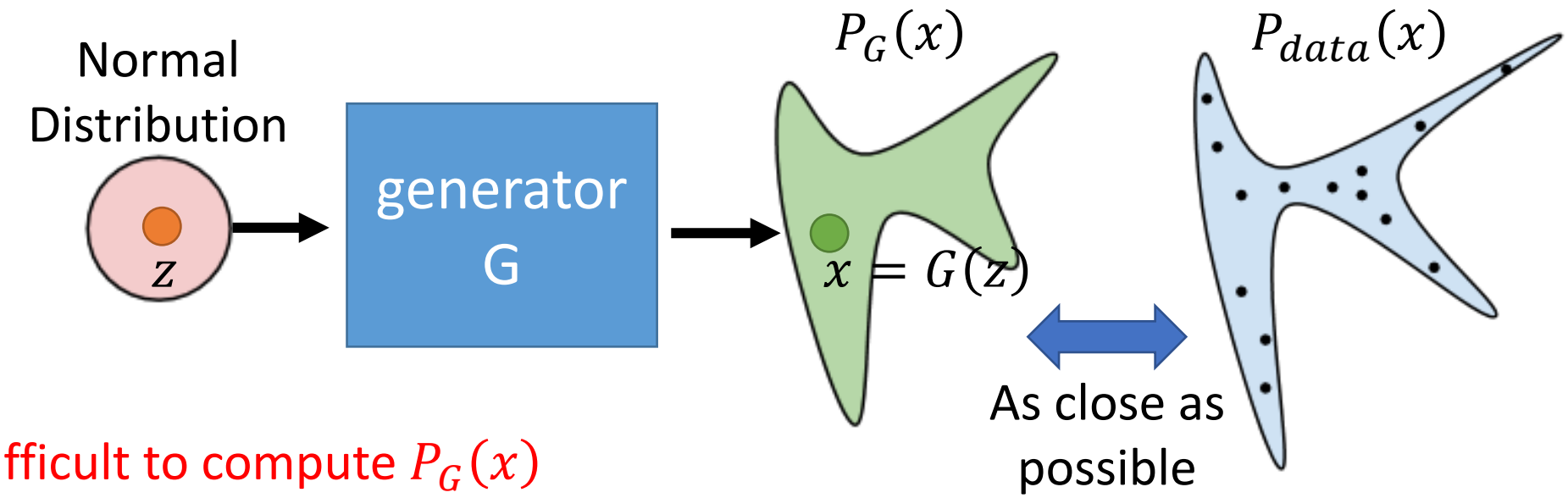
Basic Idea of GAN

- The data we want to generate has a distribution $P_{data}(x)$



Basic Idea of GAN

- A generator G is a network. The network defines a probability distribution.

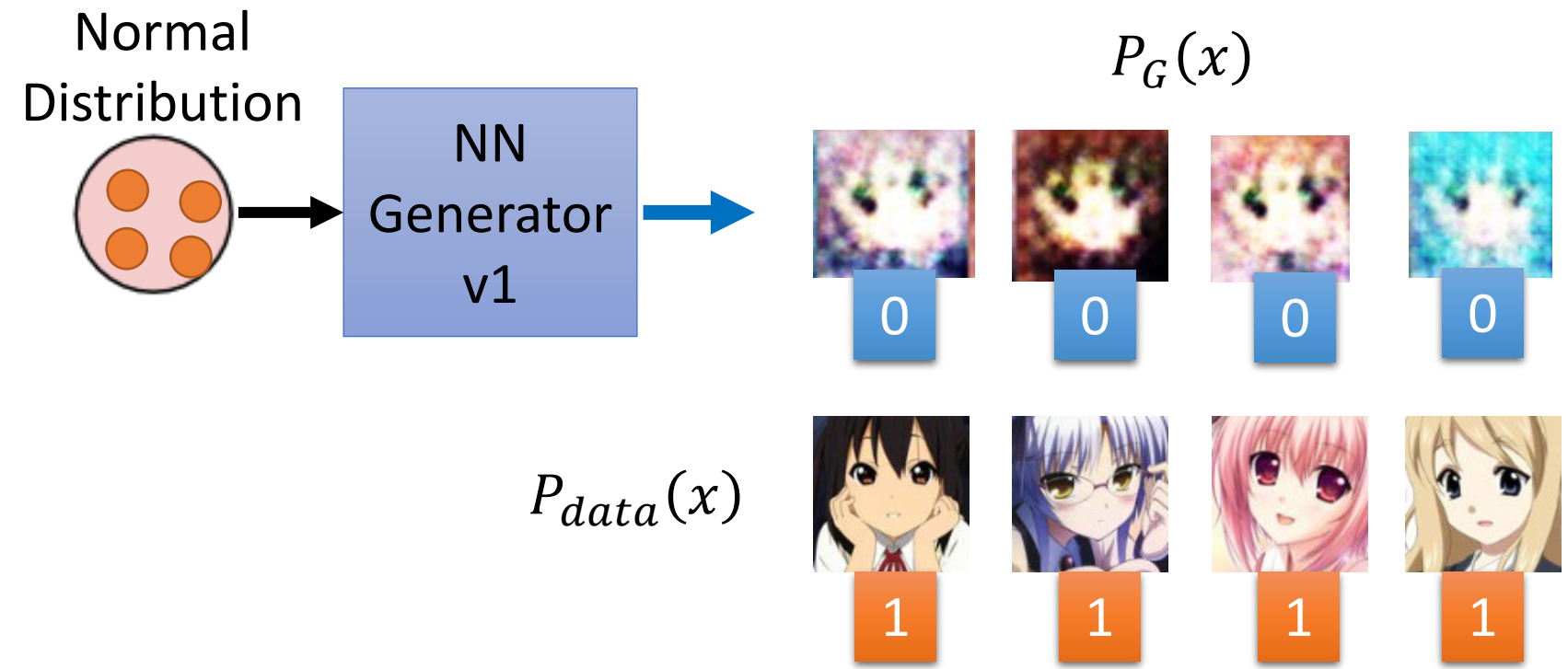


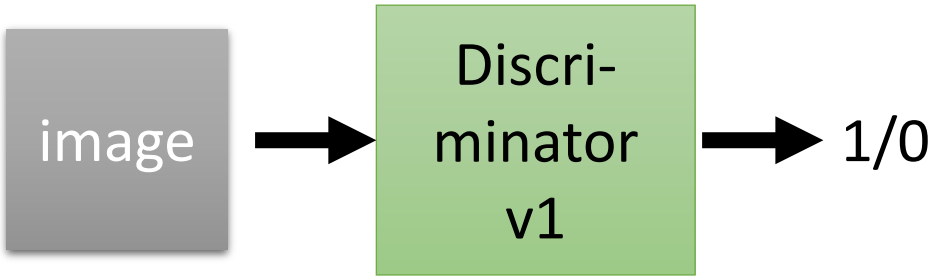
It is difficult to compute $P_G(x)$

We do not know what the distribution looks like.

https://blog.openai.com/generative-models/

Basic Idea of GAN





It can be proofed that the ***loss the discriminator*** related to Jensen–Shannon divergence

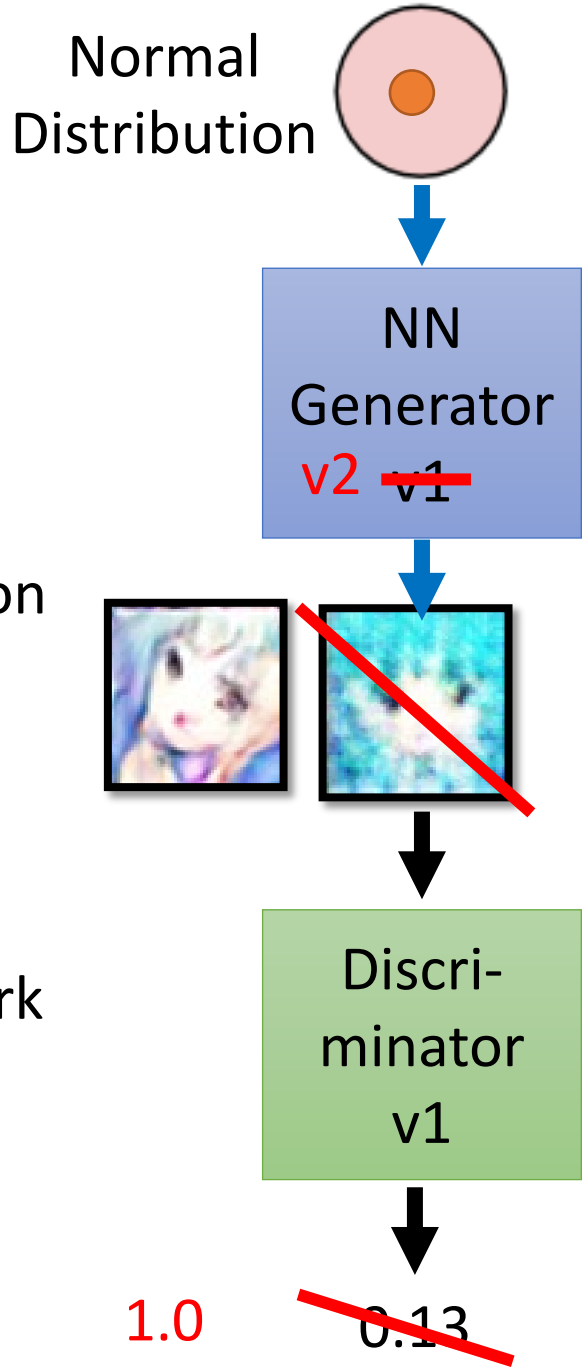
Basic Idea of GAN

- ***Next step***:
  - Updating the parameters of generator
  - To minimize the Jensen–Shannon divergence

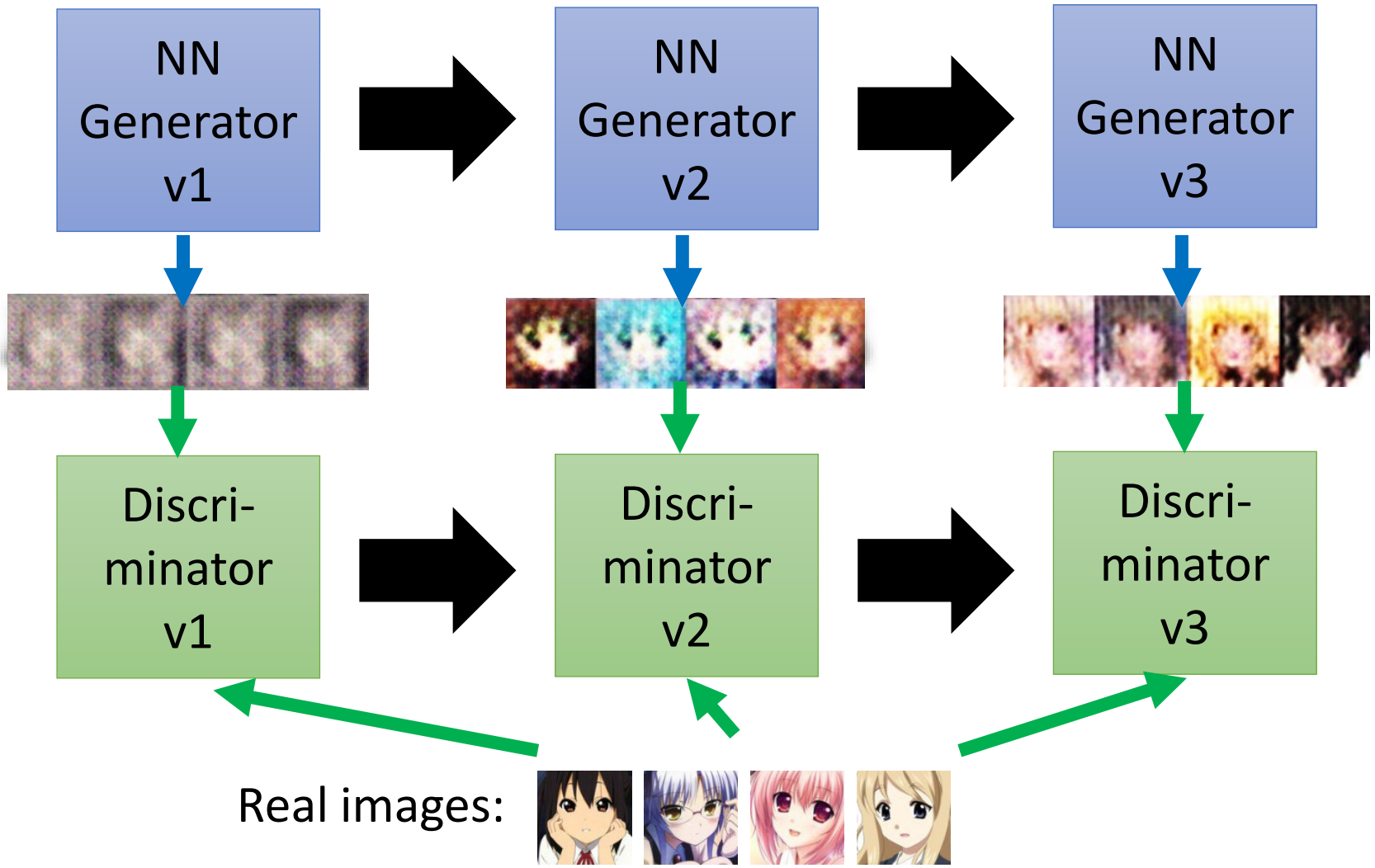
➡ The output be classified as "real" (as close to 1 as possible)

Generator + Discriminator = a network

Using gradient descent to update the parameters in the generator, but fix the discriminator



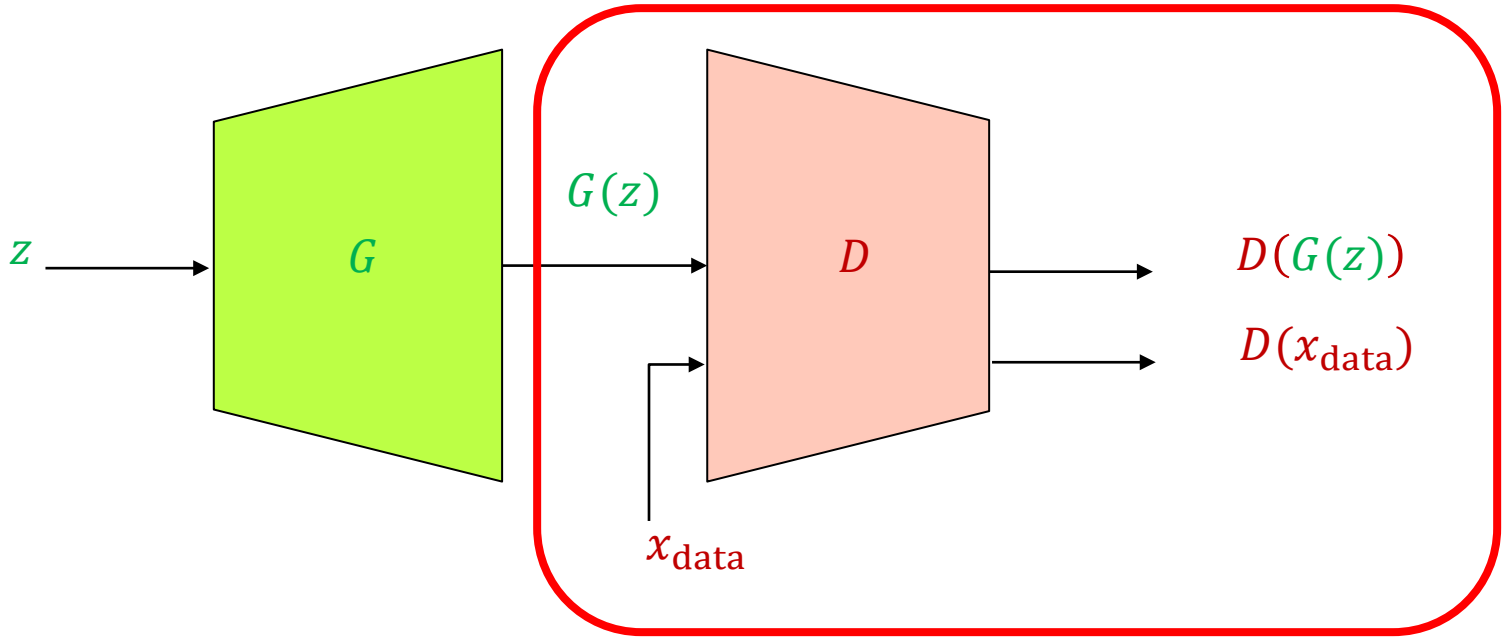
The evolution of generation



GAN: Conceptual picture

Update discriminator: push $D(x_{\text{data}})$ close to 1 and $D(G(z))$ close to 0

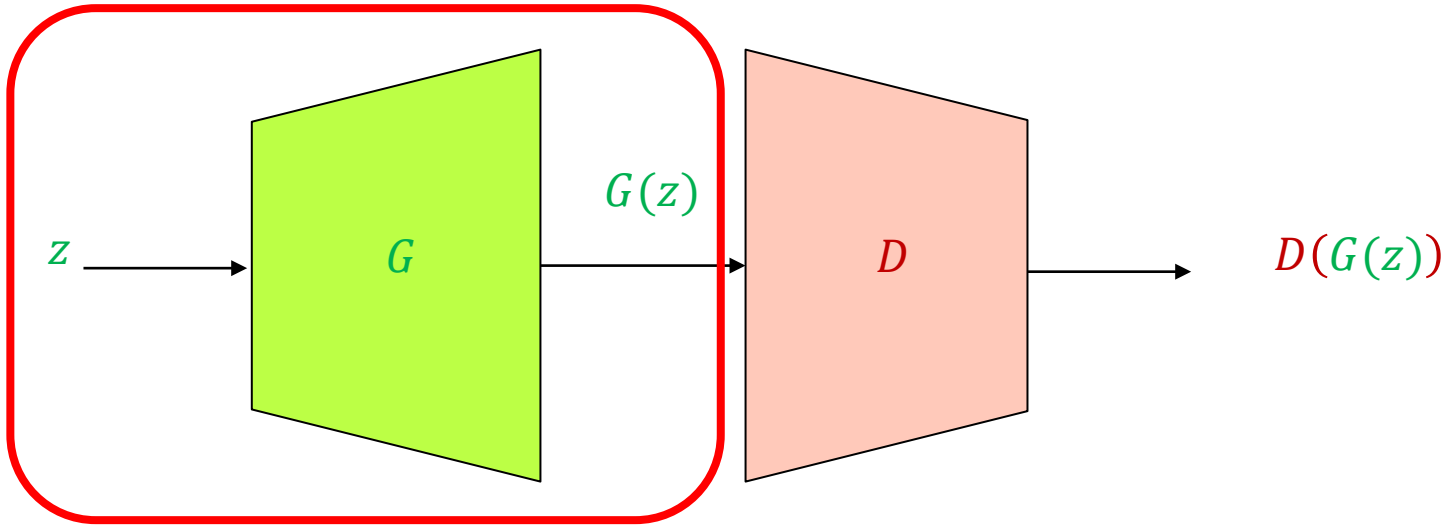
- The generator is a “black box” to the discriminator



GAN: Conceptual picture

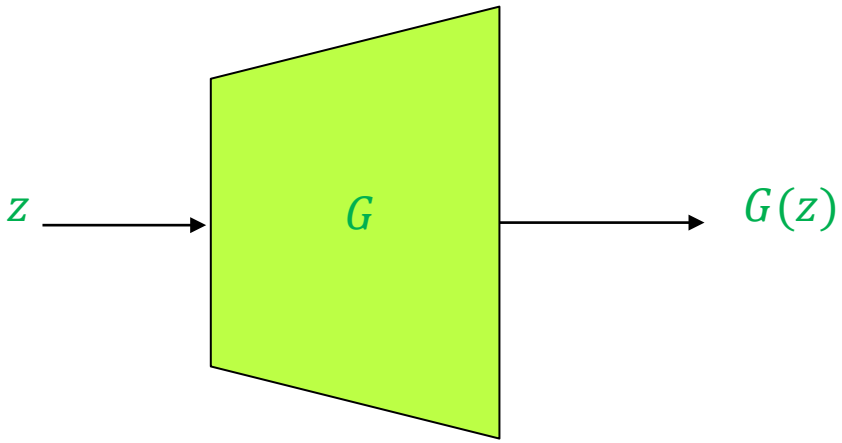
Update generator: increase $D(G(z))$

- Requires back-propagating through the composed generator-discriminator network (i.e., the discriminator cannot be a black box)
- The generator is exposed to real data only via the output of the discriminator (and its gradients)



GAN: Conceptual picture

- Test time – the discriminator is discarded



Original GAN results

MNIST digits

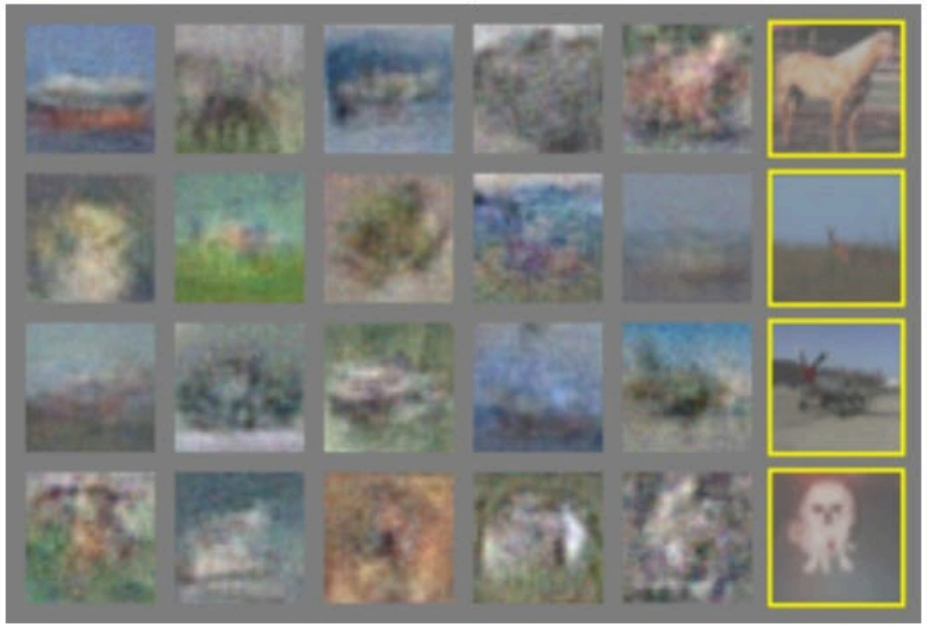
Toronto Face Dataset

Nearest real image for
sample to the left

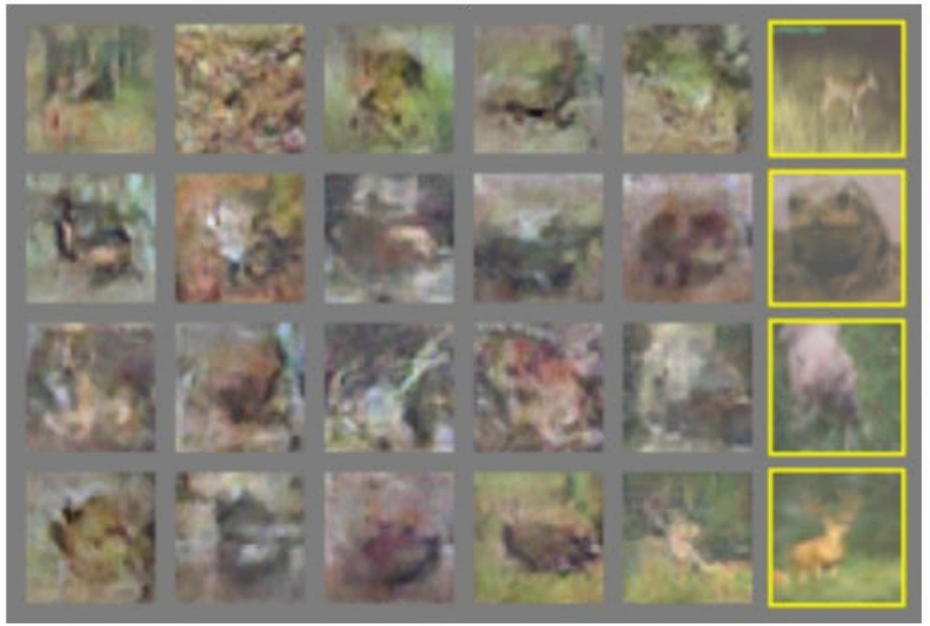
I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, NIPS 2014

Original GAN results

CIFAR-10 (FC networks)



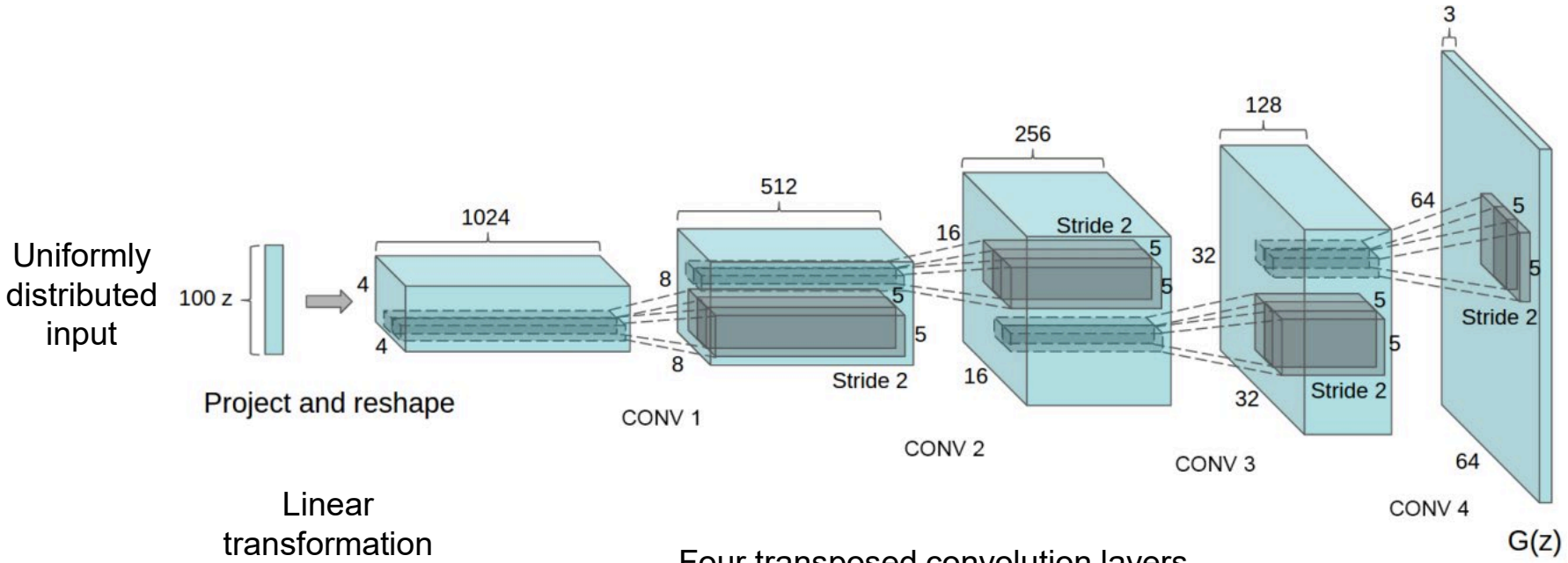
CIFAR-10 (conv networks)



I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, NIPS 2014

DCGAN

- Early, influential convolutional architecture for generator



A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, ICLR 2016

DCGAN

- Early, influential convolutional architecture for generator
- Discriminator architecture (empirically determined to give best training stability):
  - Don’t use pooling, only strided convolutions
  - Use Leaky ReLU activations (sparse gradients cause problems for training)
  - Use only one FC layer before the softmax output
  - Use batch normalization after most layers (in the generator also)

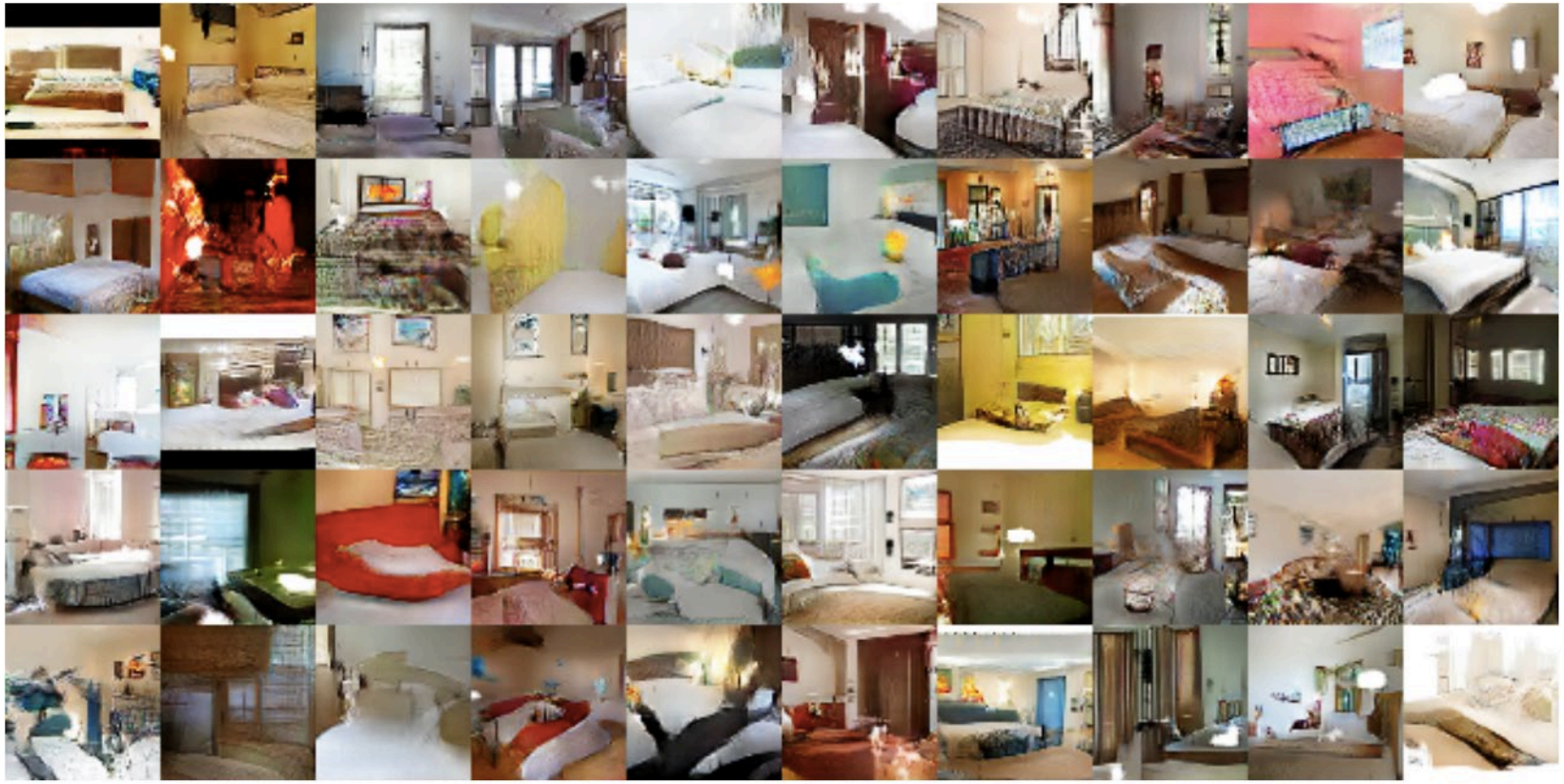
A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, ICLR 2016

DCGAN results

Generated bedrooms after one epoch

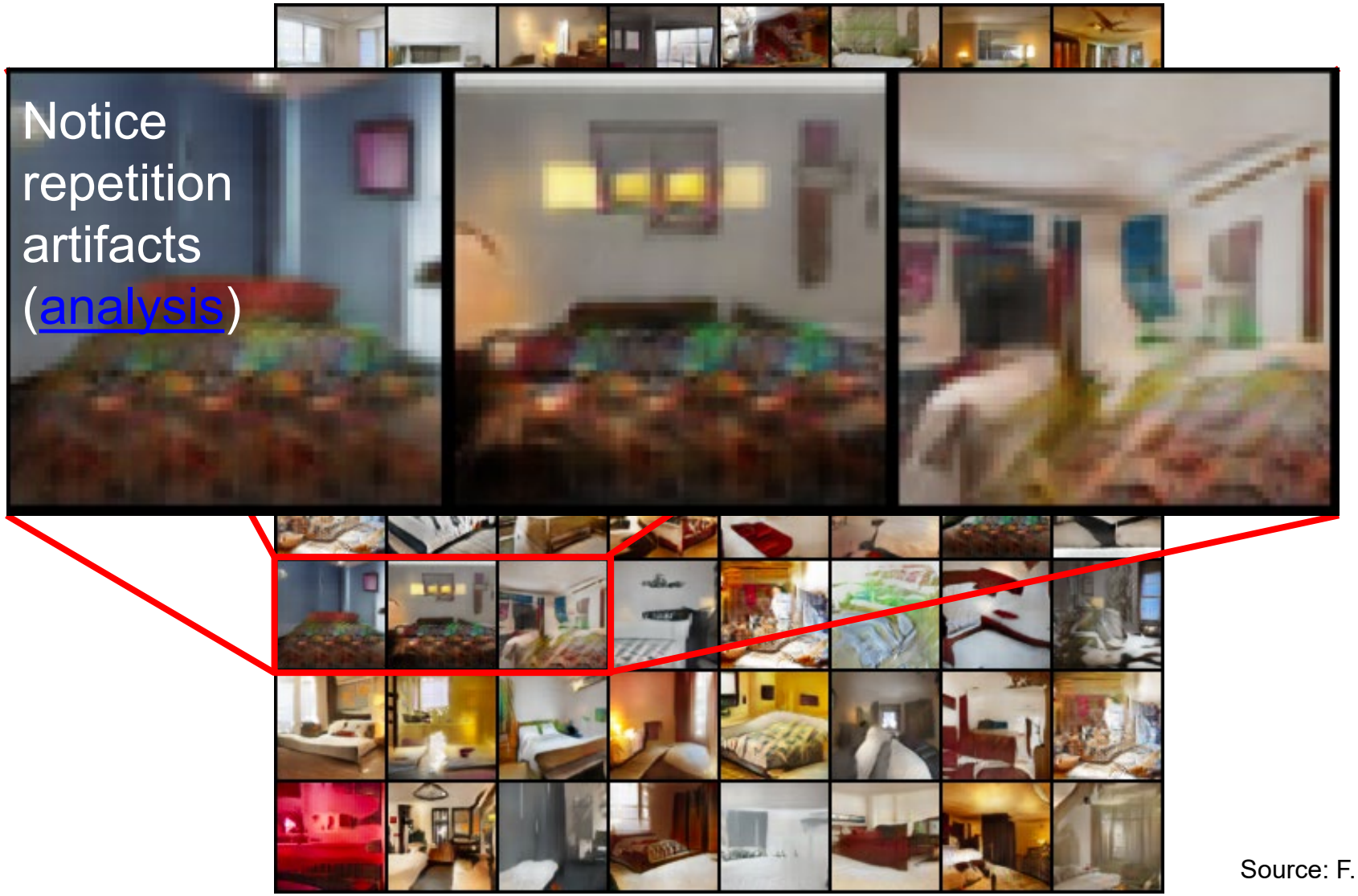
DCGAN results

Generated bedrooms after five epochs



DCGAN results

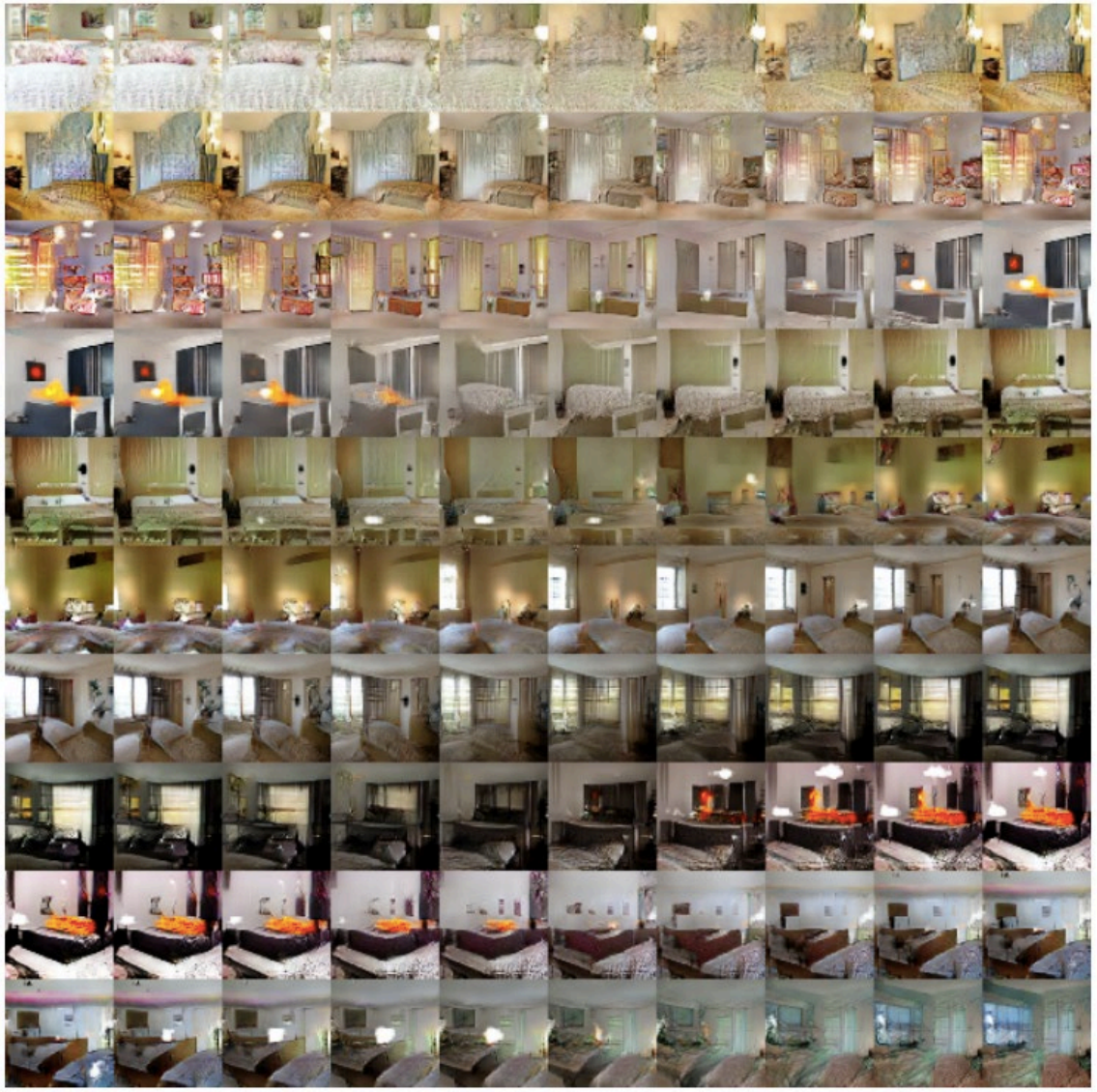
More bedrooms



Source: F. Fleuret

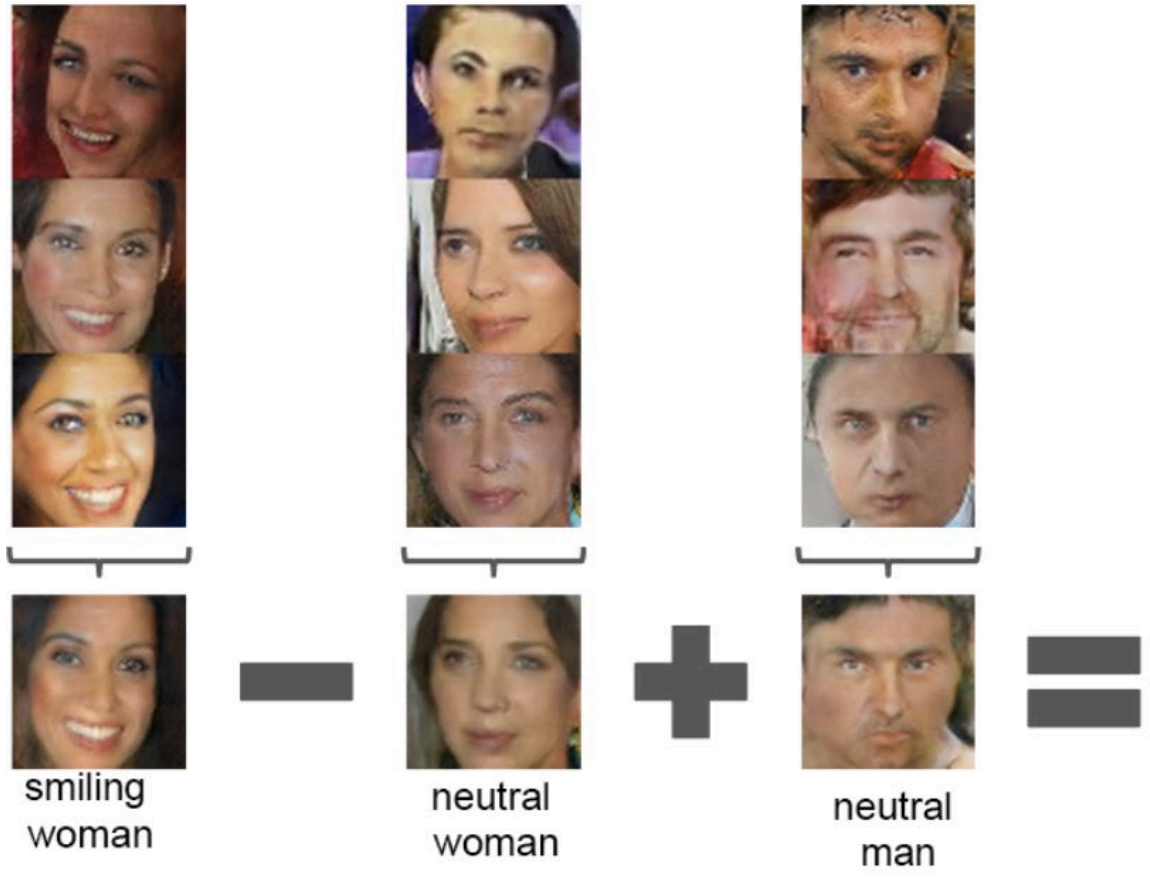
DCGAN results

Interpolation between different points in the z space



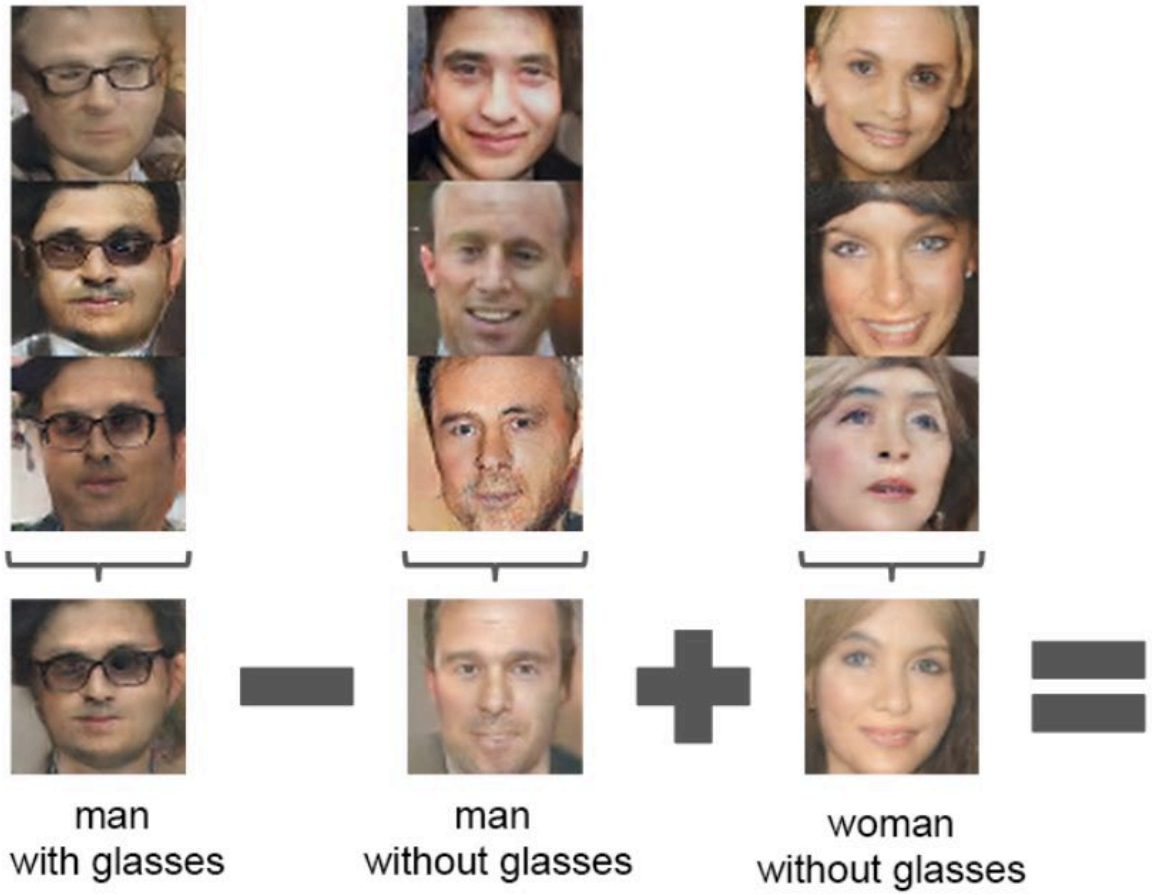
DCGAN results

- Vector arithmetic in the z space



DCGAN results

- Vector arithmetic in the z space



DCGAN results

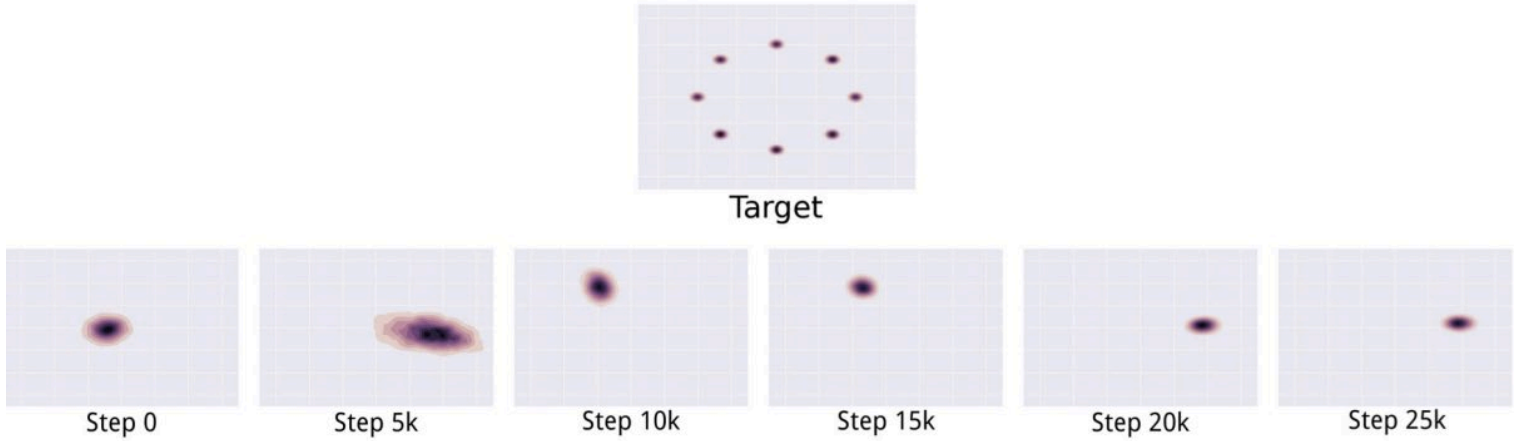
- Pose transformation by adding a “turn” vector

Problems with GAN training

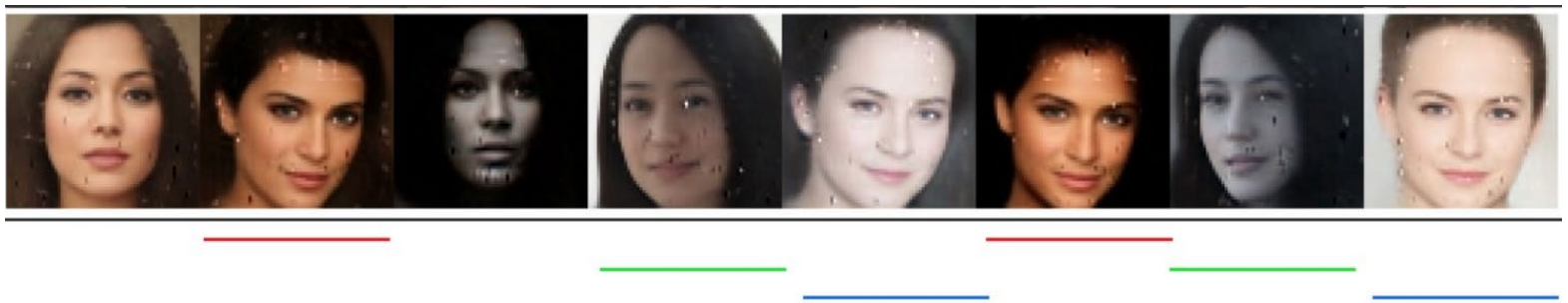
- Stability
  - Parameters can oscillate or diverge, generator loss does not correlate with sample quality
  - Behavior very sensitive to hyperparameter selection

Problems with GAN training

- Mode collapse
  - Generator ends up modeling only a small subset of the training data



Source



Source

Wasserstein GAN (WGAN)

Motivated by *Wasserstein or Earth mover's distance*, which is an alternative to JS divergence for comparing distributions

- In practice, use linear activation instead of sigmoid in the discriminator and drop the logs from the objective:

$$\min_G \max_D \left[\mathbb{E}_{x \sim p_{\text{data}}} D(x) - \mathbb{E}_{z \sim p} D(G(z)) \right]$$

- Due to theoretical considerations, important to ensure smoothness of discriminator
- This paper's suggested method is clipping weights to fixed range $[-c, c]$

M. Arjovsky, S. Chintala, L. Bottou, Wasserstein generative adversarial networks, ICML 2017

Least Squares GAN (LSGAN)

- Use least squares cost for generator and discriminator
  - Equivalent to minimizing Pearson χ^2 divergence

$$D^* = \arg\min_D \left[\mathbb{E}_{x \sim p_{\text{data}}} (D(x) - 1)^2 + \mathbb{E}_{z \sim p} (D(G(z)))^2 \right]$$

Push discrim. response on real data close to 1

Push response on generated data close to 0

$$G^* = \arg\min_G \mathbb{E}_{z \sim p} (D(G(z)) - 1)^2$$

Push response on generated data close to 1

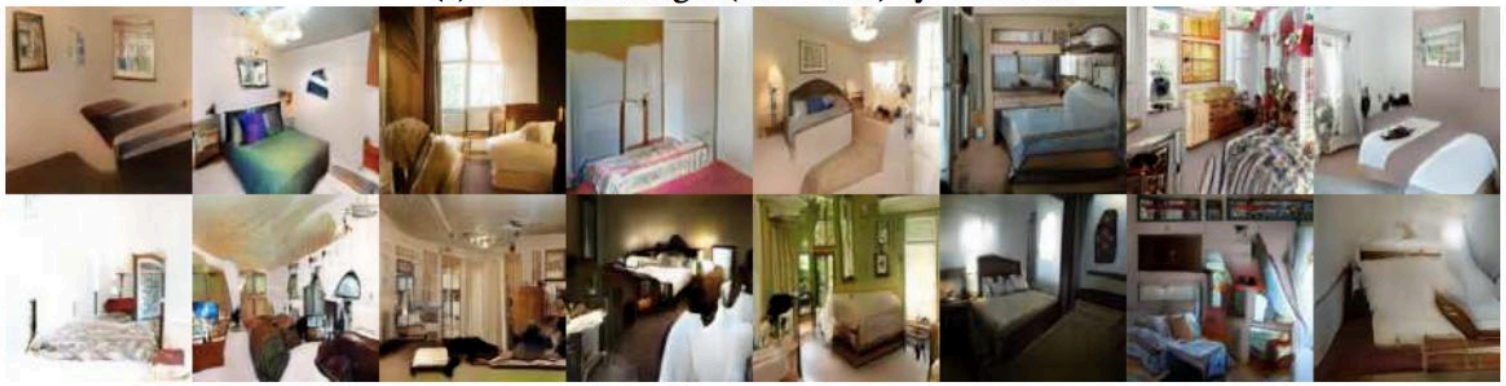
X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley, Least squares generative adversarial networks, ICCV 2017

Least Squares GAN (LSGAN)

Benefits (claimed)

- Higher-quality images

(a) Generated images (112 × 112) by LSGANs.



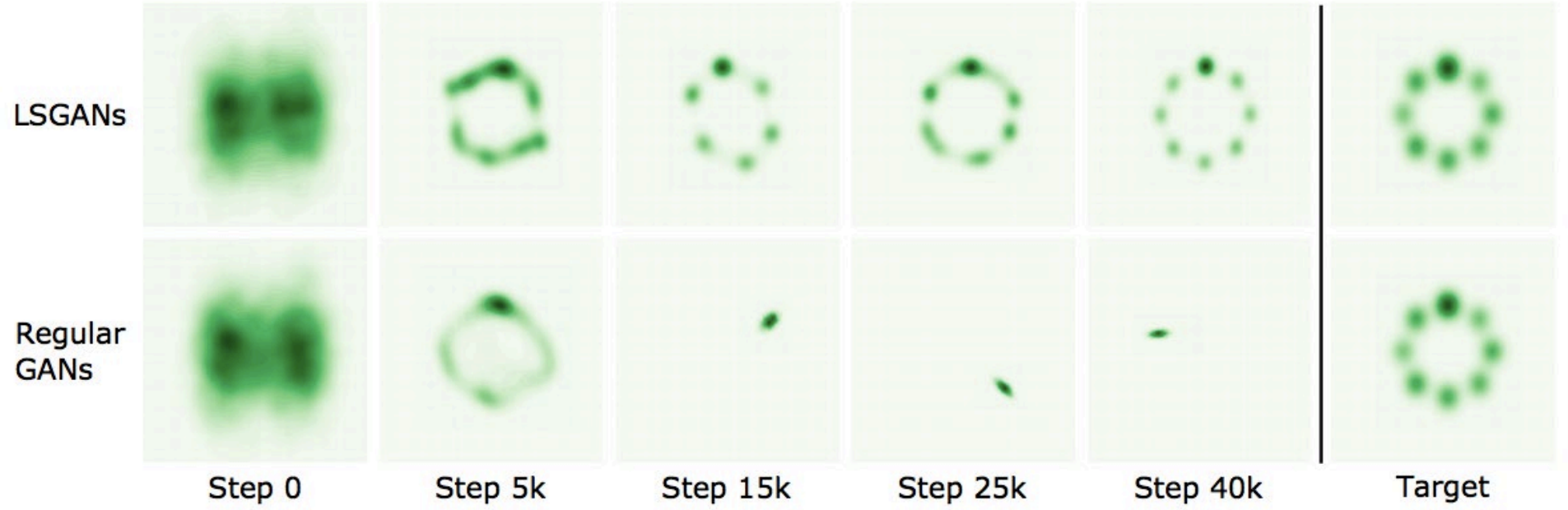
(b) Generated images (112 × 112) by DCGANs.

X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley, Least squares generative adversarial networks, ICCV 2017

Least Squares GAN (LSGAN)

Benefits (claimed)

- Higher-quality images
- More stable and resistant to mode collapse



X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley, Least squares generative adversarial networks, ICCV 2017

Recent progress in GANs

Recent progress in GANs

EBGAN (2017)

BigGAN (2018)

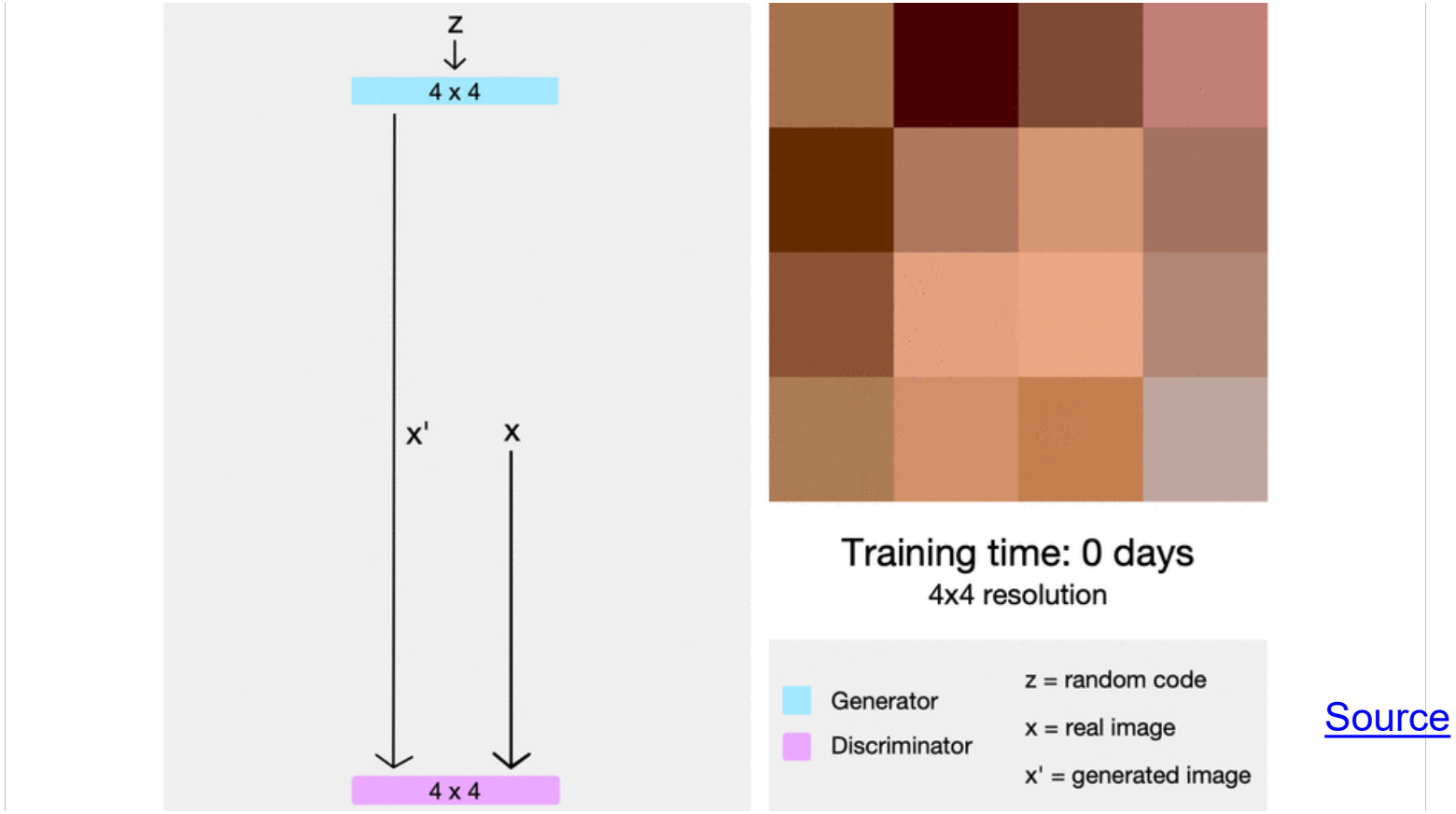
Progressive GANs

Realistic face images up to 1024 x 1024 resolution

T. Karras, T. Aila, S. Laine, J. Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. ICLR 2018

Progressive GANs

Key idea: train lower-resolution models, gradually add layers corresponding to higher-resolution outputs



Source

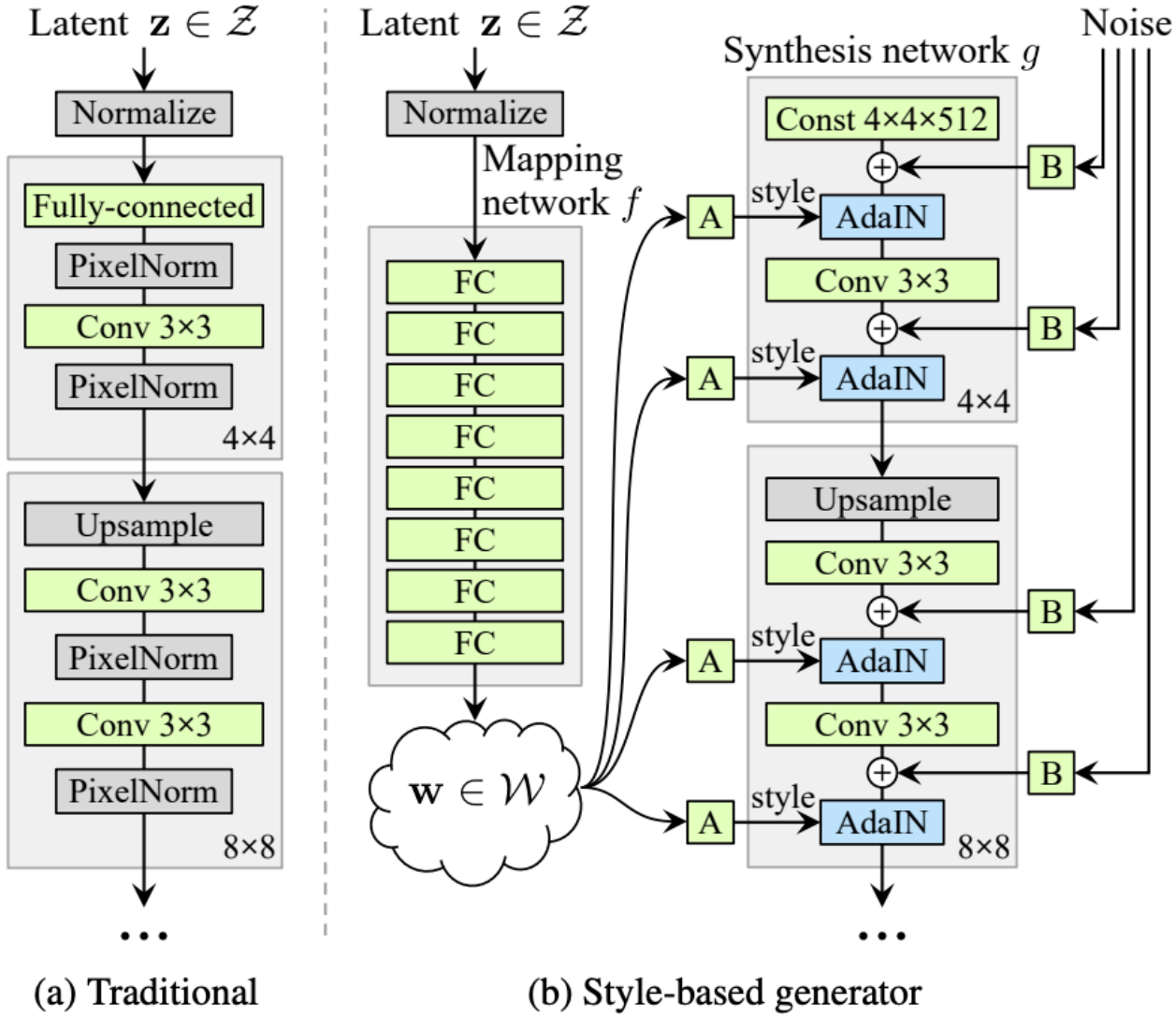
T. Karras, T. Aila, S. Laine, J. Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. ICLR 2018

Progressive GANs: Results

256 x 256 results for LSUN categories

StyleGAN

- Built on top of Progressive GAN
- Start generation with constant (instead of noise vector)
- Noise vector is transformed to latent vector w that is later specialized to *style codes*
- Style codes control *adaptive instance normalization* (AdaIN) or scaling and biasing of each feature map
- Add noise after each convolution and before nonlinearity (enables stochastic detail)

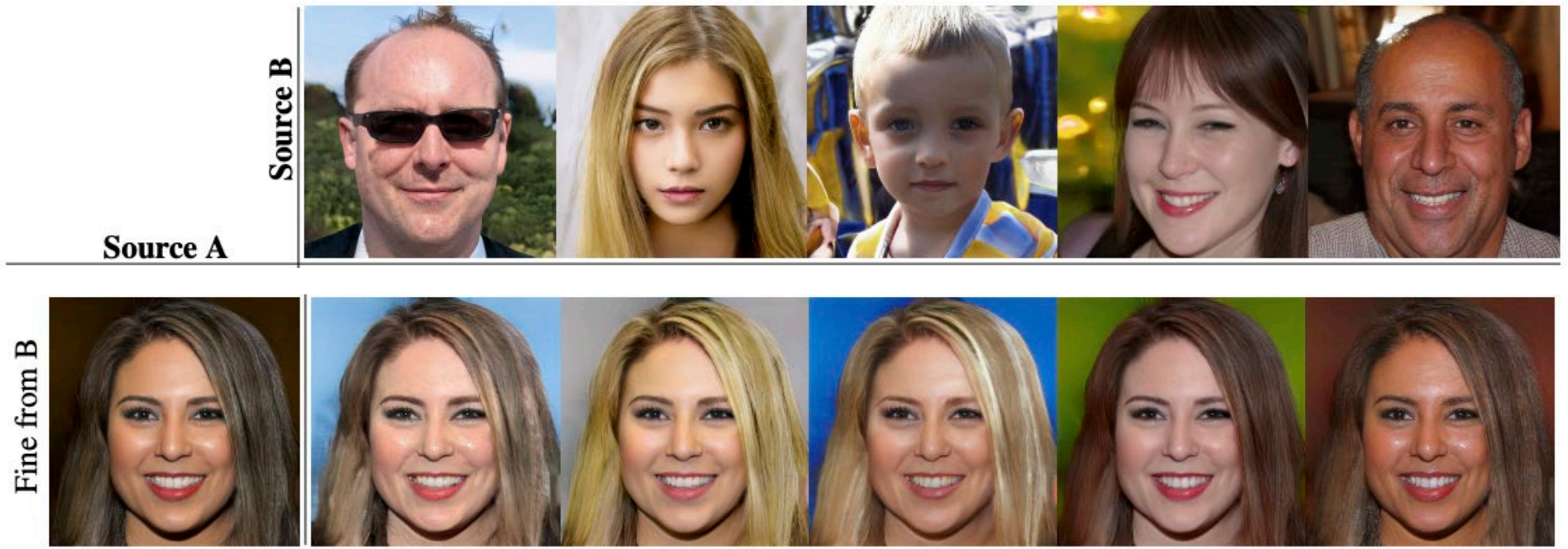


T. Karras, S. Laine, T. Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. CVPR 2019

StyleGAN: Results

T. Karras, S. Laine, T. Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. CVPR 2019

Mixing styles

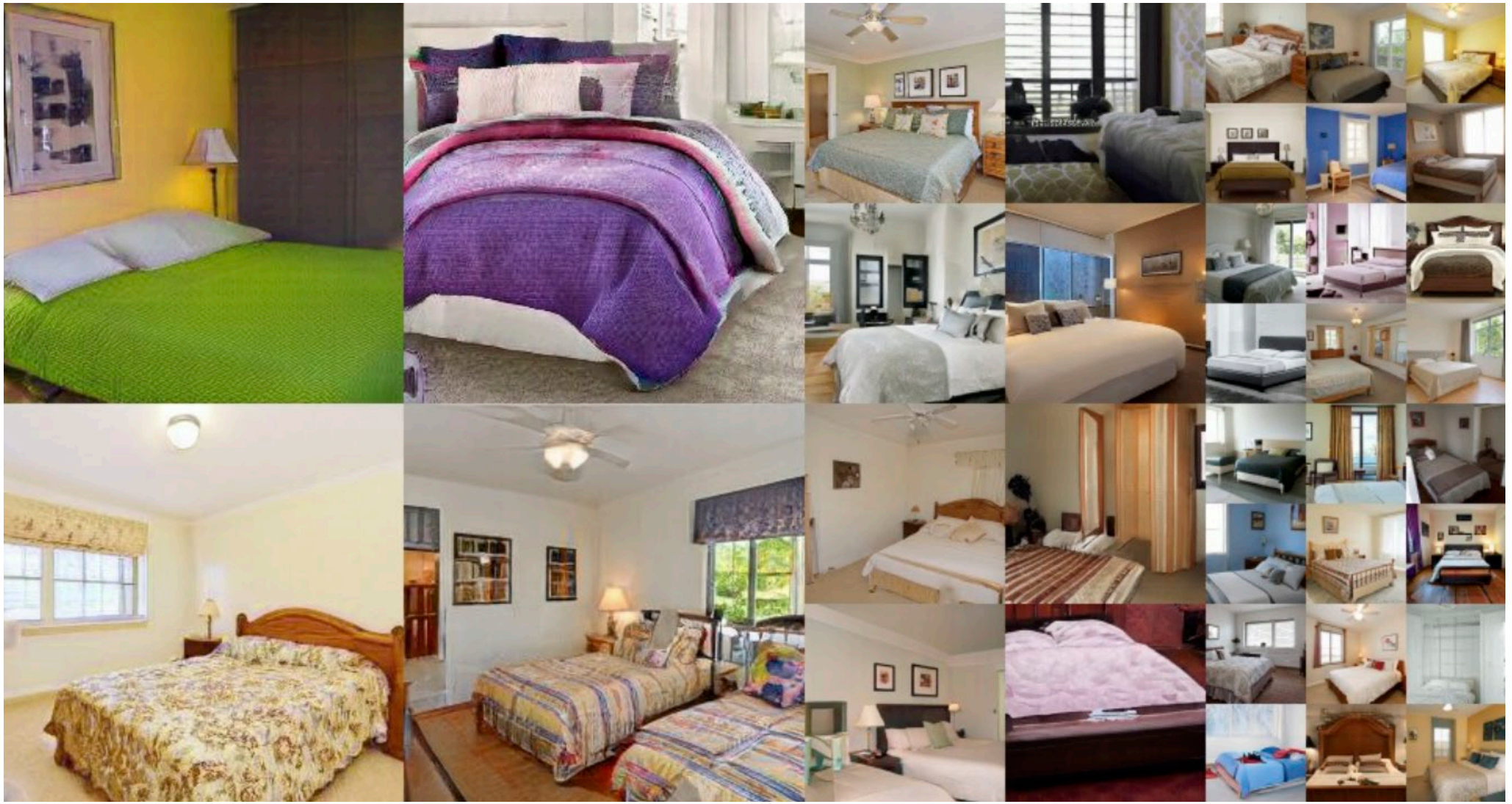


“Two sets of images were generated from their respective latent codes (sources A and B); the rest of the images were generated by copying a specified subset of styles from source B and taking the rest from source A.”

Mixing styles

Mixing styles

StyleGAN: Bedrooms



T. Karras, S. Laine, T. Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. CVPR 2019

StyleGAN: Cars

T. Karras, S. Laine, T. Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. CVPR 2019

StyleGAN2

- Change normalization, remove progressive growing to address StyleGAN artifacts

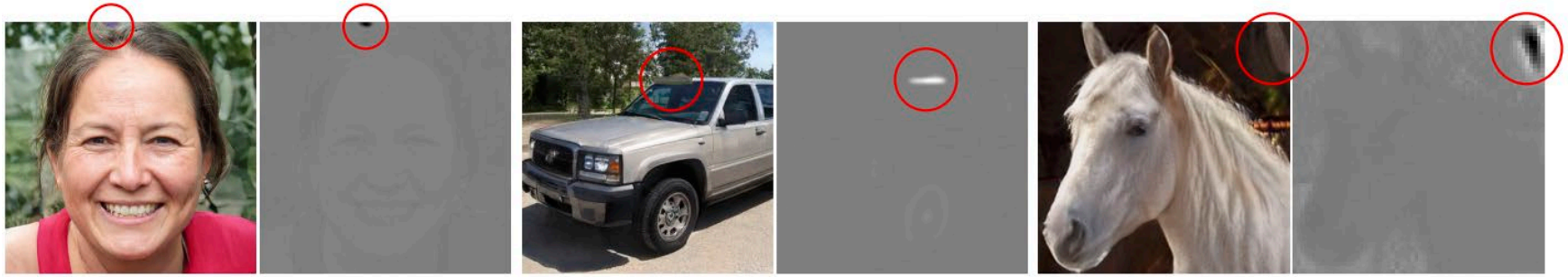


Figure 1. Instance normalization causes water droplet -like artifacts in StyleGAN images. These are not always obvious in the generated images, but if we look at the activations inside the generator network, the problem is always there, in all feature maps starting from the 64x64 resolution. It is a systemic problem that plagues all StyleGAN images.

Figure 6. Progressive growing leads to "phase" artifacts. In this example the teeth do not follow the pose but stay aligned to the camera, as indicated by the blue line.

T. Karras et al. Analyzing and Improving the Image Quality of StyleGAN. CVPR 2020

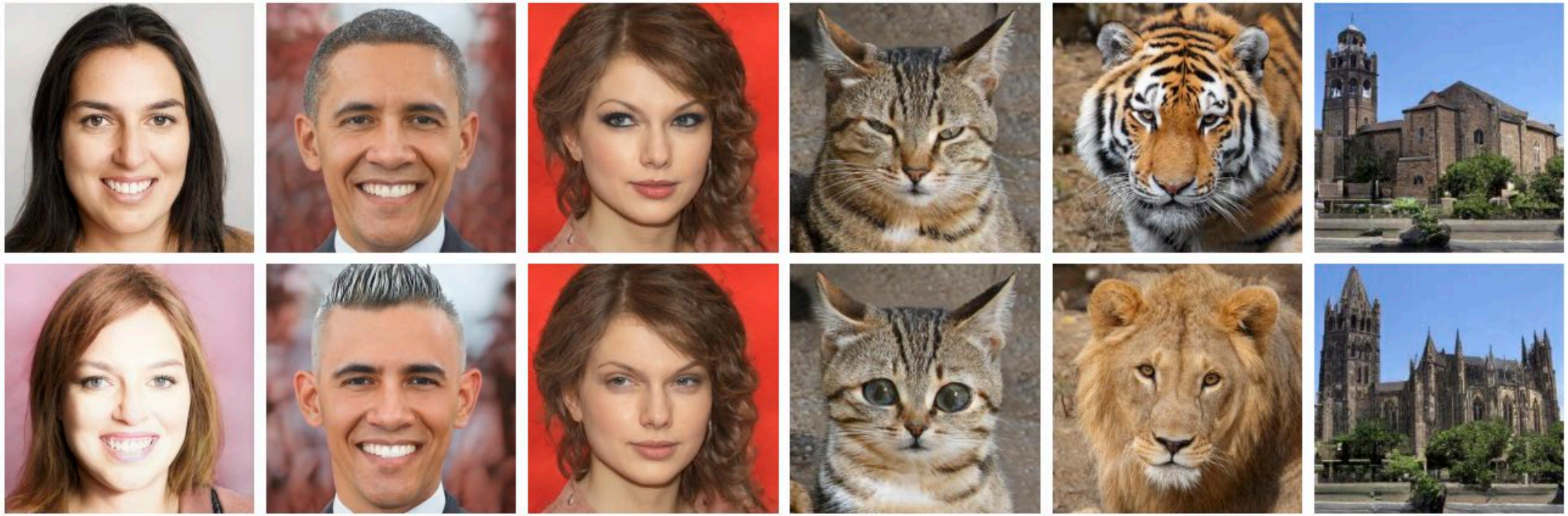
GAN editing

- Supervised training to find latent space directions corresponding to pose, smile, age, gender, eyeglasses

Y. Shen, J. Gu, X. Tang, B. Zhou. Interpreting the Latent Space of GANs for Semantic Face Editing. CVPR 2020

StyleCLIP

- Combine powerful recent image-text embedding technique (CLIP) with pre-trained StyleGAN for text-based image editing



O. Patashnik et al.. StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery. arXiv 2021

Diffusion models

https://www.nytimes.com/2023/04/08/technology/ai-photos-pope-francis.html

Denoising diffusion probabilistic models (DDPMs)

Denoising Diffusion Probabilistic Models

Jonathan Ho
UC Berkeley
jonathanho@berkeley.edu

Ajay Jain
UC Berkeley
ajayj@berkeley.edu

Pieter Abbeel
UC Berkeley
pabbeel@cs.berkeley.edu

Abstract

We present high quality image synthesis results using diffusion probabilistic models, a class of latent variable models inspired by considerations from nonequilibrium thermodynamics. Our best results are obtained by training on a weighted variational bound designed according to a novel connection between diffusion probabilistic models and denoising score matching with Langevin dynamics, and our models naturally admit a progressive lossy decompression scheme that can be interpreted as a generalization of autoregressive decoding. On the unconditional CIFAR10 dataset, we obtain an Inception score of 9.46 and a state-of-the-art FID score of 3.17. On 256x256 LSUN, we obtain sample quality similar to ProgressiveGAN. Our implementation is available at https://github.com/hojonathanho/diffusion.

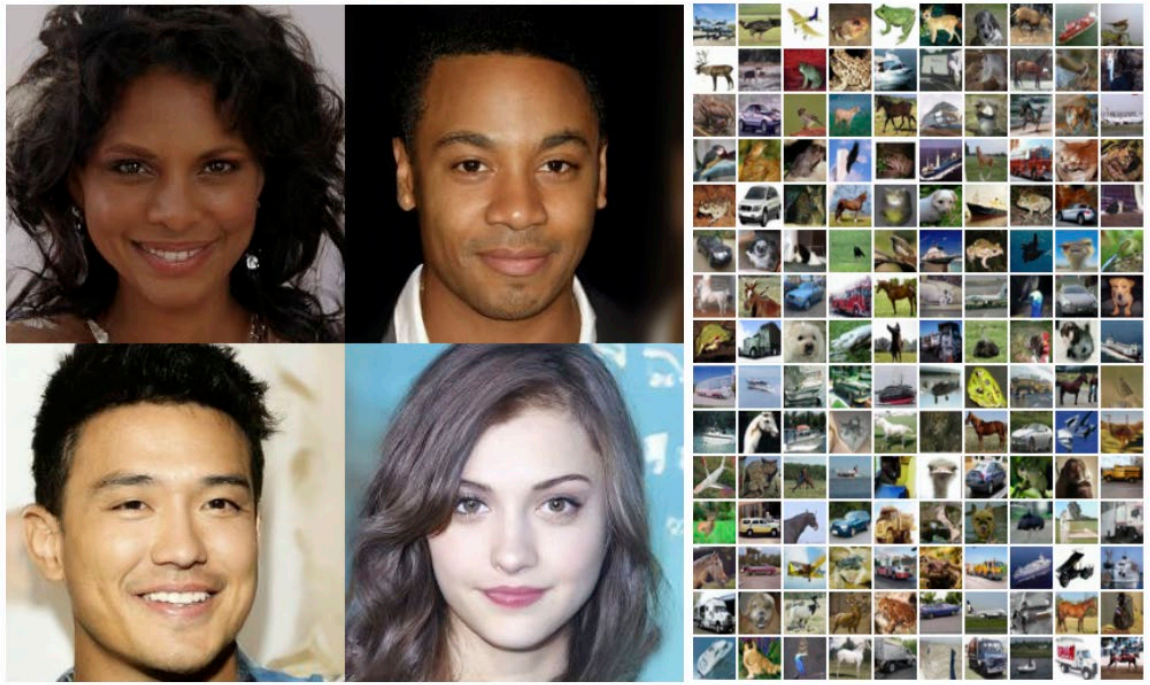
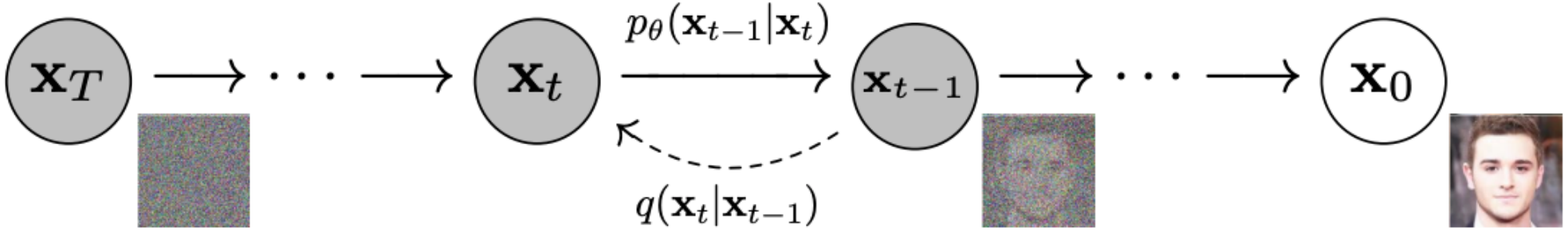


Figure 1: Generated samples on CelebA-HQ 256 × 256 (left) and unconditional CIFAR10 (right)

J. Ho et al. Denoising diffusion probabilistic models. NeurIPS 2020

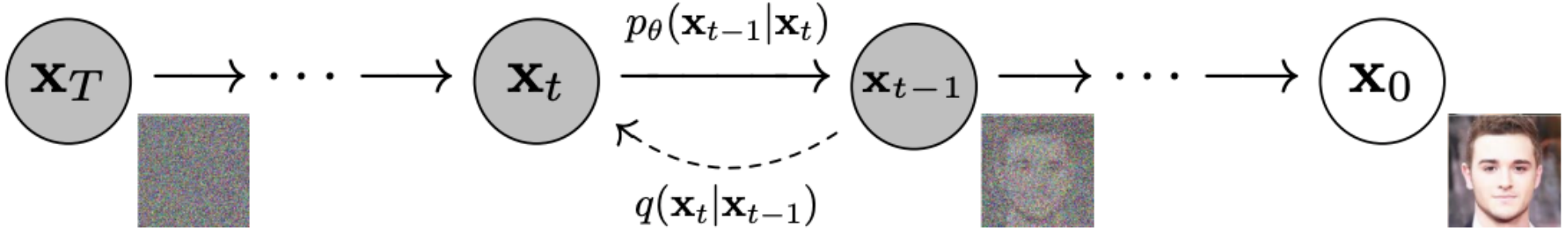
DDPMs: Basic idea



Unconditional CIFAR10 sample generation

J. Ho et al. Denoising diffusion probabilistic models. NeurIPS 2020
Blog introduction: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/
CVPR 2022 tutorial

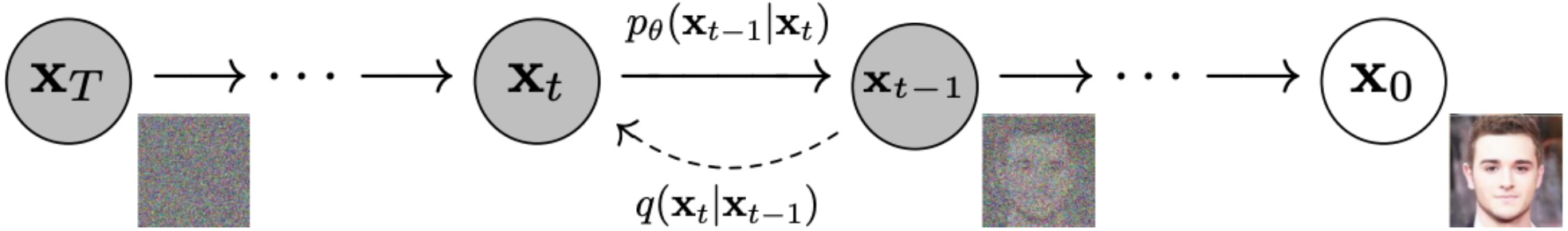
DDPMs: Basic idea



- *Forward process* q turns images into Gaussian noise
- *Reverse process* p turns noise into images
- Provided the increments of t are small enough, $p_\theta(x_{t-1}|x_t)$ is Gaussian and we can train a neural network to estimate the mean of x_{t-1} given x_t

J. Ho et al. Denoising diffusion probabilistic models. NeurIPS 2020

DDPMs: Basic idea



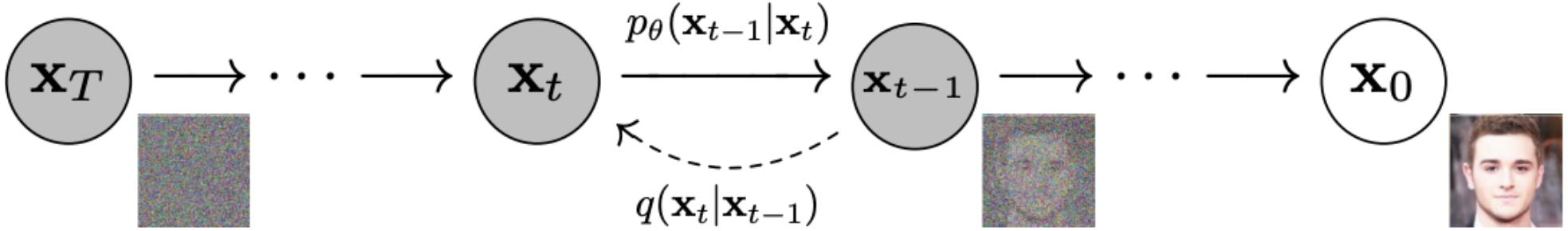
Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\quad x_t \quad , t) \right\|^2$
6: **until** converged

- $\epsilon_\theta(x_t, t)$ is the predicted noise component of image x_t given noise level t
- Network parameters θ are updated to reduce L2 error between actual noise ϵ and predicted noise $\epsilon_\theta(x_t, t)$

J. Ho et al. Denoising diffusion probabilistic models. NeurIPS 2020

DDPMs: Basic idea



Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

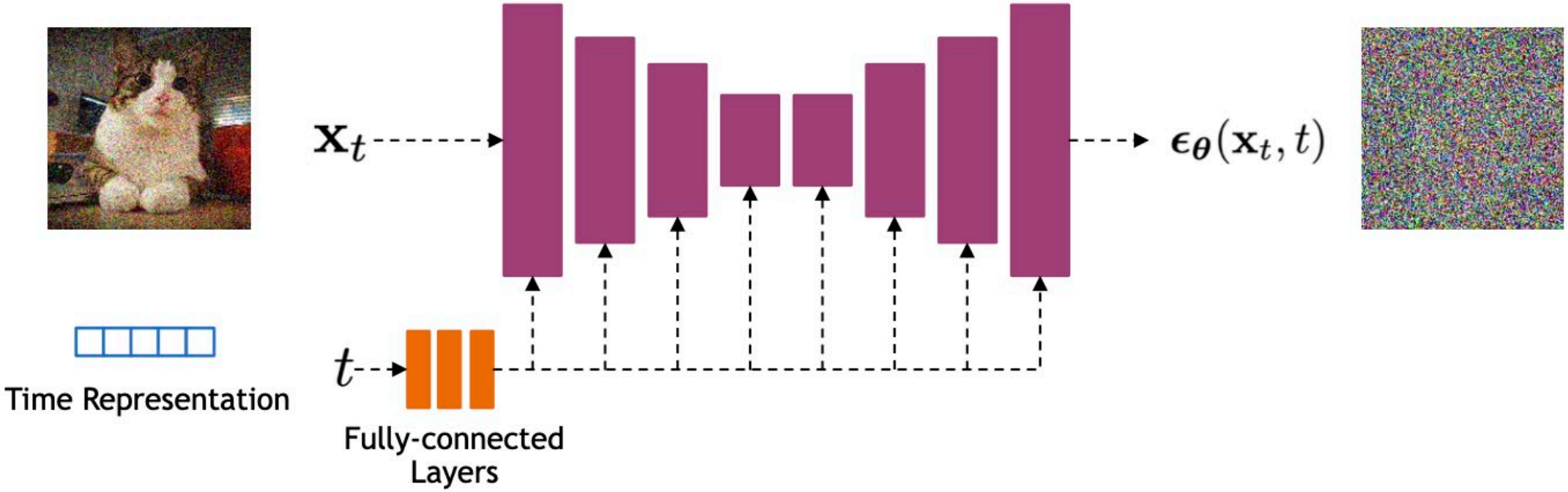
Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

J. Ho et al. Denoising diffusion probabilistic models. NeurIPS 2020

DDPMs: Implementation

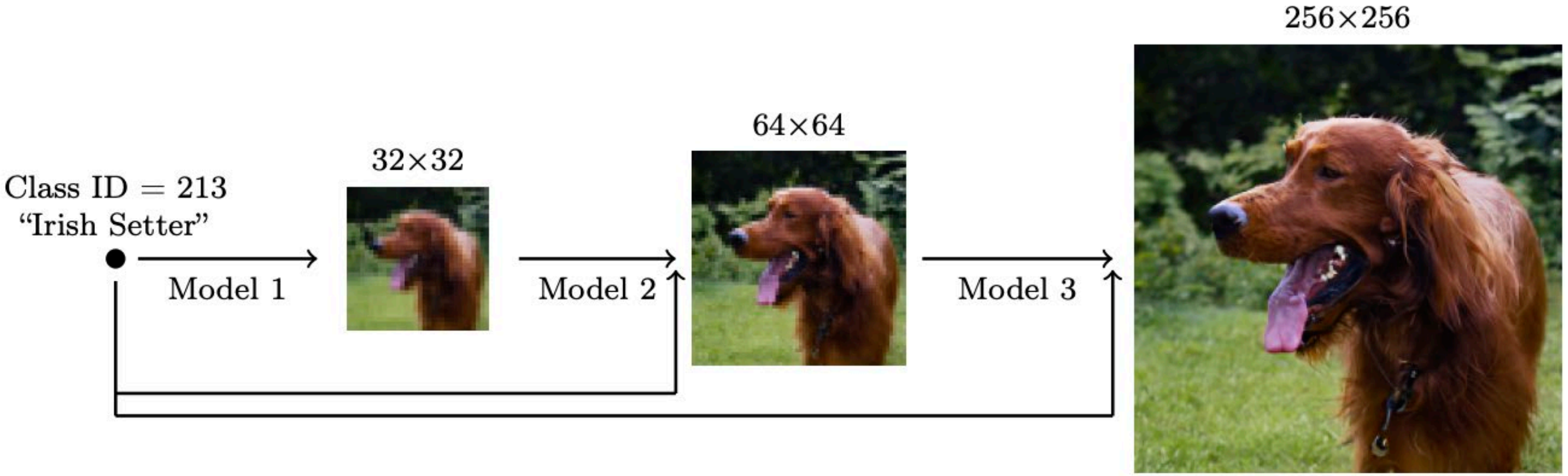
- U-Net architectures are typically used to represent $\epsilon_\theta(x_t, t)$
  - Bells and whistles: residual blocks, self-attention



- Time is encoded using sinusoidal positional embeddings or random Fourier features, fed into the U-Net using addition or adaptive normalization

Source: CVPR 2020 DM tutorial

Efficient sampling at high resolutions: Cascaded generation



- In practice, data augmentation for inputs to upsampling models is crucial (esp. adding Gaussian noise or early stopping for base model)

J. Ho et al. Cascaded Diffusion Models for High Fidelity Image Generation. JMLR 2022

Class-conditioned DDPMs

- “We can sample with as few as 25 forward passes while maintaining FIDs comparable to BigGAN”

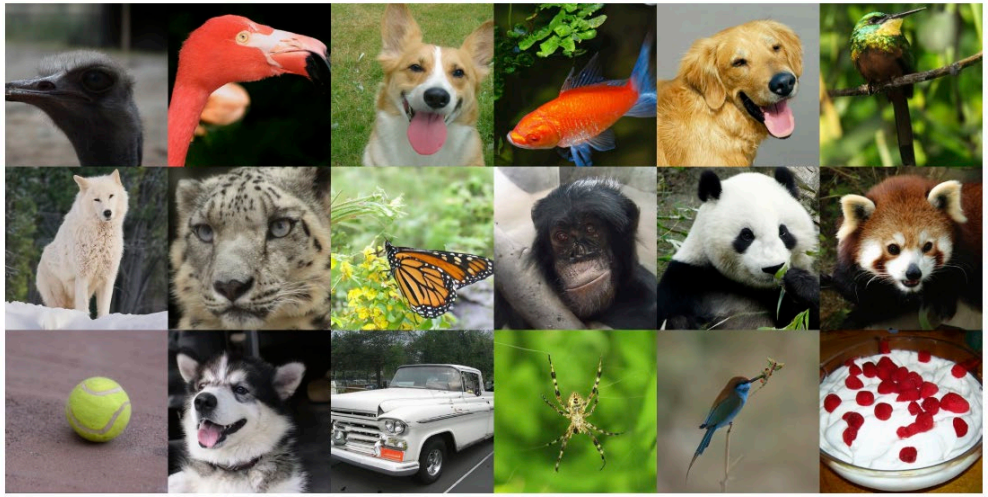


Figure 1: Selected samples from our best ImageNet 512×512 model (FID 3.85)

Abstract

We show that diffusion models can achieve image sample quality superior to the current state-of-the-art generative models. We achieve this on unconditional image synthesis by finding a better architecture through a series of ablations. For conditional image synthesis, we further improve sample quality with classifier guidance: a simple, compute-efficient method for trading off diversity for fidelity using gradients from a classifier. We achieve an FID of 2.97 on ImageNet 128×128, 4.59 on ImageNet 256×256, and 7.72 on ImageNet 512×512, and we match BigGAN-deep even with as few as 25 forward passes per sample, all while maintaining better coverage of the distribution. Finally, we find that classifier guidance combines well with upsampling diffusion models, further improving FID to 3.94 on ImageNet 256×256 and 3.85 on ImageNet 512×512. We release our code at `https://github.com/openai/guided-diffusion`.

P. Dhariwal and A. Nichol. Diffusion Models Beat GANs on Image Synthesis. NeurIPS 2021

Classifier guidance

- We can sample from the class-conditional density $q(x_t|c)$ with the help of a pre-trained classifier $P(c|x_t)$
- Bayes rule:

$$q(x_t|c) \propto P(c|x_t)q(x_t)$$

$$\log q(x_t|c) = \log P(c|x_t) + \log q(x_t) + \text{const.}$$

$$\nabla_{x_t} \log q(x_t|c) = \nabla_{x_t} \log P(c|x_t) + \nabla_{x_t} \log q(x_t)$$

conditional score function

obtained from classifier output

unconditional *score function* (pre-trained)

- To sample from class c, steer sample in the modified direction $\nabla_{x_t}[\log q(x_t) + w \log P(c|x_t)]$

Classifier-free guidance

- Instead of training an additional classifier, get an "implicit classifier" by jointly training a conditional and unconditional diffusion model: $P(c|x_t) \propto q(x_t|c)/q(x_t)$
- Both $q(x_t|c)$ and $q(x_t)$ are represented using the same network, trained by dropping out c with some probability (corresponding to the unconditional case)
- The modified score function corresponding to this implicit classifier is

$$\nabla_{x_t}[\log q(x_t) + w \log P(c|x_t)]$$
$$= \nabla_{x_t}[\log q(x_t) + w(\log q(x_t|c) - \log q(x_t))]$$

Sample is steered away from the unconditional distribution in the direction of the conditional one

J. Ho and T. Salimans. Classifier-Free Diffusion Guidance. arXIv 2021

Classifier-free guidance

Figure 1: Classifier-free guidance on the malamute class for a 64x64 ImageNet diffusion model. Left to right: increasing amounts of classifier-free guidance, starting from non-guided samples on the left.

J. Ho and T. Salimans. Classifier-Free Diffusion Guidance. arXIv 2021

Text-guided diffusion

- Instead of a class label, c can be an encoded text prompt, injected into the U-Net using *cross-attention*

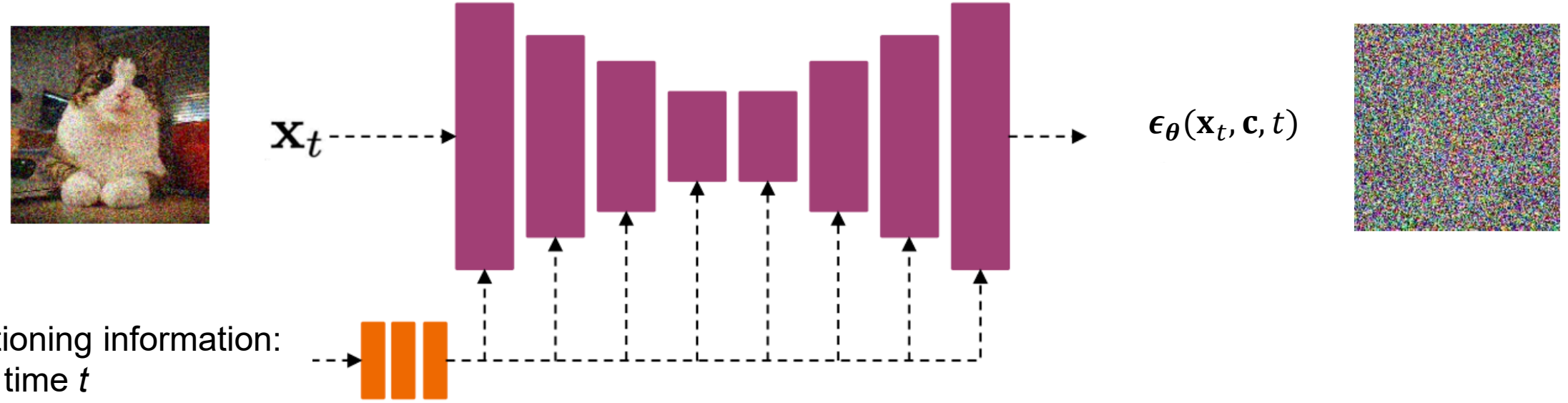


Image source

Text-guided diffusion

- Instead of a class label, c can be an encoded text prompt, injected into the U-Net using *cross-attention*

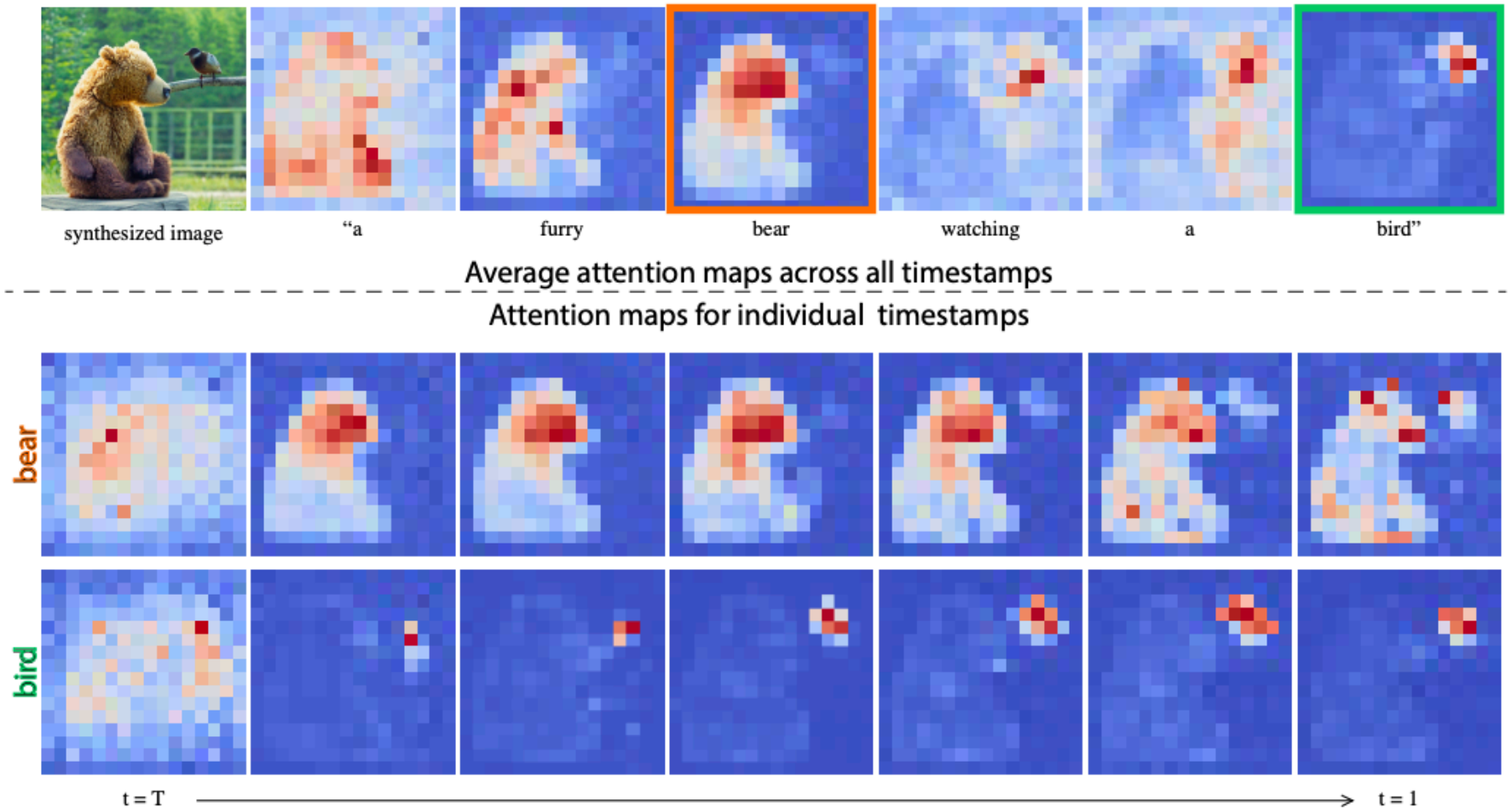


Image source

Text-guided diffusion

- Instead of a class label, c can be an encoded text prompt, injected into the U-Net using *cross-attention*
- Classifier-free guidance works the same way as before, by training both conditional and unconditional models using text dropout
- CLIP guidance: steer samples in the direction of $\nabla_{x_t} \mathrm{CLIP}(x_t, c)$
- Note: both classifier and CLIP must be *noise-aware* (trained on noised images)

OpenAI GLIDE

- Text-to-image generation using classifier-free or CLIP guidance, generate at 64x64, upsample to 256x256

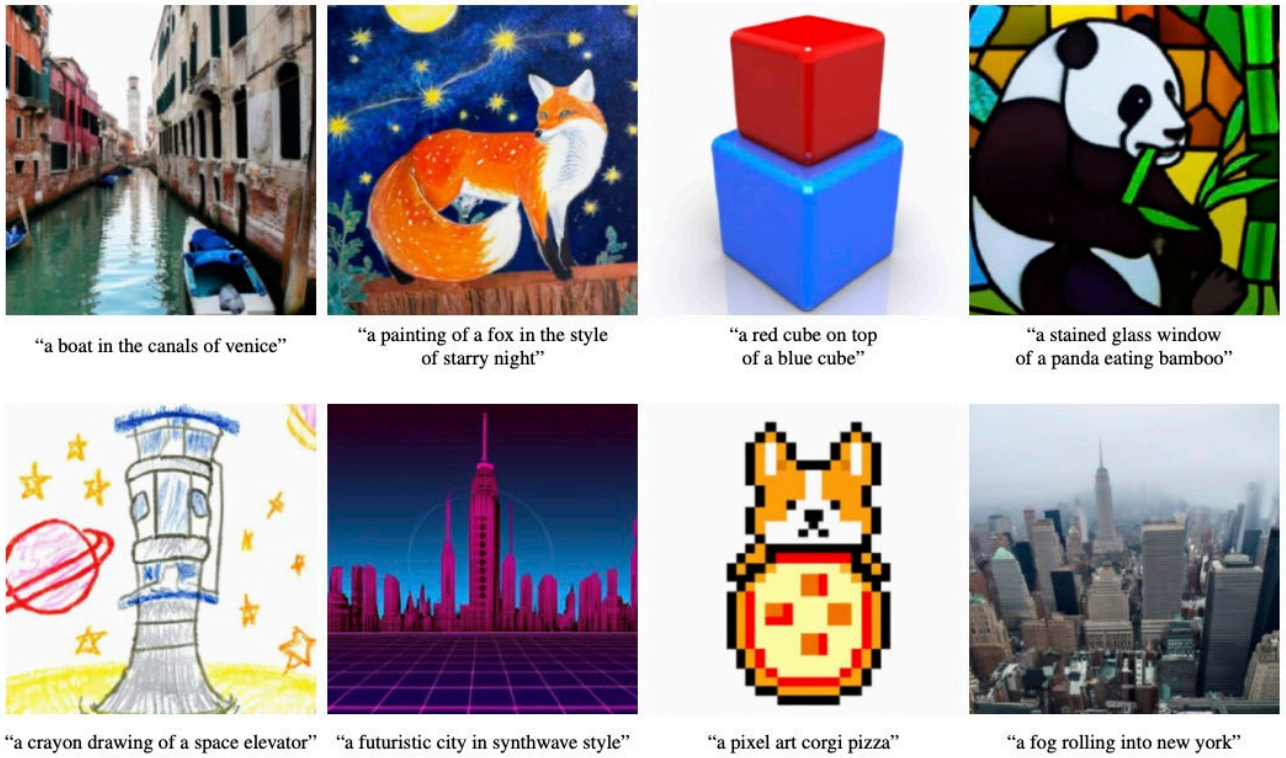


Figure 1. Selected samples from GLIDE using classifier-free guidance. We observe that our model can produce photorealistic images with shadows and reflections, can compose multiple concepts in the correct way, and can produce artistic renderings of novel concepts. For random sample grids, see Figure 17 and 18.

A. Nichol et al. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. ICML 2022

OpenAI GLIDE

- Text-to-image generation using classifier-free or CLIP guidance, generate at 64x64, upsample to 256x256
  - 64x64 generator: 2.3B parameters
  - Upsampler: 1.5B parameters
  - Text encoder: 1.2B parameters
- Dataset: same as for DALL-E, 250M image-text pairs from the Internet

A. Nichol et al. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. ICML 2022

OpenAI GLIDE

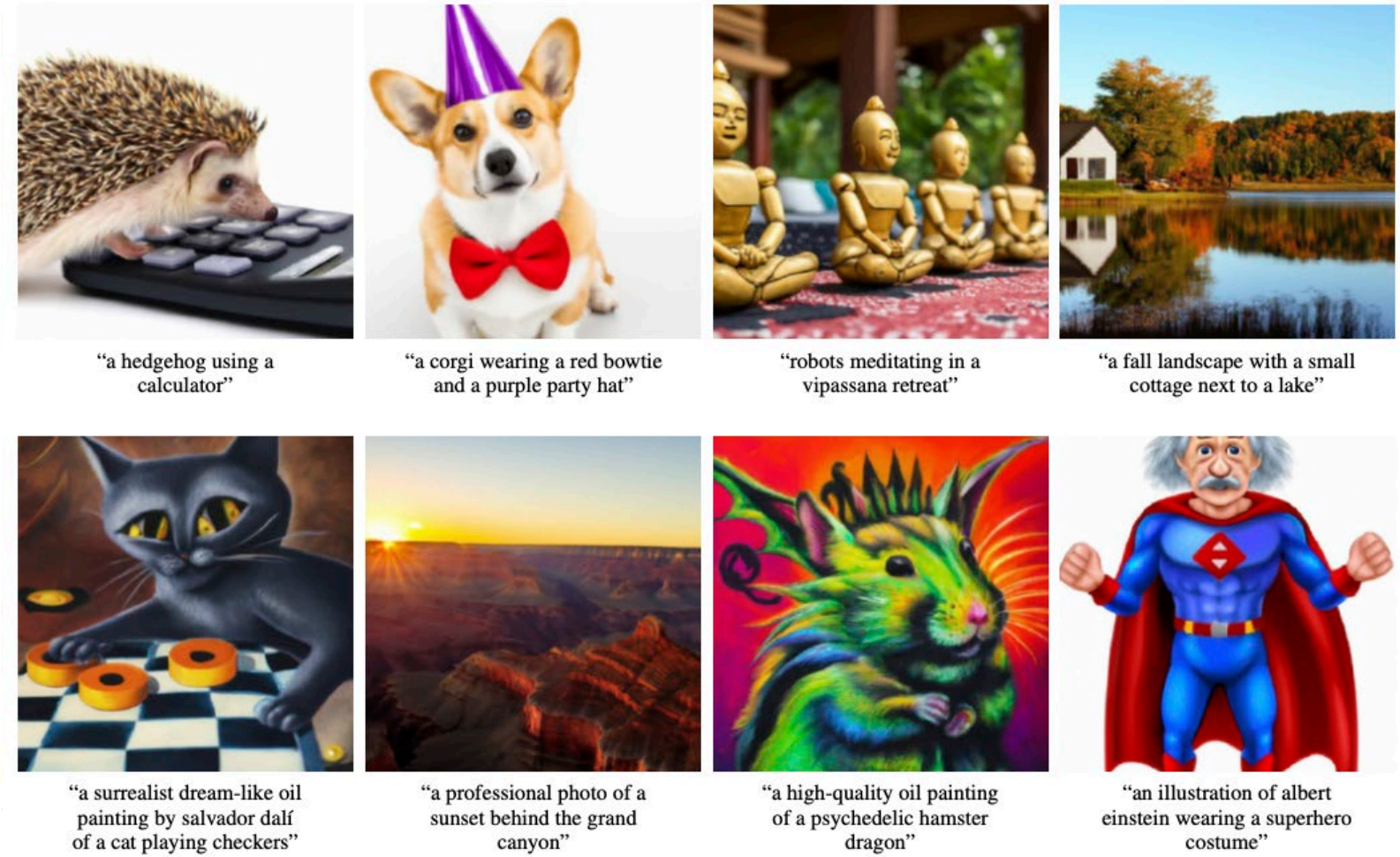
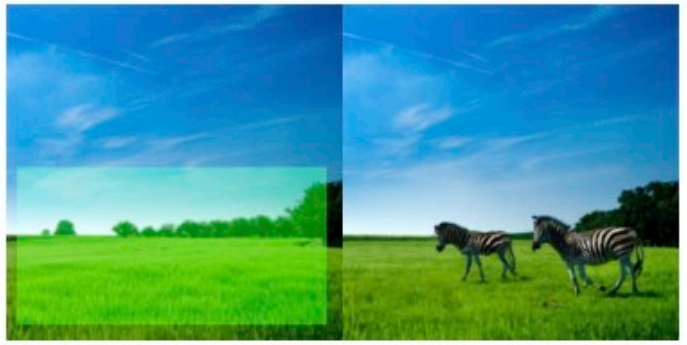


Figure 1. Selected samples from GLIDE using classifier-free guidance. We observe that our model can produce photorealistic images with shadows and reflections, can compose multiple concepts in the correct way, and can produce artistic renderings of novel concepts. For random sample grids, see Figure 17 and 18.

A. Nichol et al. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. ICML 2022

OpenAI GLIDE



"zebras roaming in the field"

"a girl hugging a corgi on a pedestal"

"a man with red hair"

"a vase of flowers"

"an old car in a snowy forest"

"a man wearing a white hat"

DALL-E 2

A. Ramesh et al. Hierarchical text-conditional image generation with CLIP latents. Preprint 2022

DALL-E 2

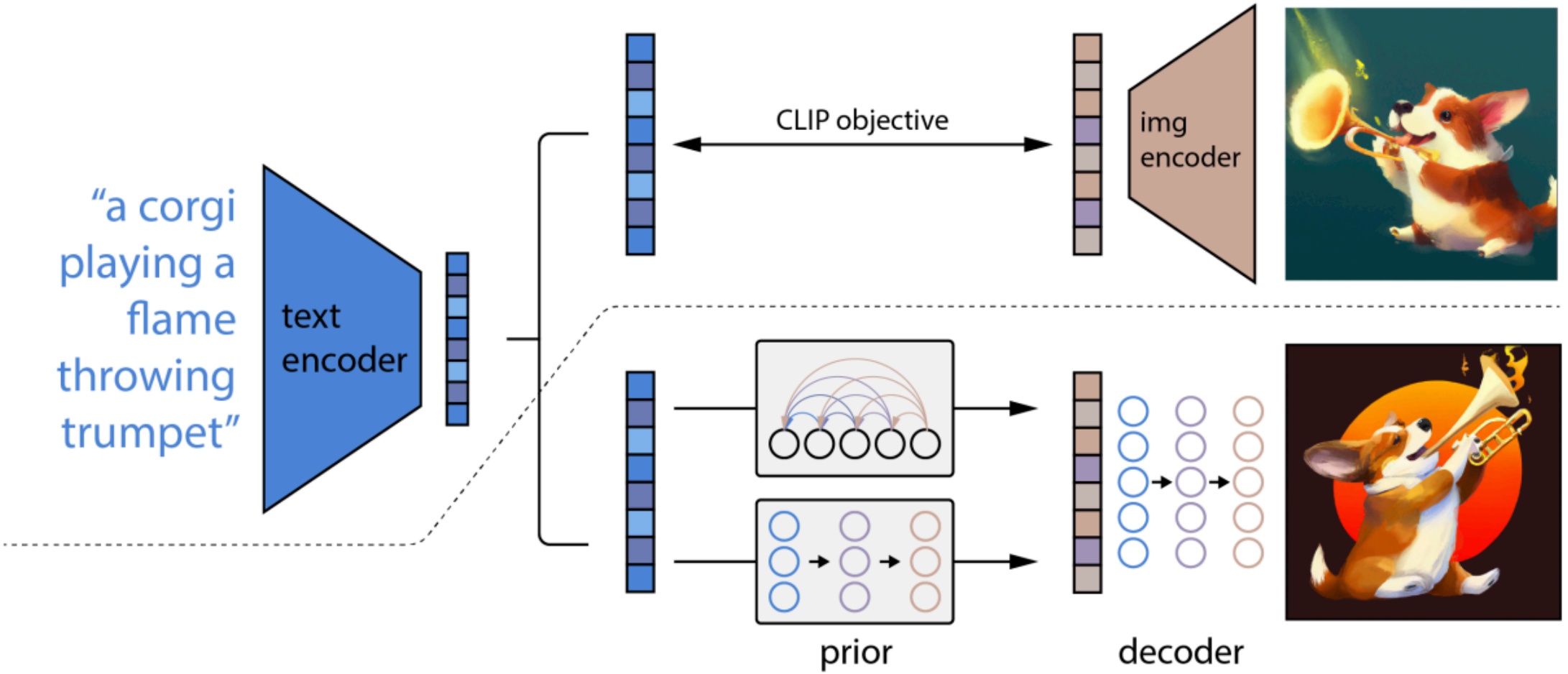
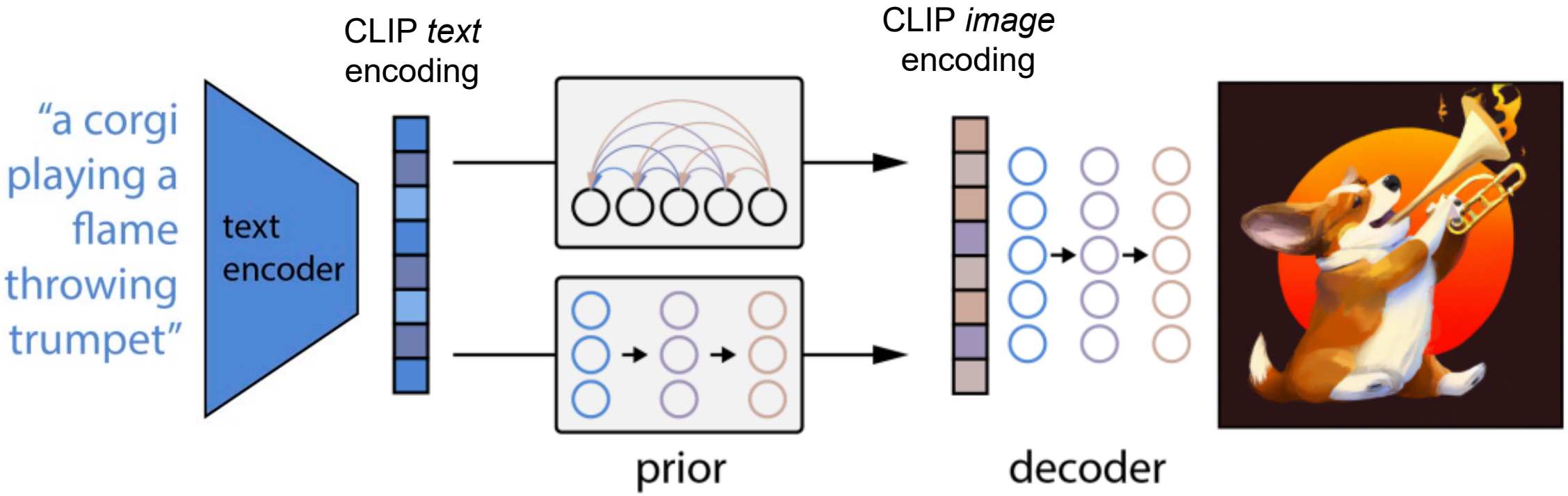


Figure 2: A high-level overview of unCLIP. Above the dotted line, we depict the CLIP training process, through which we learn a joint representation space for text and images. Below the dotted line, we depict our text-to-image generation process: a CLIP text embedding is first fed to an autoregressive or diffusion prior to produce an image embedding, and then this embedding is used to condition a diffusion decoder which produces a final image. Note that the CLIP model is frozen during training of the prior and decoder.

DALL-E 2



Generative model to produce CLIP *image* encoding given CLIP *text* encoding

Diffusion model (GLIDE) conditioned on CLIP image embedding and text prompt

Generate at 64x64, upsample to 256x256, then upsample to 1024x1024

DALL-E 2: Results

Figure 19: Random samples from unCLIP for prompt "A close up of a handpalm with leaves growing from it."

Figure 18: Random samples from unCLIP for prompt "Vibrant portrait painting of Salvador Dali with a robotic half face"

DALL-E 2: Results

Figure 3: Variations of an input image by encoding with CLIP and then decoding with a diffusion model. The variations preserve both semantic information like presence of a clock in the painting and the overlapping strokes in the logo, as well as stylistic elements like the surrealism in the painting and the color gradients in the logo, while varying the non-essential details.

DALL-E 2: Results

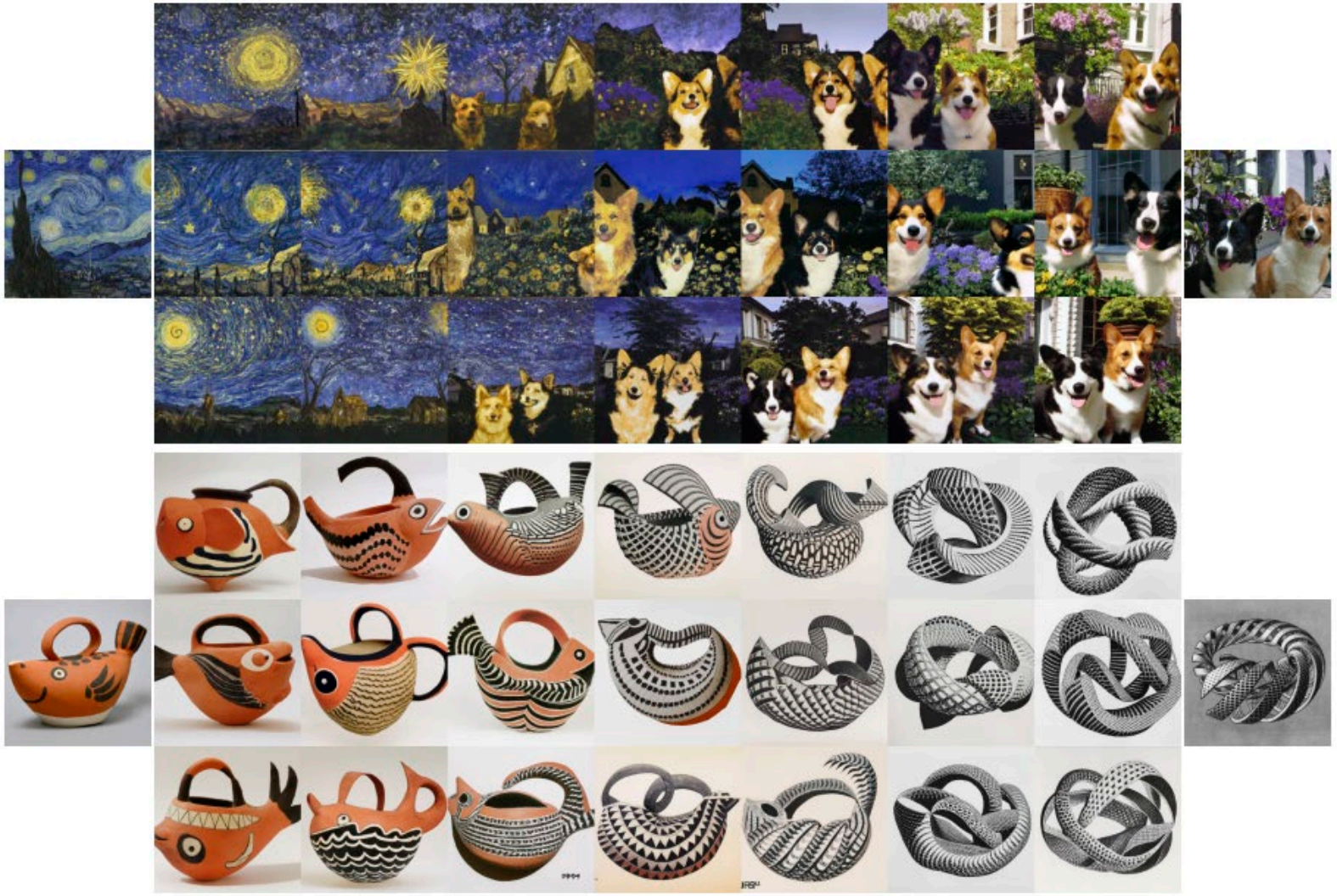
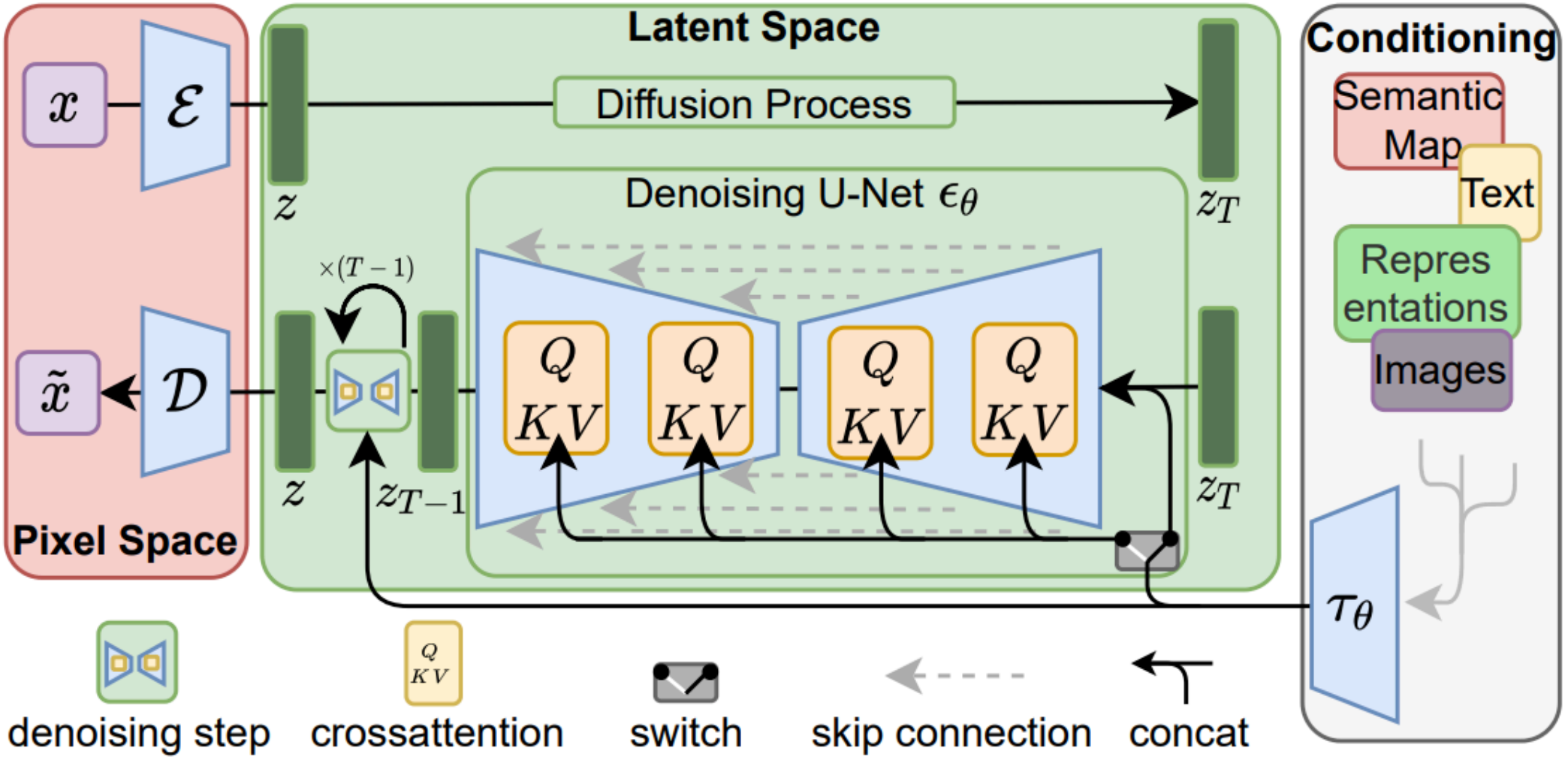


Figure 4: Variations between two images by interpolating their CLIP image embedding and then decoding with a diffusion model. We fix the decoder seed across each row. The intermediate variations naturally blend the content and style from both input images.

DALL-E 2: Limitations

Figure 15: Reconstructions from the decoder for difficult binding problems. We find that the reconstructions mix up objects and attributes. In the first two examples, the model mixes up the color of two objects. In the rightmost example, the model does not reliably reconstruct the relative size of two objects.

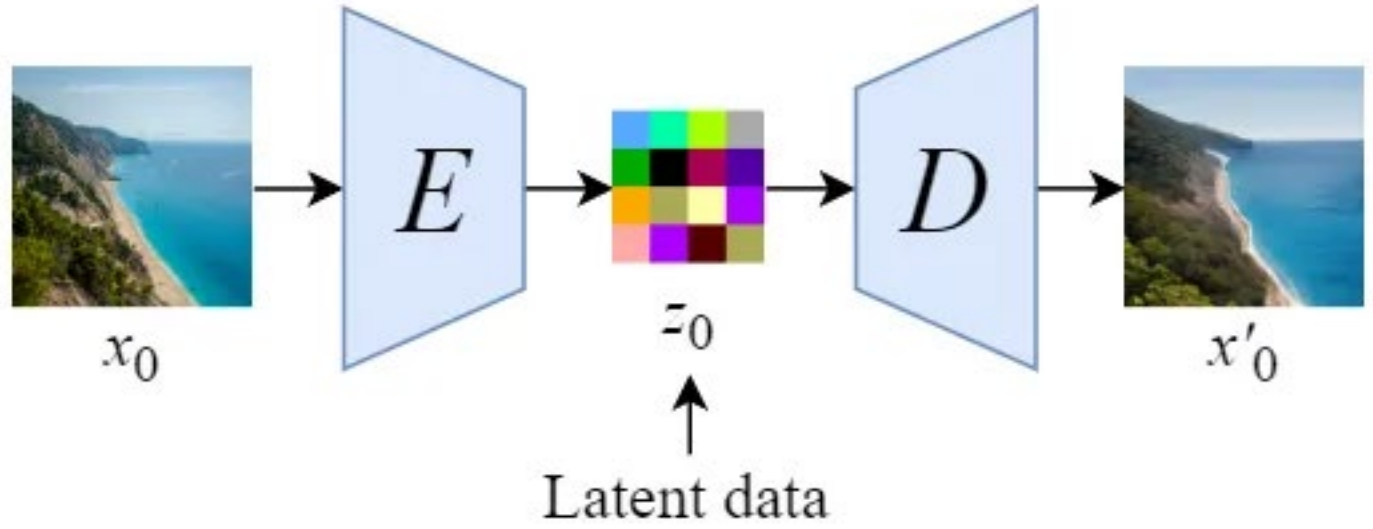
Latent diffusion model (basis of Stable Diffusion)



R. Rombach et al. [High-Resolution Image Synthesis with Latent Diffusion Models](). CVPR 2022

Latent diffusion model (basis of Stable Diffusion)

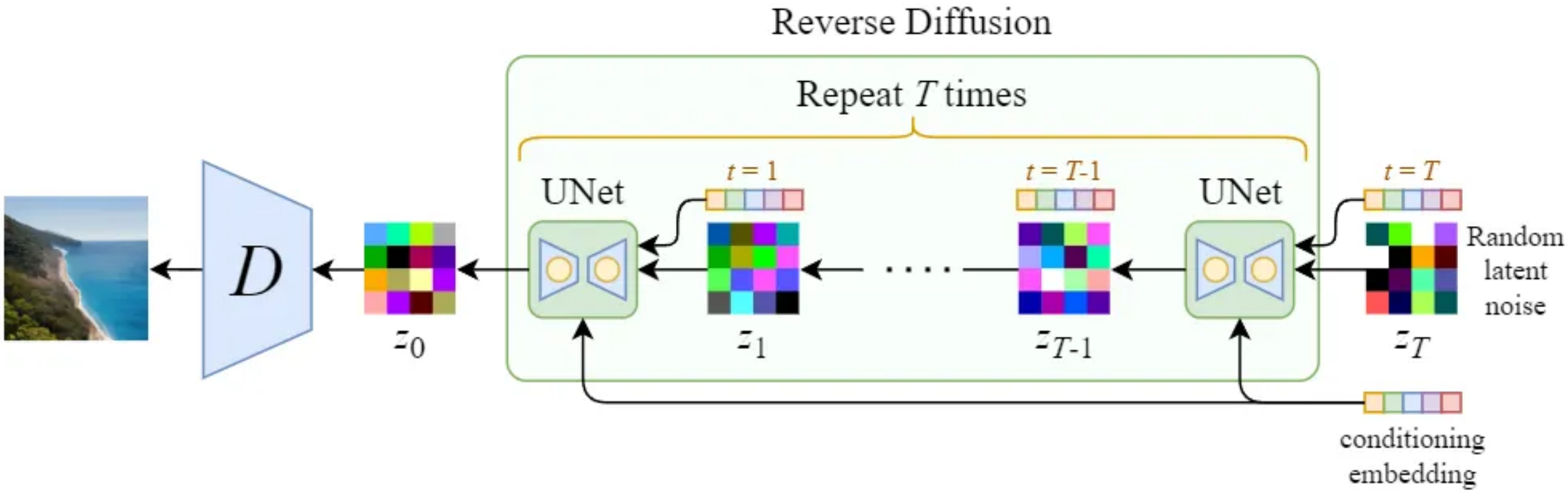
- Key idea: train a separate *encoder* and *decoder* to convert images to and from a lower-dimensional latent space, run conditional diffusion model in latent space



https://medium.com/@steinsfu/stable-diffusion-clearly-explained-ed008044e07e

Latent diffusion model (basis of Stable Diffusion)

- Key idea: train a separate *encoder* and *decoder* to convert images to and from a lower-dimensional latent space, run conditional diffusion model in latent space

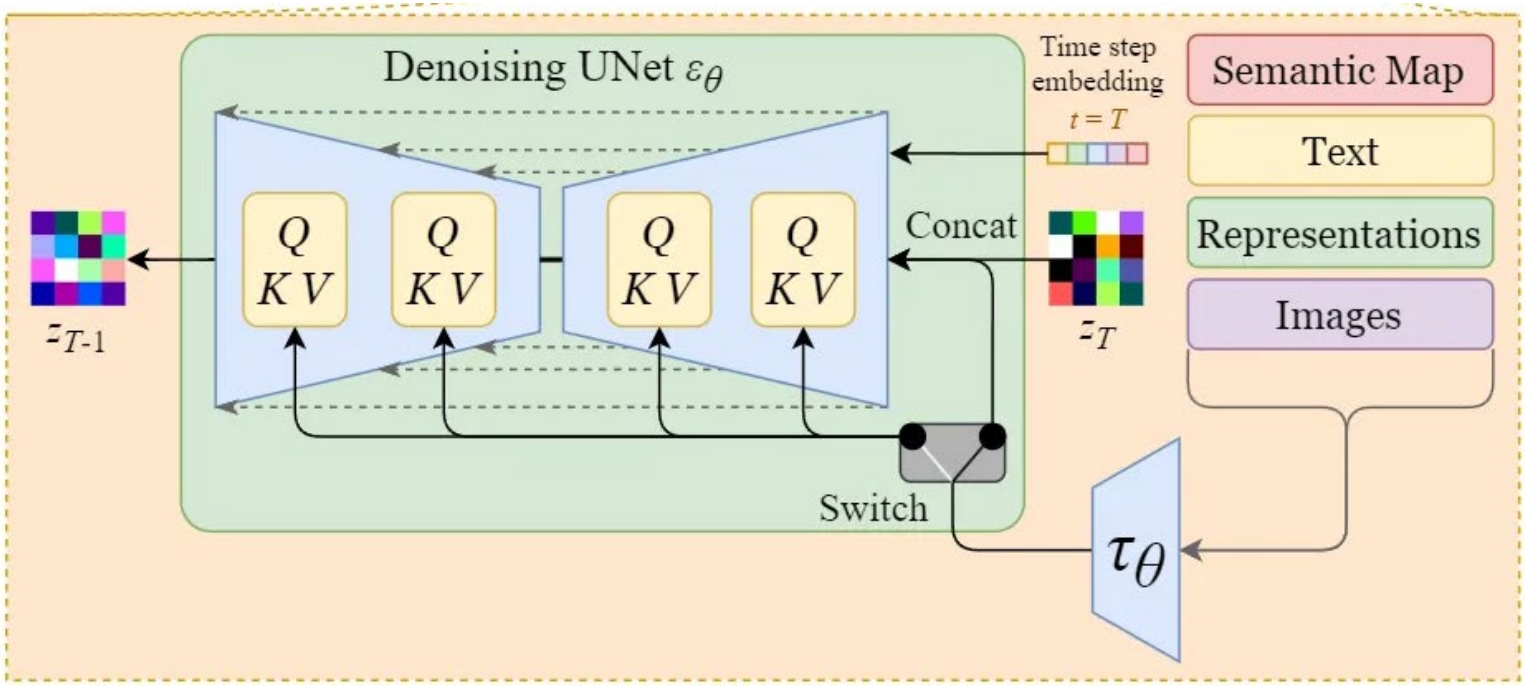


https://medium.com/@steinsfu/stable-diffusion-clearly-explained-ed008044e07e

Latent diffusion model (basis of Stable Diffusion)

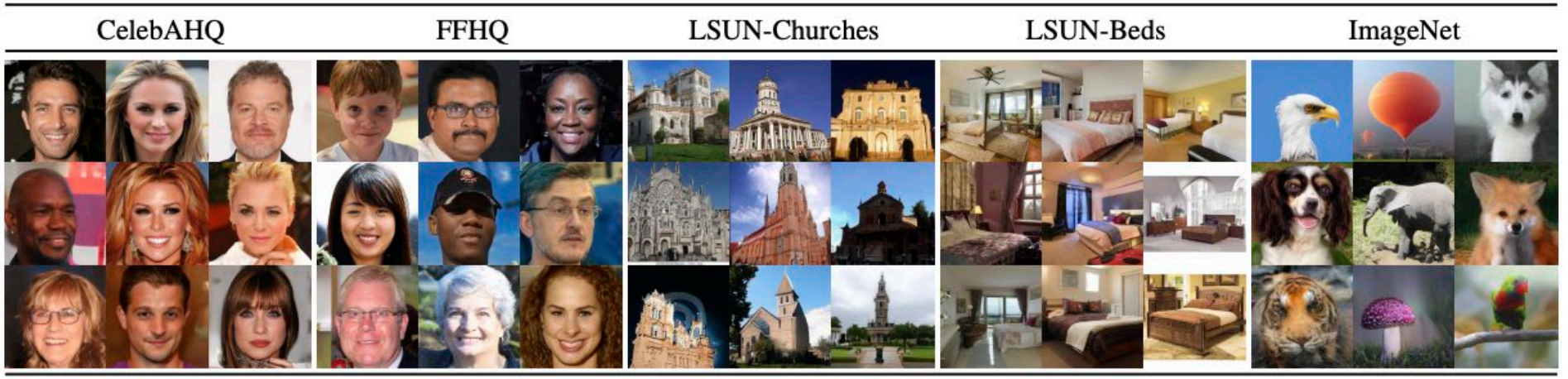
- Key idea: train a separate *encoder* and *decoder* to convert images to and from a lower-dimensional latent space, run conditional diffusion model in latent space

Close-up of U-Net: Conditioning information incorporated using cross-attention

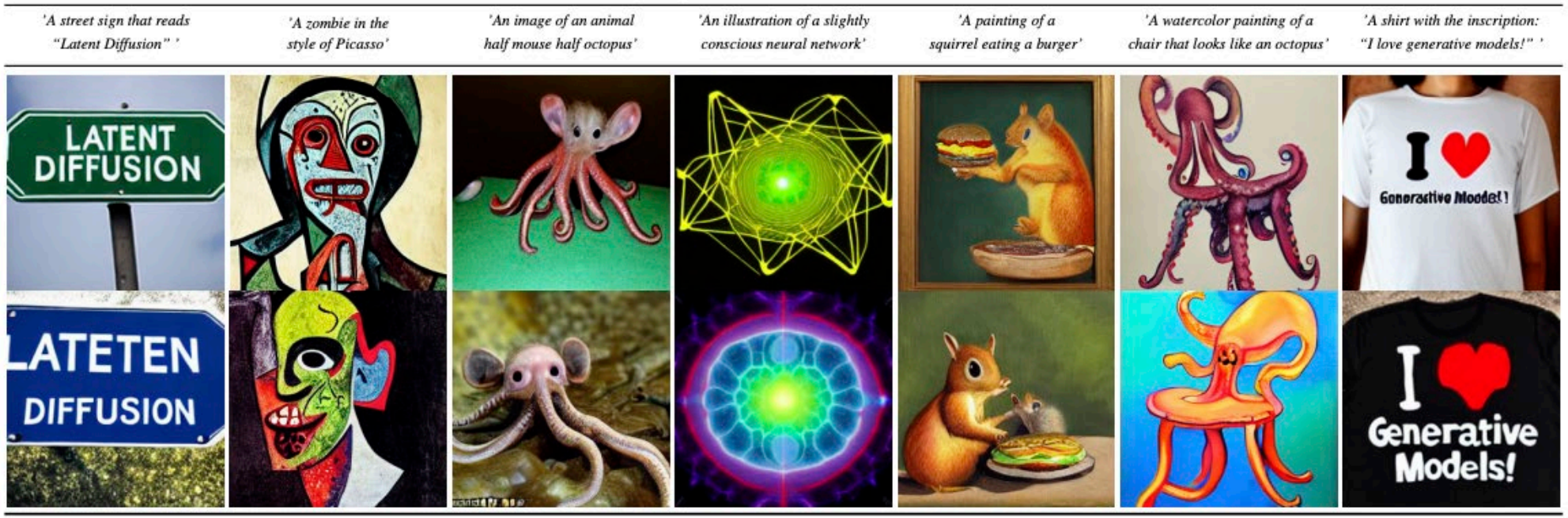


https://medium.com/@steinsfu/stable-diffusion-clearly-explained-ed008044e07e

Latent diffusion model (basis of Stable Diffusion)



Text-to-Image Synthesis on LAION. 1.45B Model.



Google Imagen (not public)

Sprouts in the shape of text 'Imagen' coming out of a fairytale book.

A photo of a Shiba Inu dog with a backpack riding a bike. It is wearing sunglasses and a beach hat.

A high contrast portrait of a very happy fuzzy panda dressed as a chef in a high end kitchen making dough. There is a painting of flowers on the wall behind him.

Teddy bears swimming at the Olympics 400m Butterfly event.

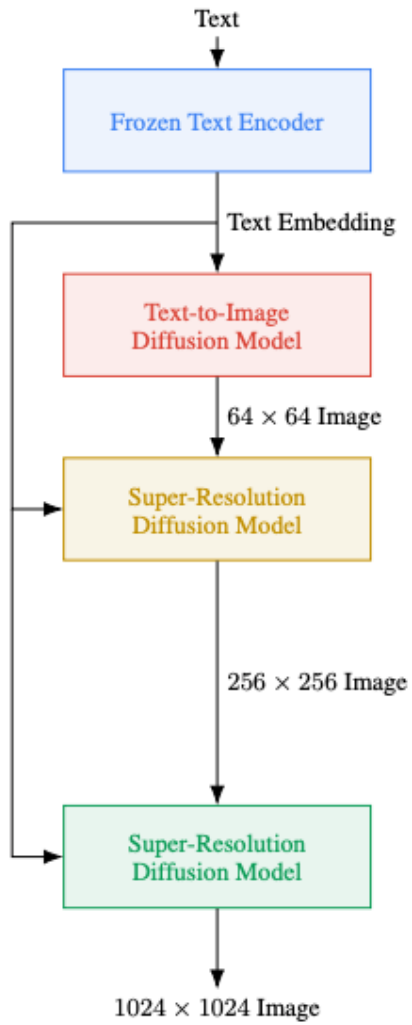
A cute corgi lives in a house made out of sushi.

A cute sloth holding a small treasure chest. A bright golden glow is coming from the chest.

C. Saharia et al. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. NeurIPS 2022

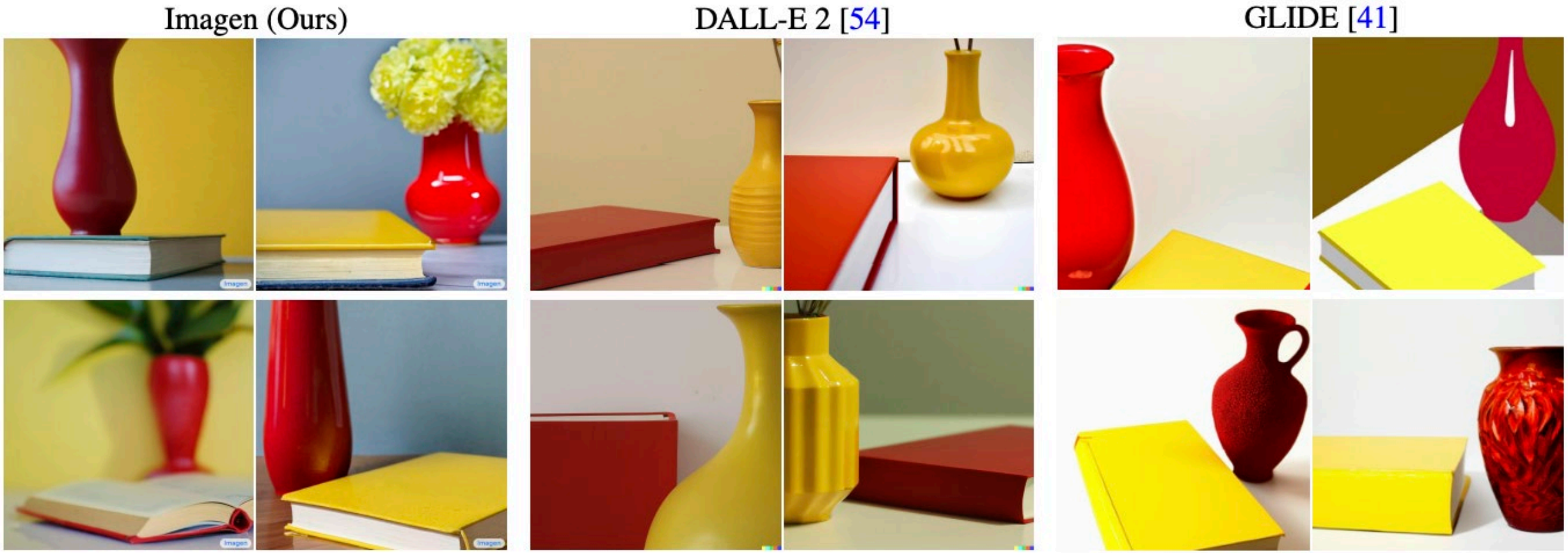
Google Imagen: Details

- Text encoder is a large language model (4.6B parameters) trained on text only
- Diffusion model to generate at 64x64, upsample to 256x256, then 1024x1024
  - Architecture: *efficient U-Net* (2B parameters): more parameters at lower resolutions, convolutions *after* downsampling and *before* upsampling
  - Classifier-free guidance with a *dynamic thresholding* technique, enabling good generation quality with high guidance weights
  - Training dataset: 460M image-text pairs (internally collected), 400M pairs from the LAION dataset



C. Saharia et al. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. NeurIPS 2022

Imagen vs. DALL-E 2 vs. GLIDE



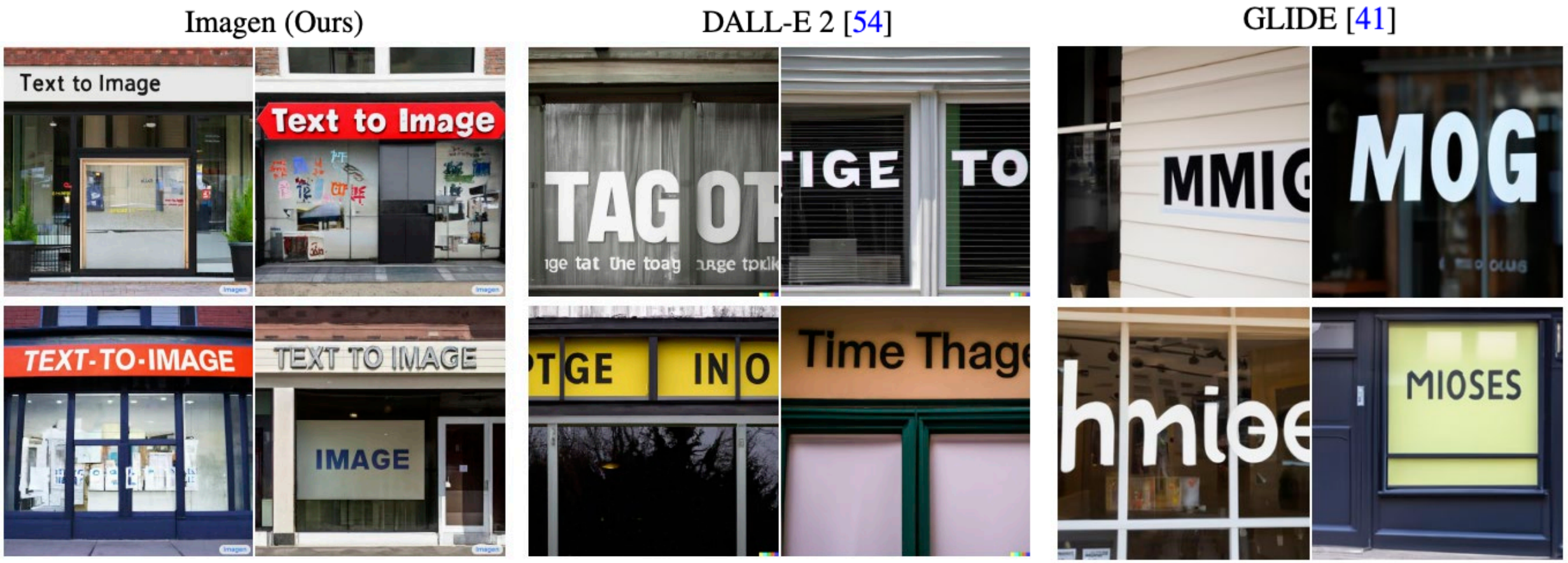
“A yellow book and a red vase”

Imagen vs. DALL-E 2 vs. GLIDE

“A black apple and a green backpack”

“We observe that GLIDE is better than DALL-E 2 in assigning the colors to the objects.”

Imagen vs. DALL-E 2 vs. GLIDE

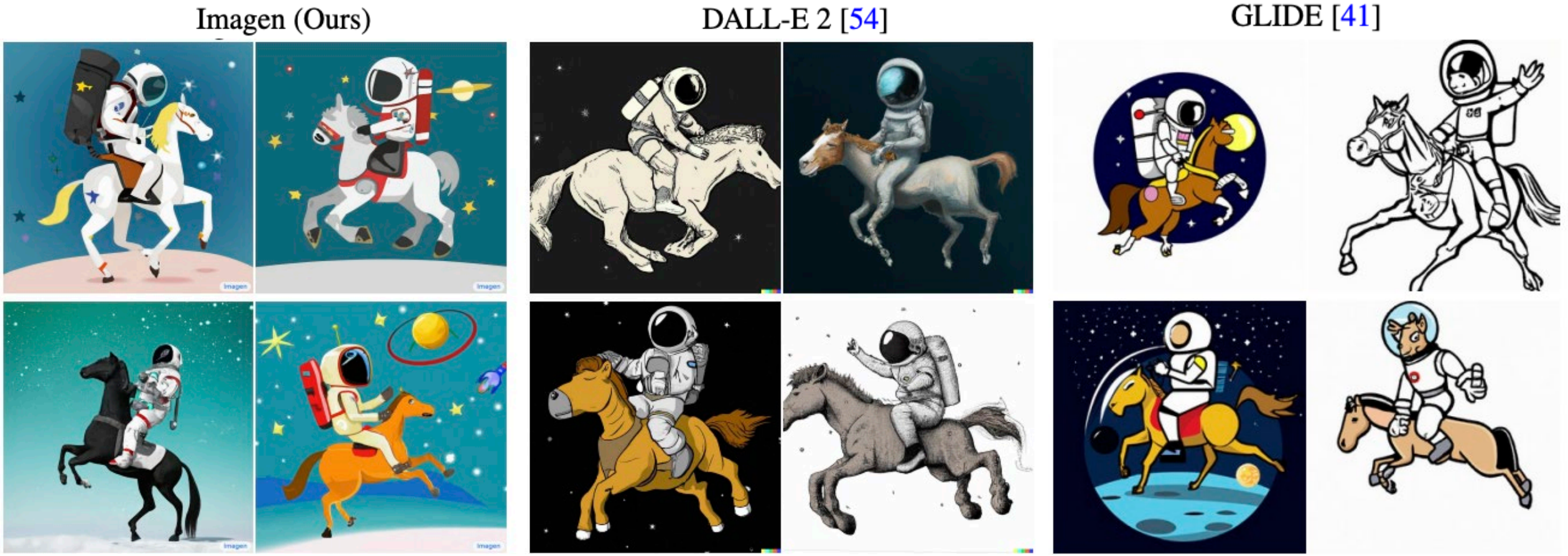


“A storefront with Text to Image written on it”

Imagen vs. DALL-E 2 vs. GLIDE

“A panda making latte art”

Imagen vs. DALL-E 2 vs. GLIDE



“A horse riding an astronaut”

InstructPix2Pix

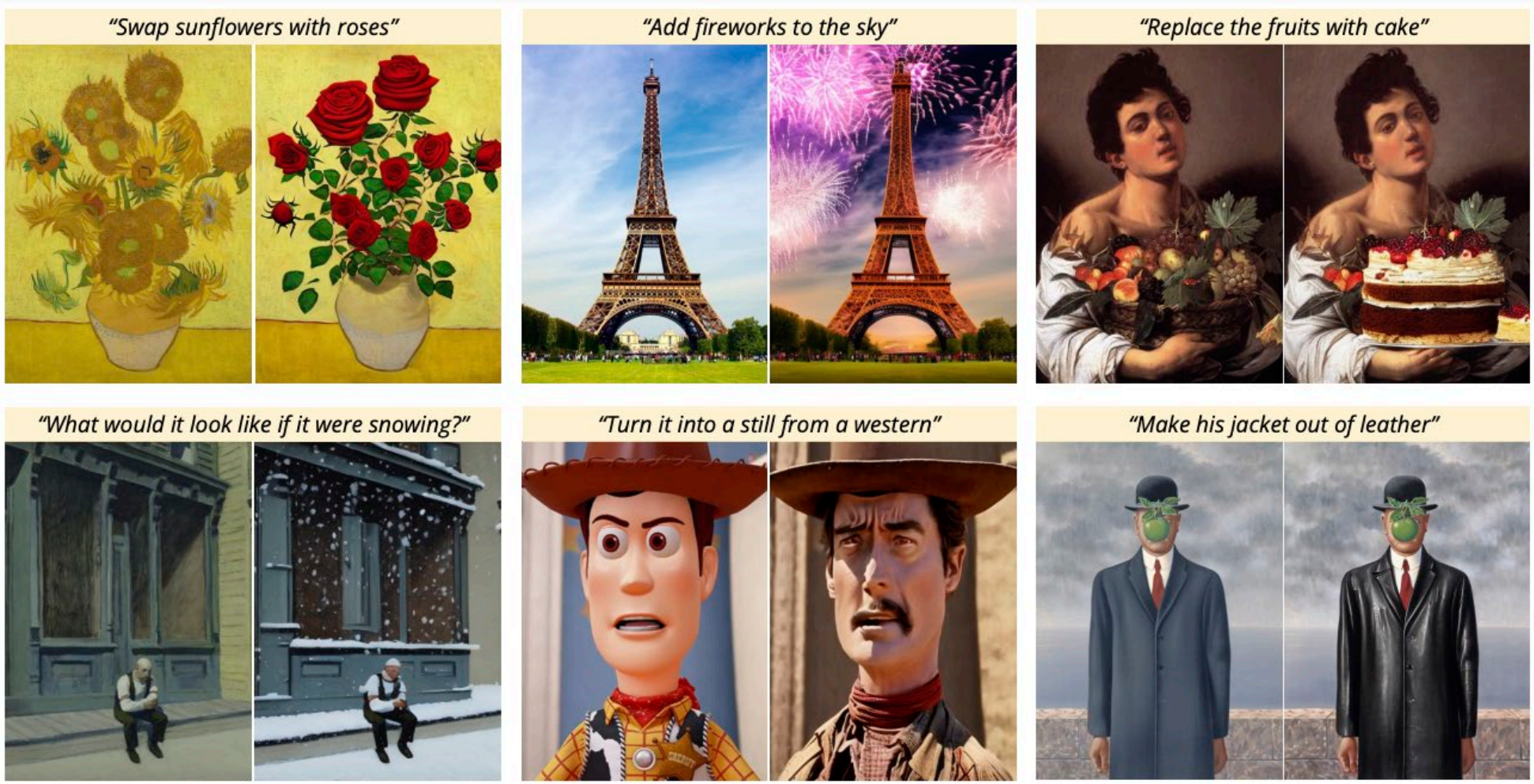


Figure 1. Given **an image** and **an instruction** for how to edit that image, our model performs the appropriate edit. Our model does not require full descriptions for the input or output image, and edits images in the forward pass without per-example inversion or fine-tuning.

T. Brooks, A. Holynski, and A. Efros. InstructPix2Pix: Learning to Follow Image Editing Instructions. CVPR 2023

InstructPix2Pix

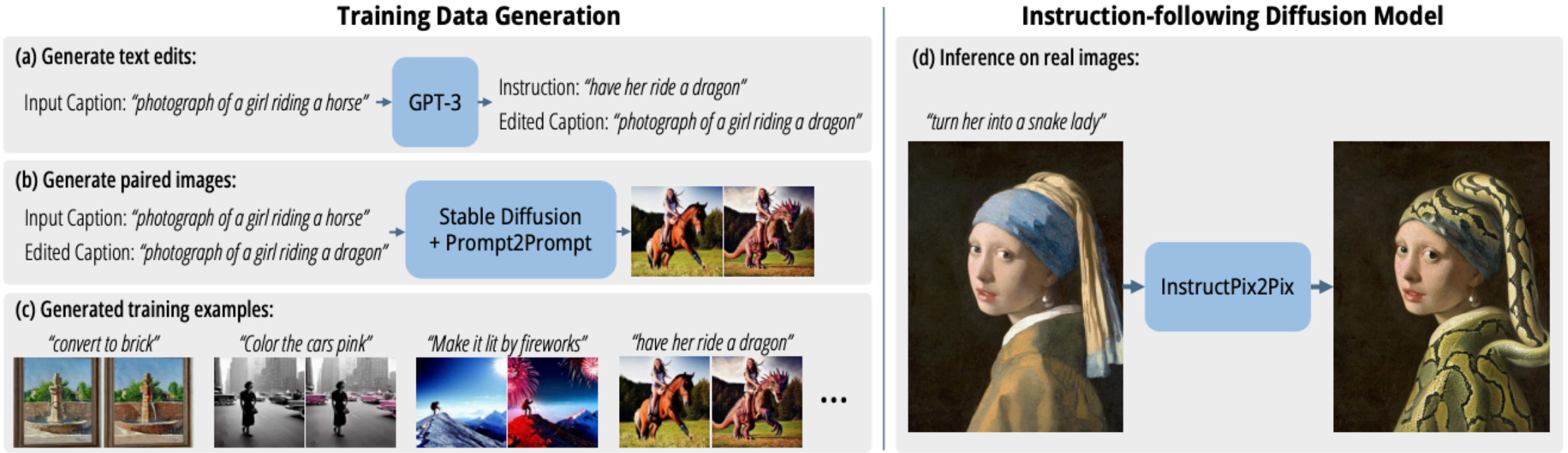


Figure 2. Our method consists of two parts: generating an image editing dataset, and training a diffusion model on that dataset. (a) We first use a finetuned GPT-3 to generate instructions and edited captions. (b) We then use StableDiffusion [52] in combination with Prompt-to-Prompt [17] to generate pairs of images from pairs of captions. We use this procedure to create a dataset (c) of over 450,000 training examples. (d) Finally, our InstructPix2Pix diffusion model is trained on our generated data to edit images from instructions. At inference time, our model generalizes to edit real images from human-written instructions.

T. Brooks, A. Holynski, and A. Efros. InstructPix2Pix: Learning to Follow Image Editing Instructions. CVPR 2023

InstructPix2Pix

- Fine-tuning GPT-3:

	Input LAION caption	Edit instruction	Edited caption
Human-written (700 edits)	*Yefim Volkov, Misty Morning*	*make it afternoon*	*Yefim Volkov, Misty Afternoon*
	girl with horse at sunset	*change the background to a city*	*girl with horse at sunset in front of city*
	painting-of-forest-and-pond	*Without the water.*	*painting-of-forest*
	...	...	...
GPT-3 generated (>450,000 edits)	*Alex Hill, Original oil painting on canvas, Moonlight Bay*	*in the style of a coloring book*	*Alex Hill, Original coloring book illustration, Moonlight Bay*
	The great elf city of Rivendell, sitting atop a waterfall as cascades of water spill around it	*Add a giant red dragon*	*The great elf city of Rivendell, sitting atop a waterfall as cascades of water spill around it with a giant red dragon flying overhead*
	Kate Hudson arriving at the Golden Globes 2015	*make her look like a zombie*	*Zombie Kate Hudson arriving at the Golden Globes 2015*
	...	...	...

Table 1. We label a small text dataset, finetune GPT-3, and use that finetuned model to generate a large dataset of text triplets. As the input caption for both the labeled and generated examples, we use real image captions from LAION. Highlighted text is generated by GPT-3.

InstructPix2Pix

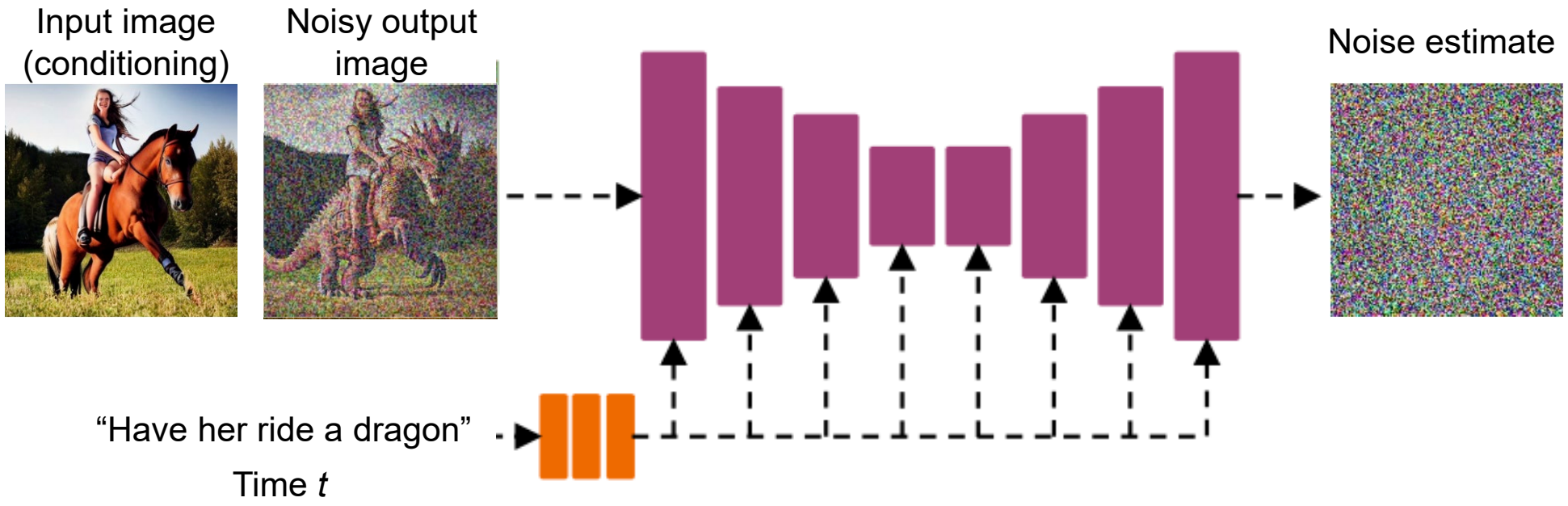
- Generating input-output image pairs:

(a) Without Prompt-to-Prompt. (b) With Prompt-to-Prompt.

Figure 3. Pair of images generated using StableDiffusion [52] with and without Prompt-to-Prompt [17]. For both, the corresponding captions are *"photograph of a girl riding a horse"* and *"photograph of a girl riding a dragon"*.

InstructPix2Pix

- Fine-tuning a DM for image-to-image translation:



InstructPix2Pix: Results

Figure 5. *Mona Lisa* transformed into various artistic mediums.

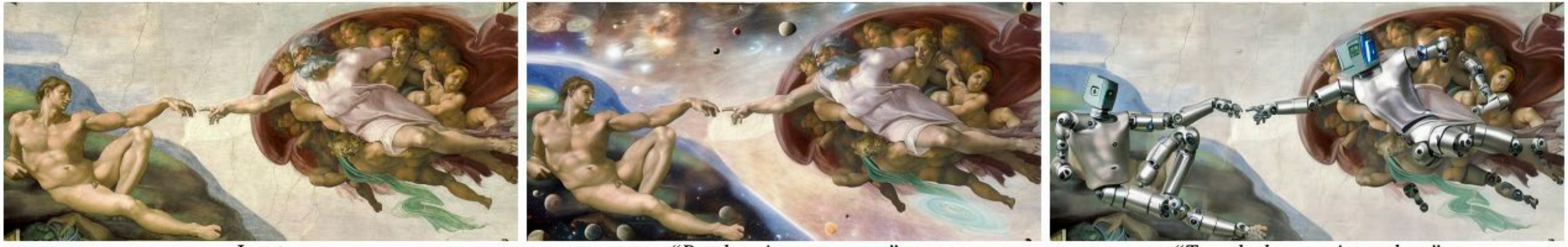


Figure 6. *The Creation of Adam* with new context and subjects (generated at 768 resolution).

InstructPix2Pix: Results

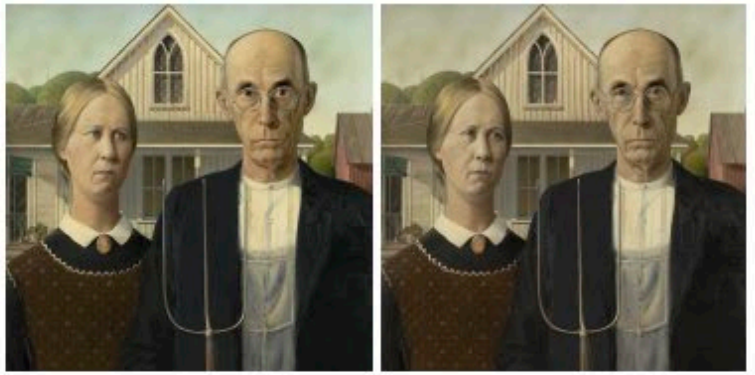
InstructPix2Pix: Results

InstructPix2Pix: Failure cases

"Zoom into the image"

"Move it to Mars"

"Color the tie blue"

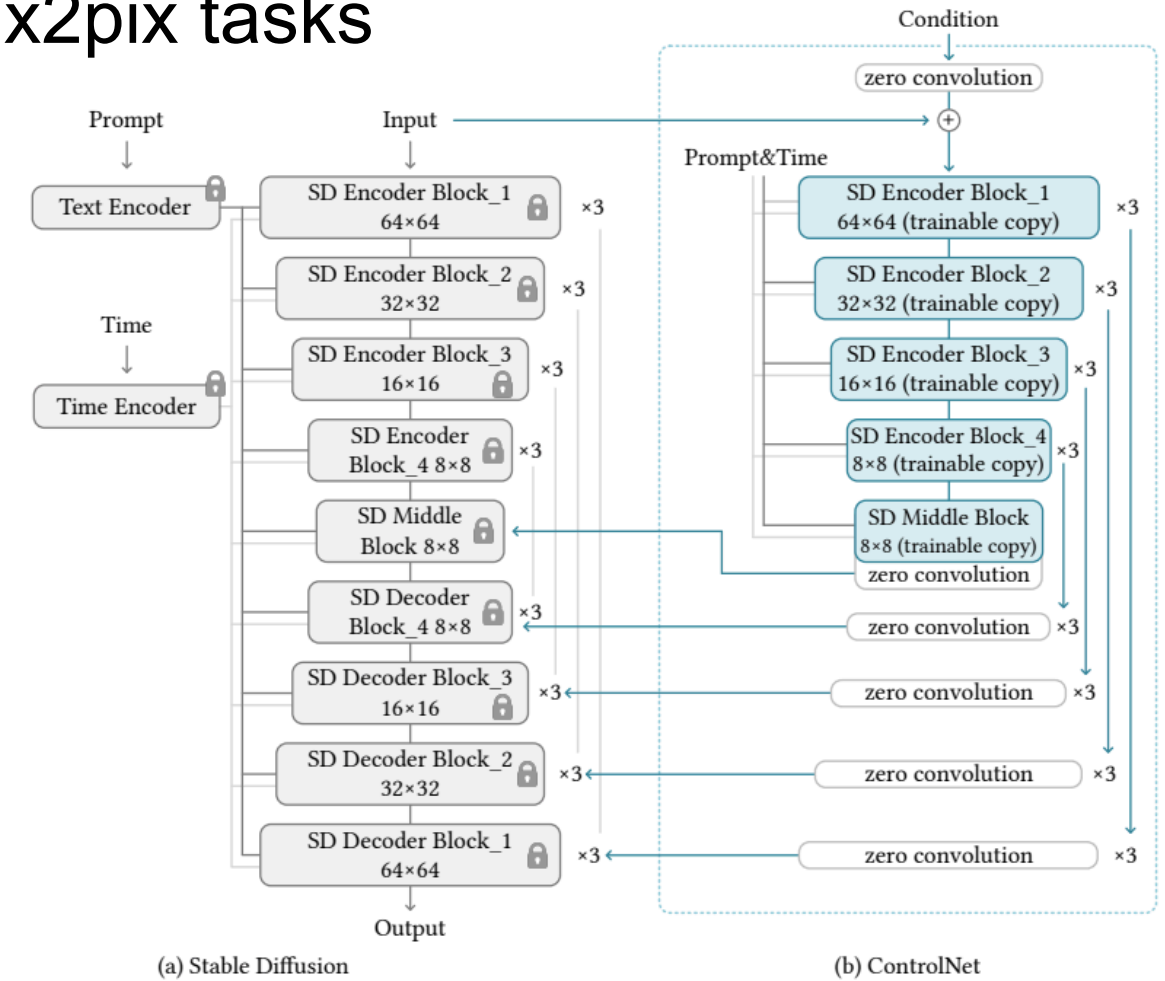


"Have the people swap places"

Figure 13. Failure cases. Left to right: our model is not capable of performing viewpoint changes, can make undesired excessive changes to the image, can sometimes fail to isolate the specified object, and has difficulty reorganizing or swapping objects with each other.

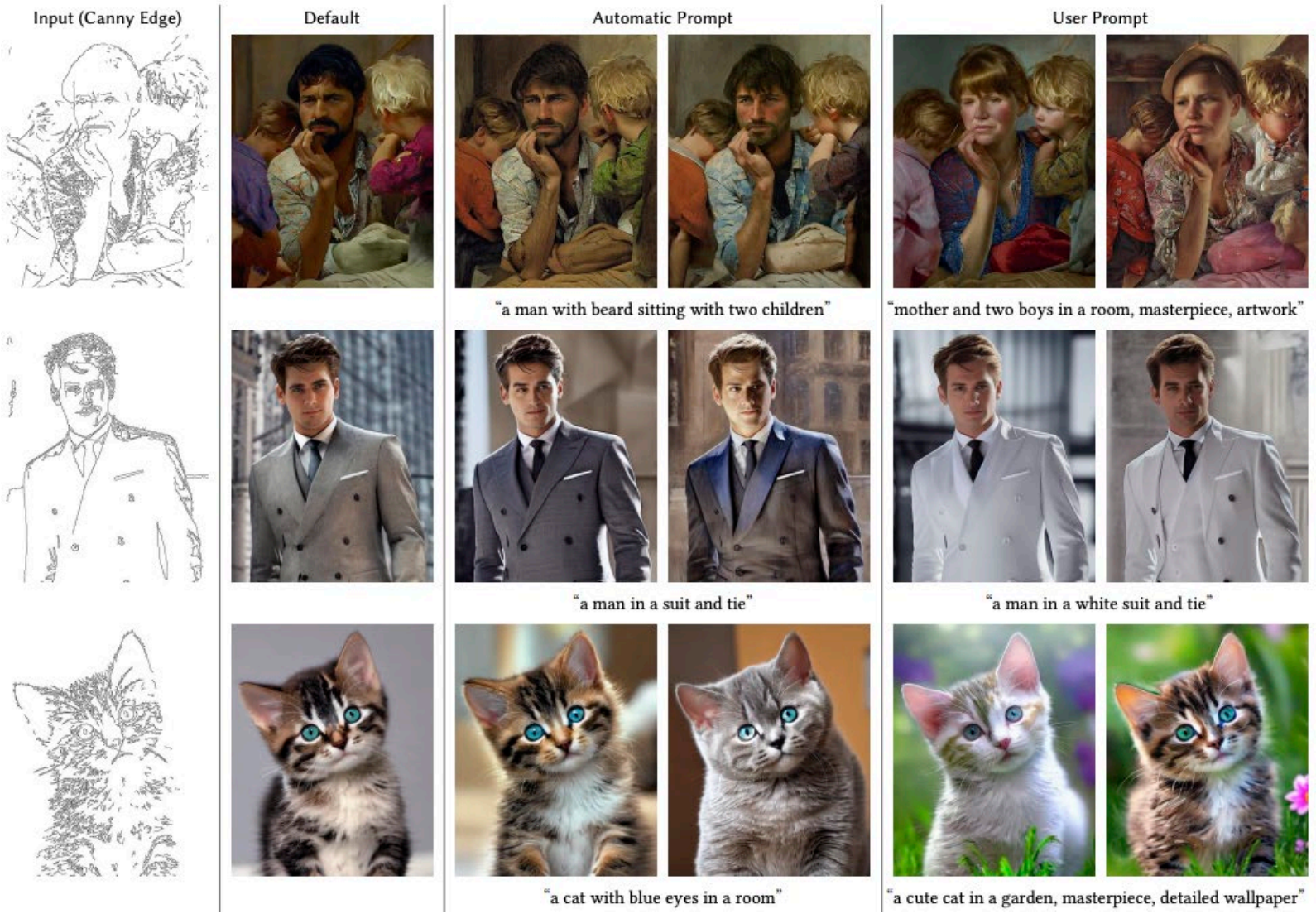
ControlNet

- Add a trainable “wrapper” around a pre-trained DM to fine-tune it for pix2pix tasks



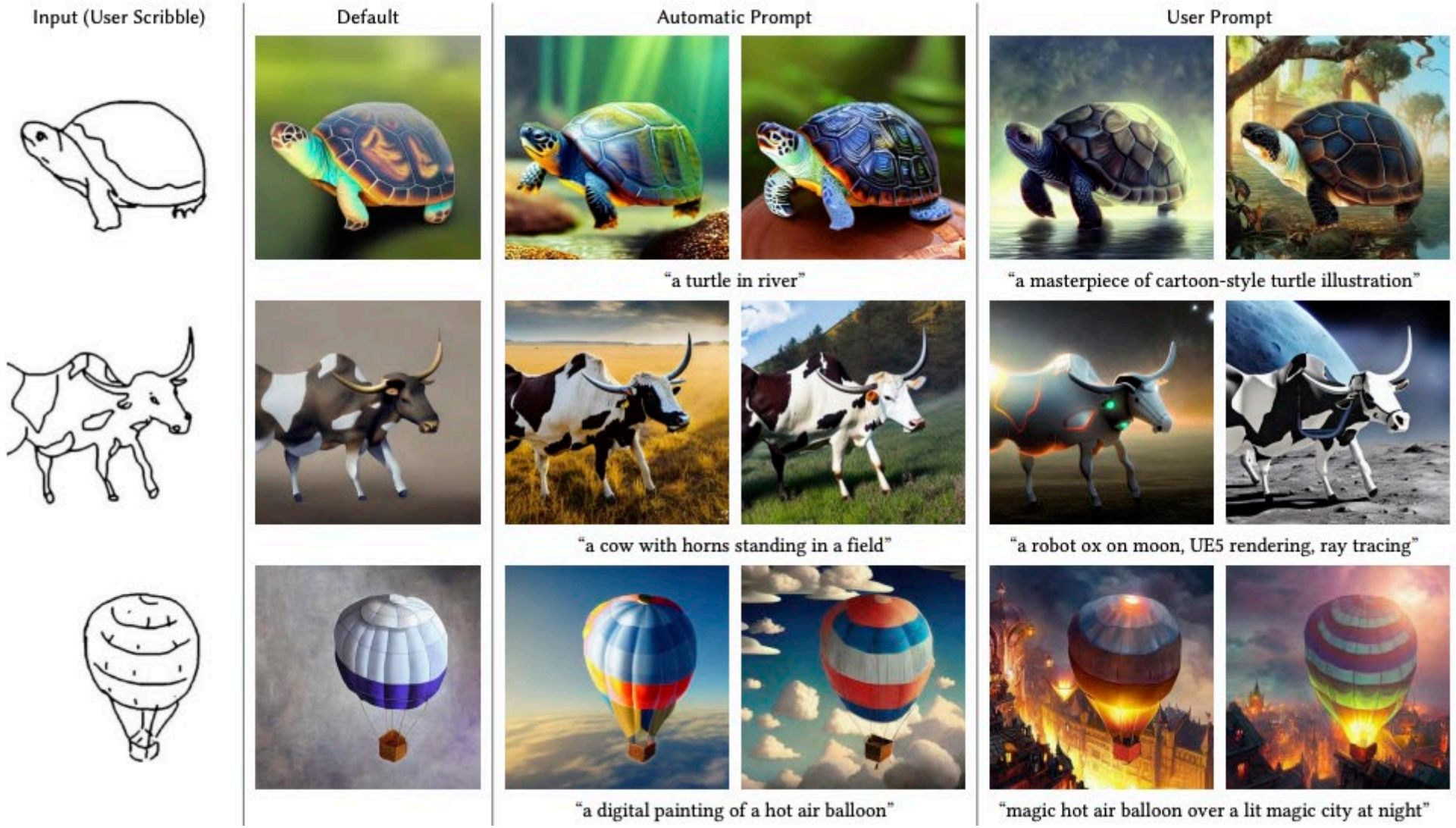
L. Zhang and M. Agrawala. Adding Conditional Control to Text-to-Image Diffusion Models. arXiv 2023

ControlNet



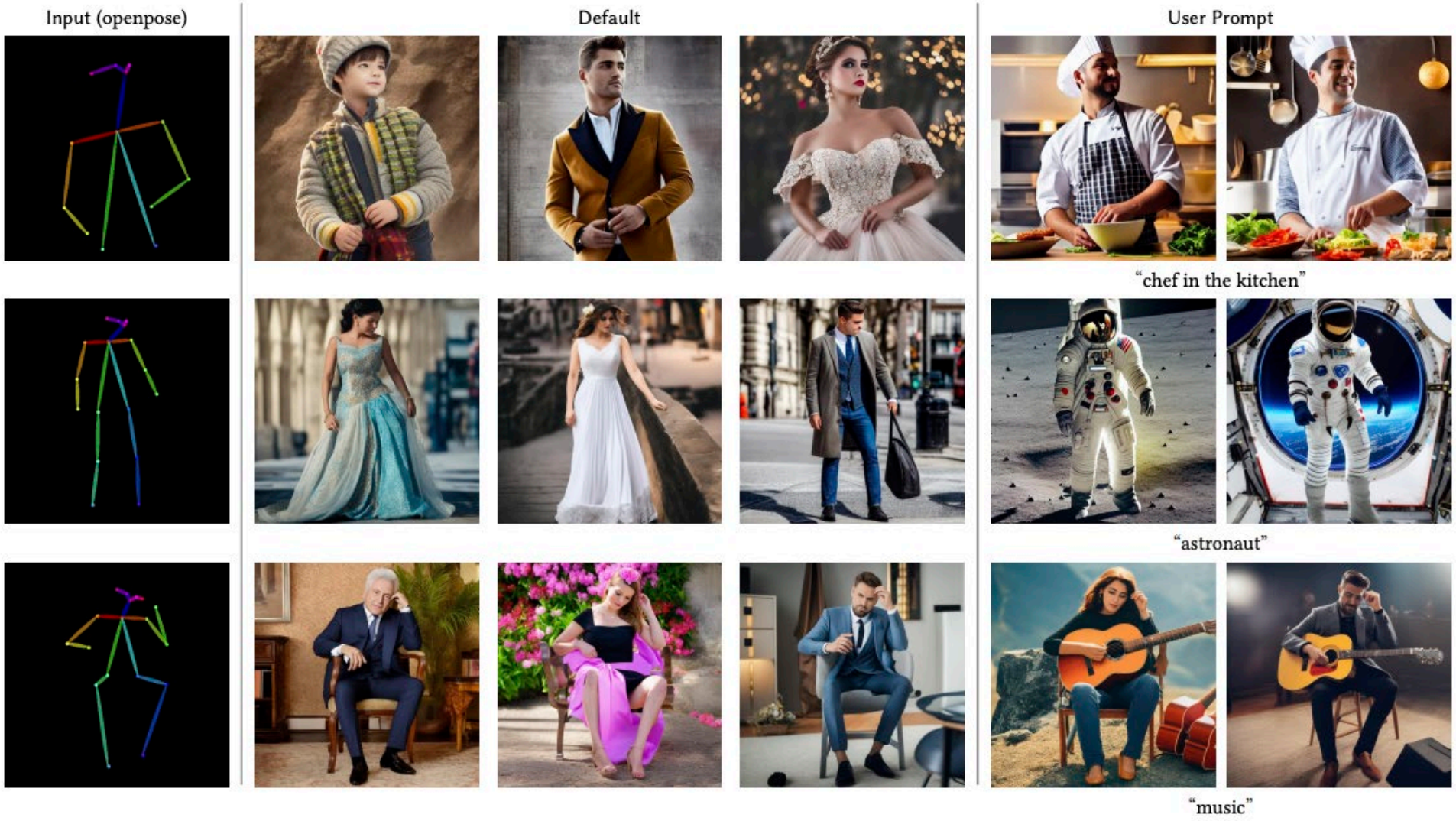
L. Zhang and M. Agrawala. Adding Conditional Control to Text-to-Image Diffusion Models. arXiv 2023

ControlNet



L. Zhang and M. Agrawala. Adding Conditional Control to Text-to-Image Diffusion Models. arXiv 2023

ControlNet



L. Zhang and M. Agrawala. Adding Conditional Control to Text-to-Image Diffusion Models. arXiv 2023

Societal, ethical, and legal issues

- Closed or open?
- Safe or unsafe?
- Potential for generating DeepFakes and misinformation
- Dataset image rights
- Artists’ rights
- The nature of creativity

In the news

ARTIFICIAL INTELLIGENCE / TECH / LAW

Getty Images is suing the creators of AI art tool Stable Diffusion for scraping its content

An image created by Stable Diffusion showing a recreation of Getty Images' watermark. Image: The Verge / Stable Diffusion

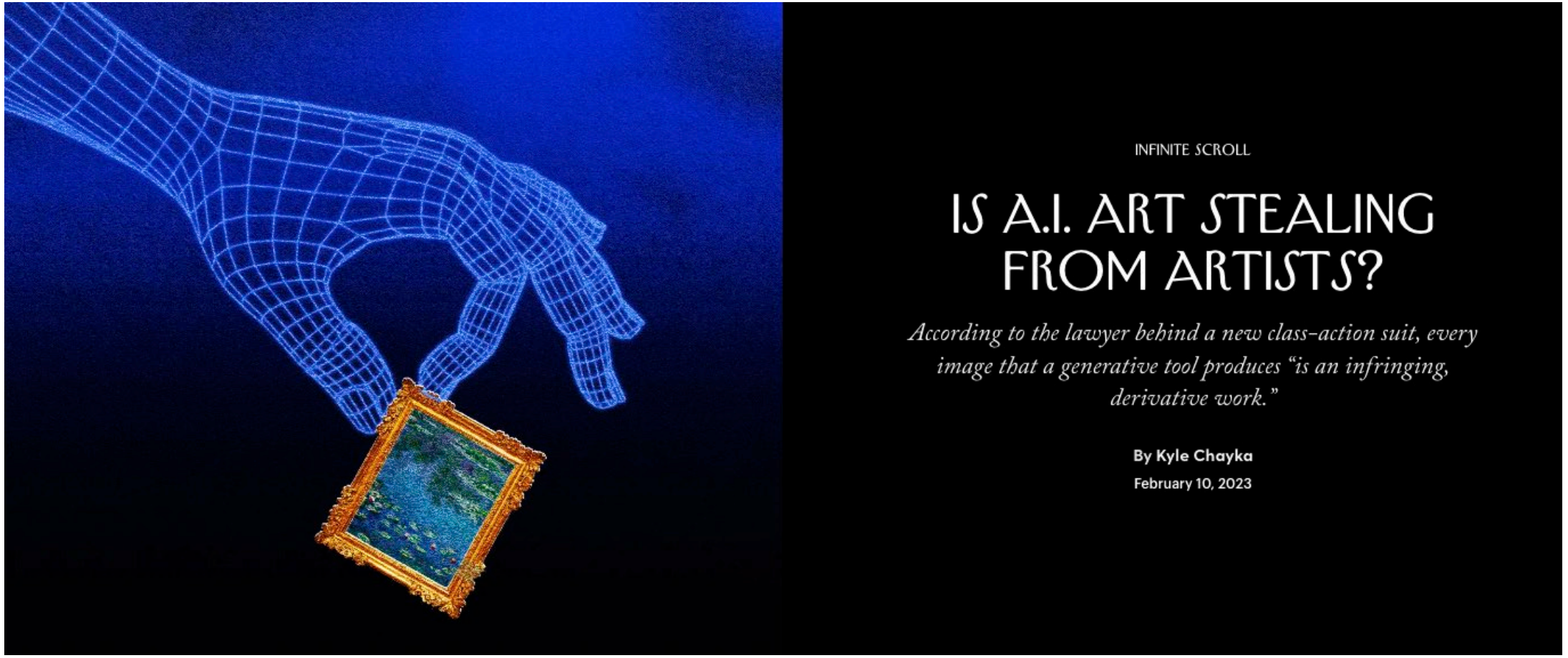
/ Getty Images claims Stability AI 'unlawfully' scraped millions of images from its site. It's a significant escalation in the developing legal battles between generative AI firms and content creators.

By JAMES VINCENT

Jan 17, 2023, 4:30 AM CST | 18 Comments / 18 New

https://www.theverge.com/2023/1/17/23558516/ai-art-copyright-stable-diffusion-getty-images-lawsuit

In the news



https://www.newyorker.com/culture/infinite-scroll/is-ai-art-stealing-from-artists

In the news

Fake Trump arrest photos: How to spot an AI-generated image

This image looks realistic, but take a closer look at Trump's right arm and neck

https://www.bbc.com/news/world-us-canada-65069316

In the news

Midjourney Bans AI Images of Chinese President Xi Jinping

APR 03, 2023 MATT GROWCOOT

https://petapixel.com/2023/04/03/midjourney-bans-ai-images-of-chinese-president-xi-jinping/