Deep Generative Learning

Deep Learning for Computer Vision

Valeriya Strizhkova

21.11.23

About myself

Valeriya Strizhkova

PhD candidate @ Inria & 3iA & UCA

Research topics:

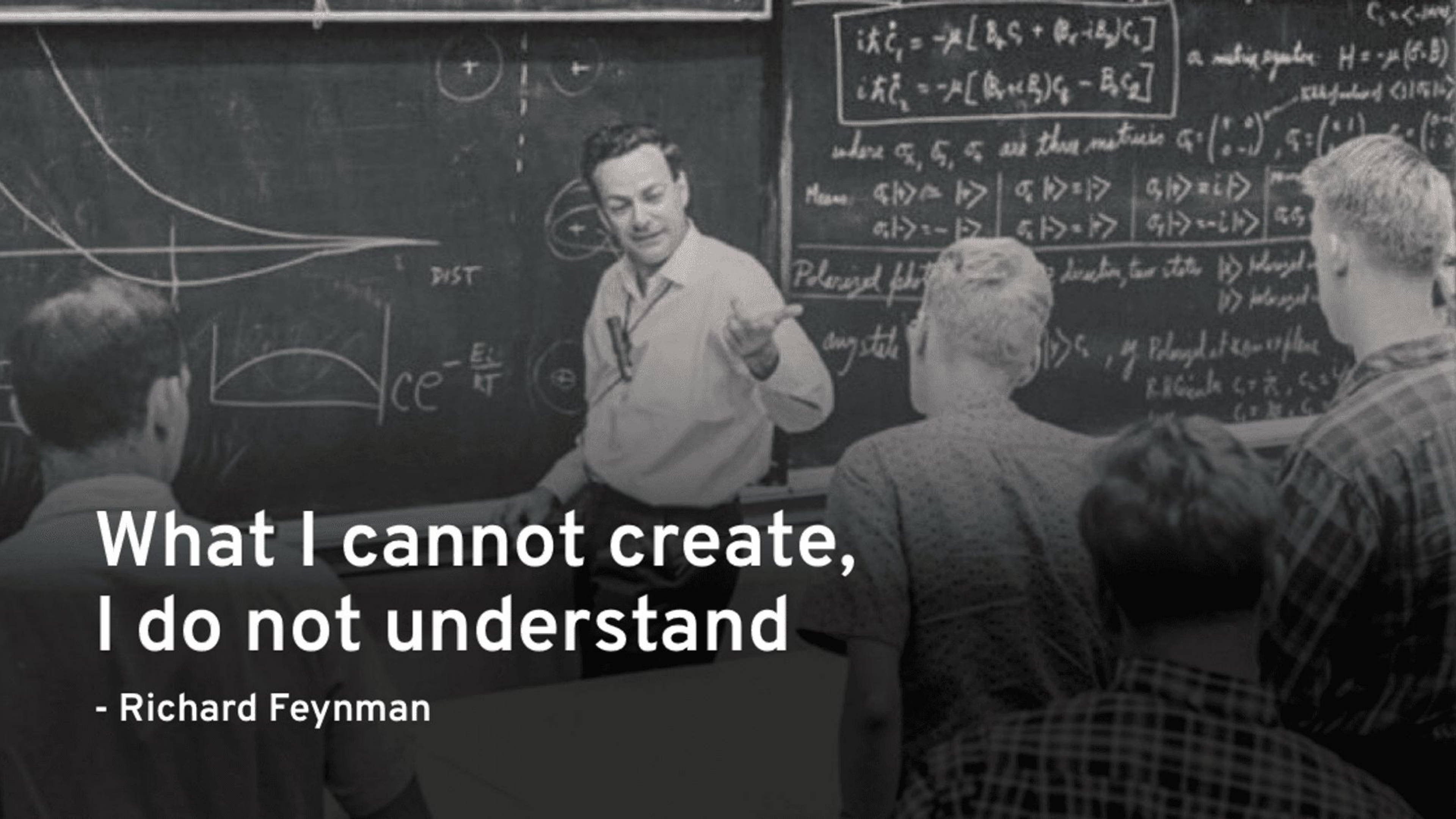
- video understanding
- human behavior understanding
- multi-sensor fusion

valeriya.strizhkova@inria.fr

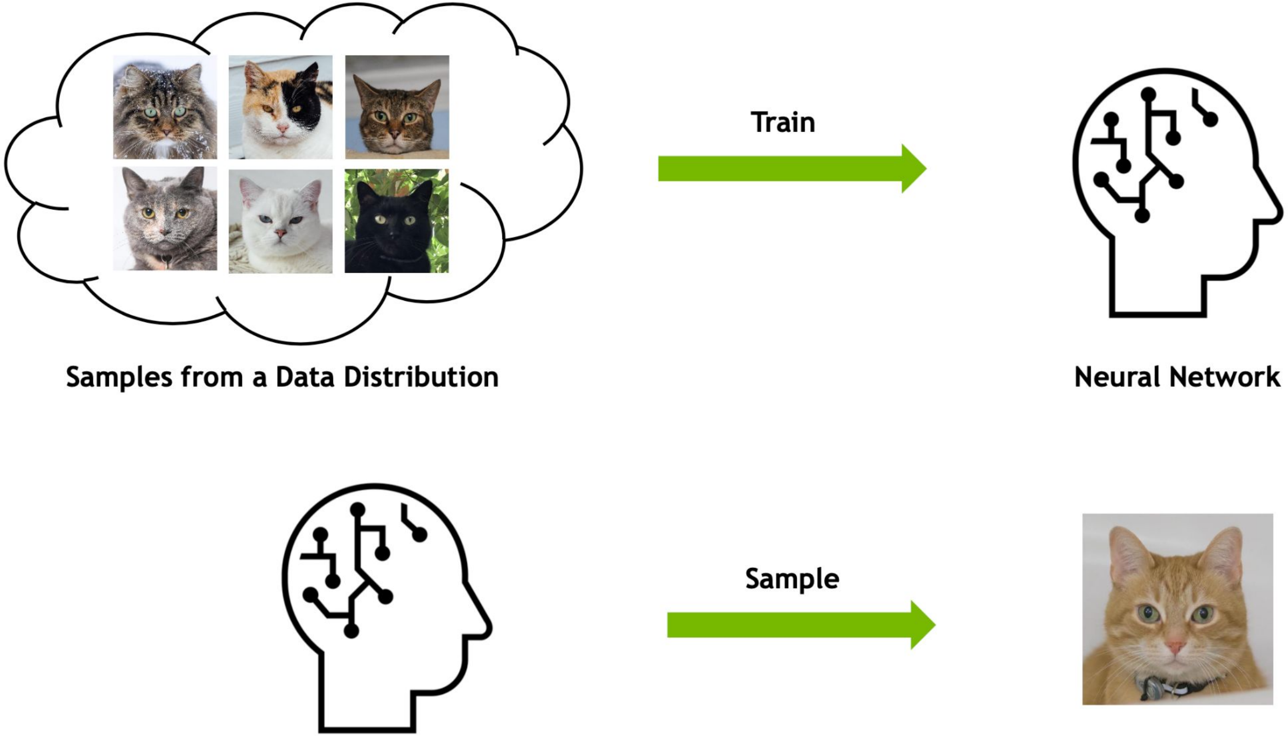
Outline

- Introduction
  - What is a generative model?
  - Types of deep generative models
  - Evaluation: IS, FID
- Image Generation
  - Autoregressive models: PixelCNN, DALL-E 1
  - Variational Autoencoders: dVAE
  - Generative Adversarial Networks: StyleGAN
  - Diffusion Models: Latent diffusion, DALL-E 2
- Video Generation
  - Phenaki
  - Everybody Dance Now
  - Make-A-Video

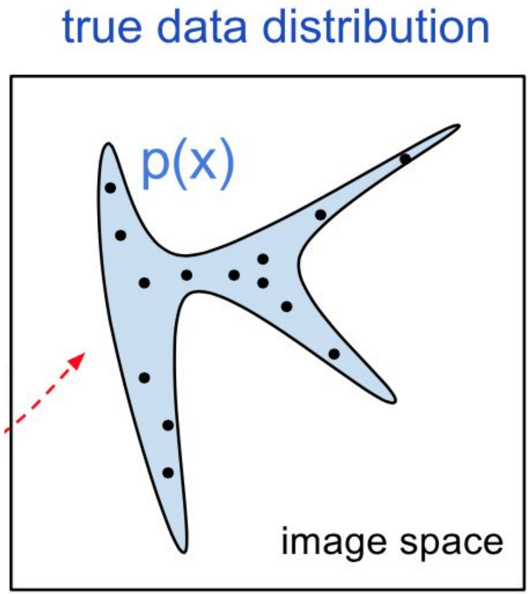
Introduction


What I cannot create,
I do not understand
- Richard Feynman

What is a Generative Model?

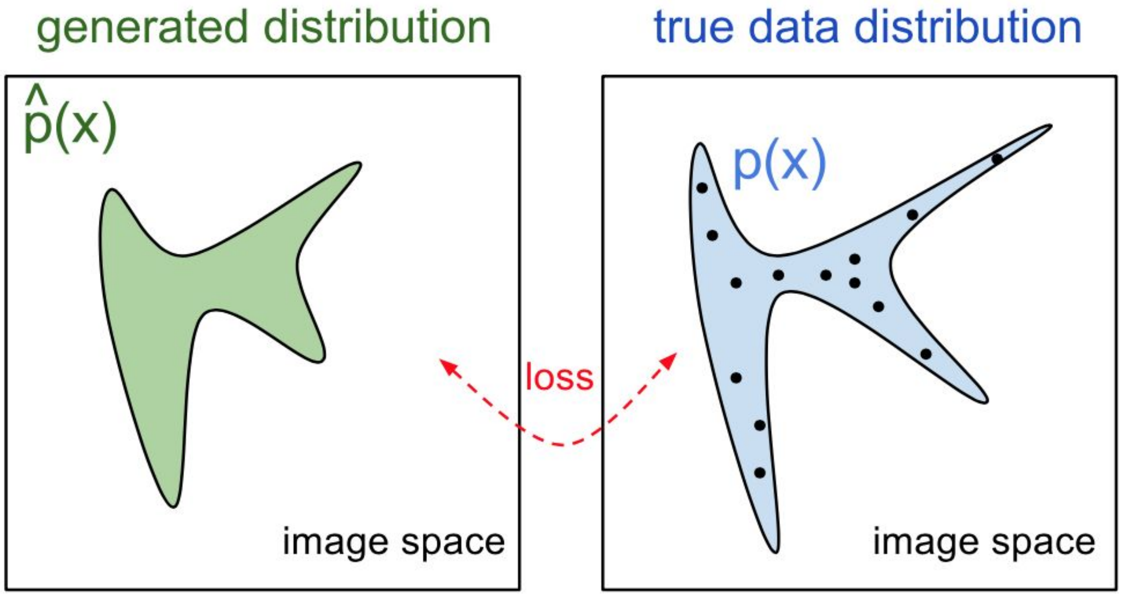


What is a Generative Model?



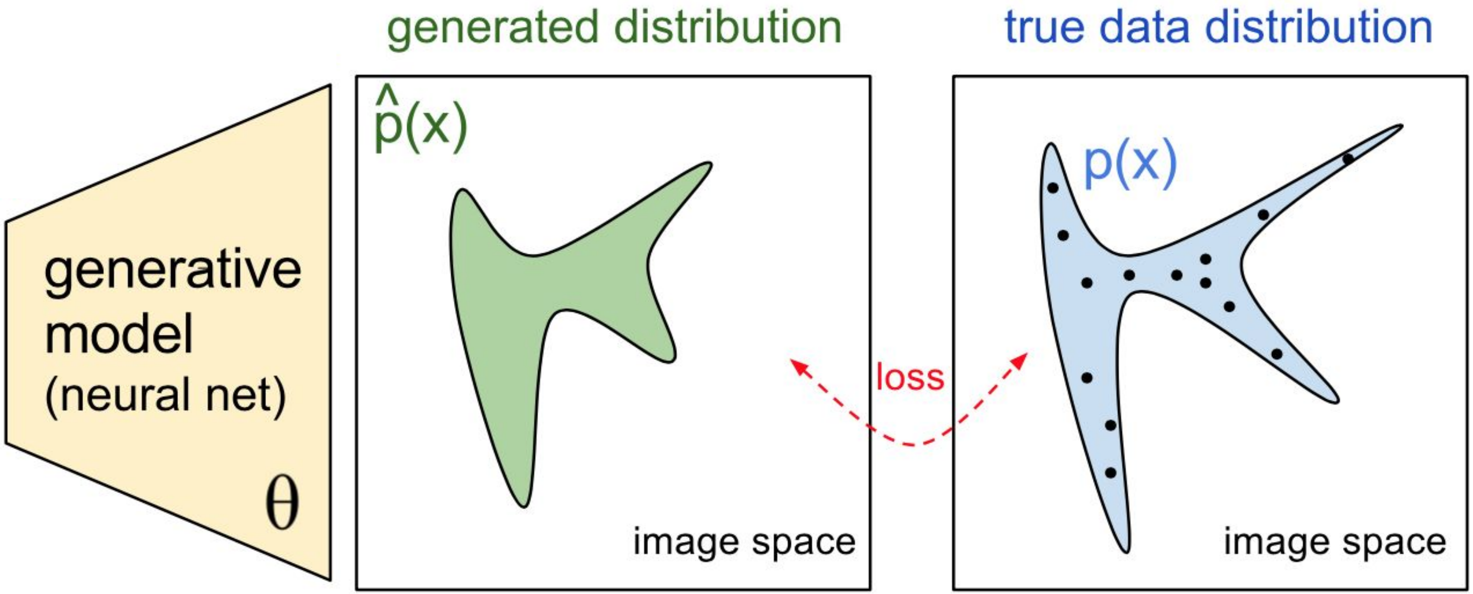
source: https://openai.com/research/generative-models

What is a Generative Model?



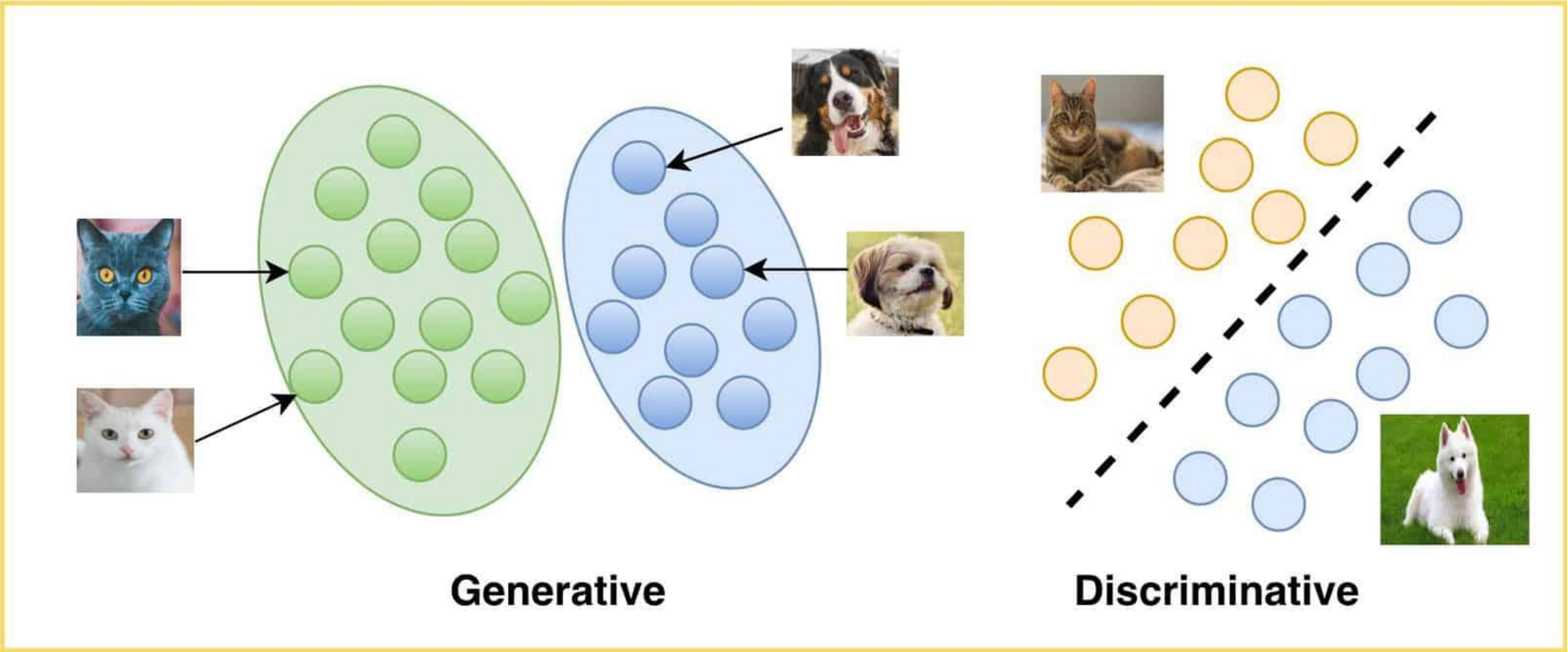
source: https://openai.com/research/generative-models

What is a Generative Model?



source: https://openai.com/research/generative-models

What is a Generative Model?



source: https://learnopencv.com/generative-and-discriminative-models/

What is a Generative Model?

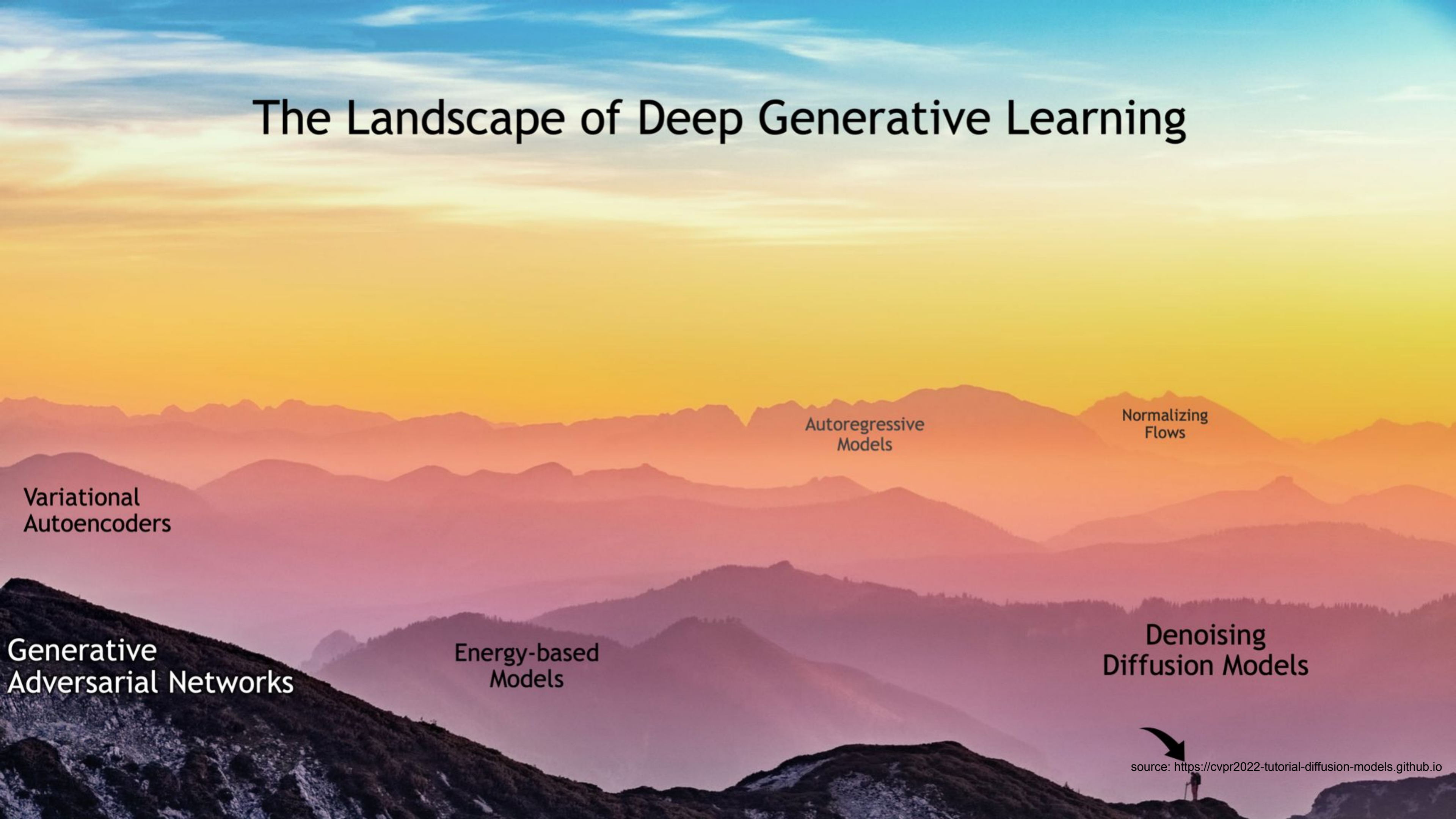
Art & Design

Content Generation

Representation Learning

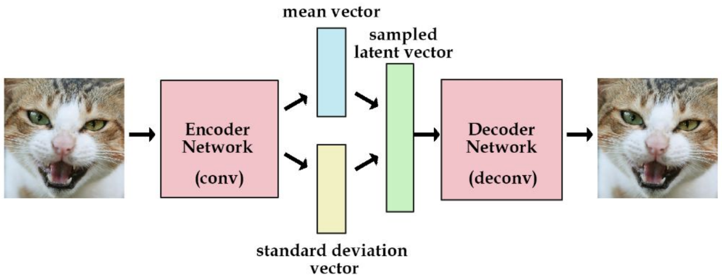
Entertainment

The Landscape of Deep Generative Learning

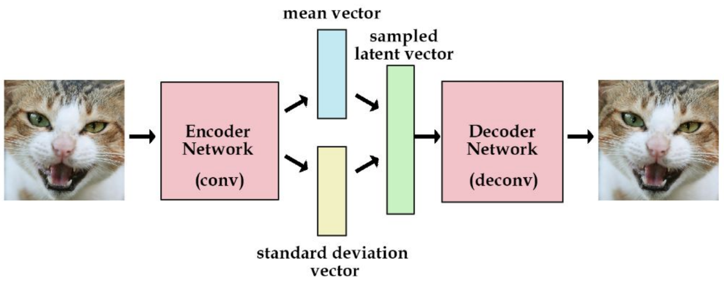


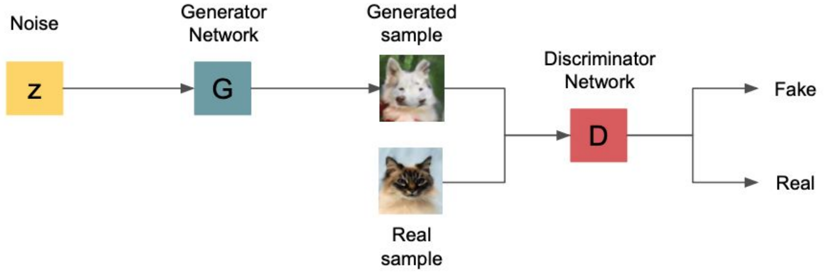
source: https://cvpr2022-tutorial-diffusion-models.github.io

Types of generative models

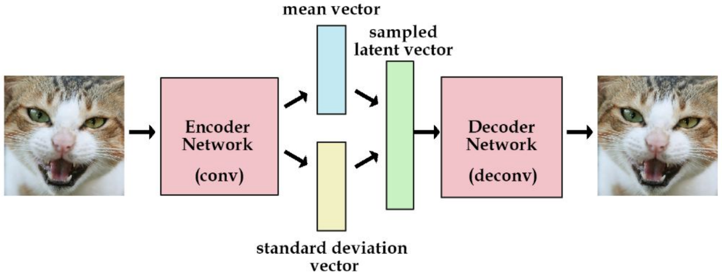


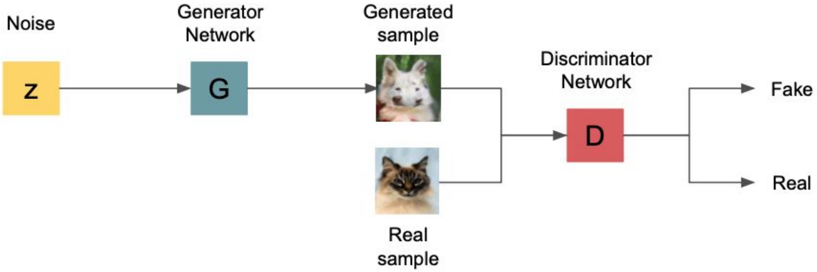
Types of generative models



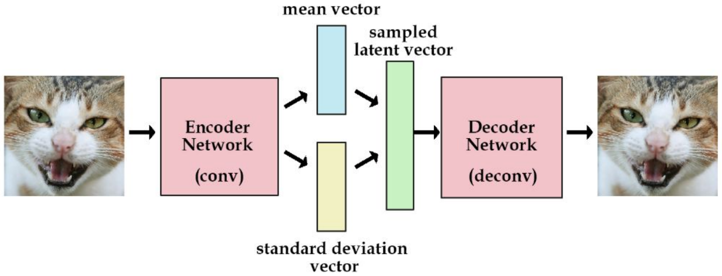


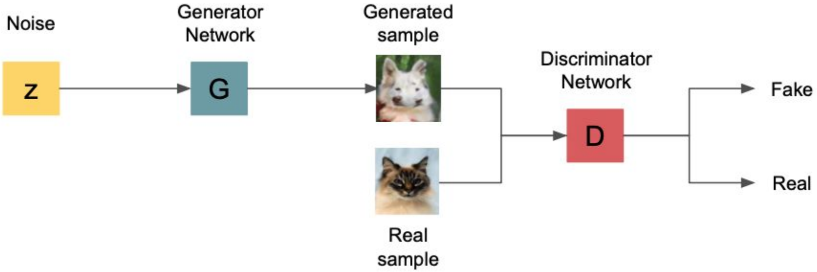
Types of generative models

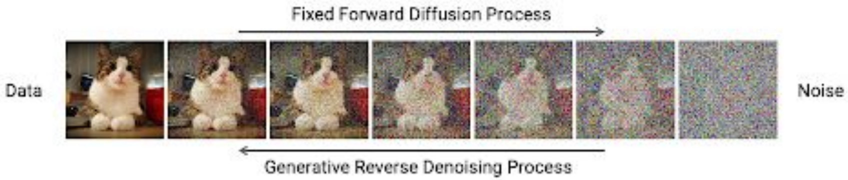




Types of generative models

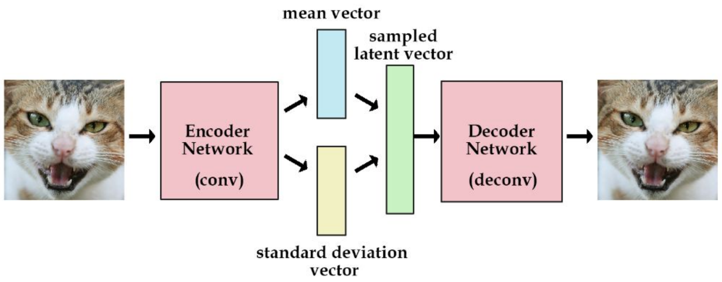






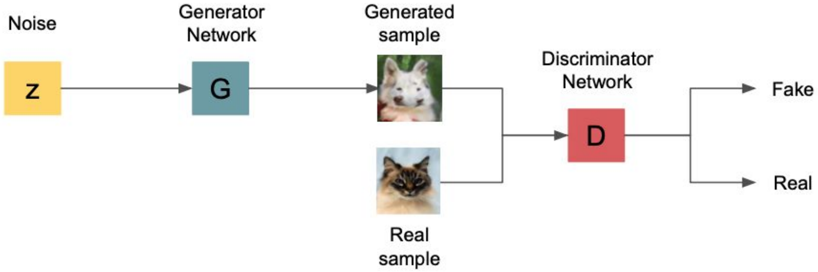
Types of generative models

Variational Autoencoder:

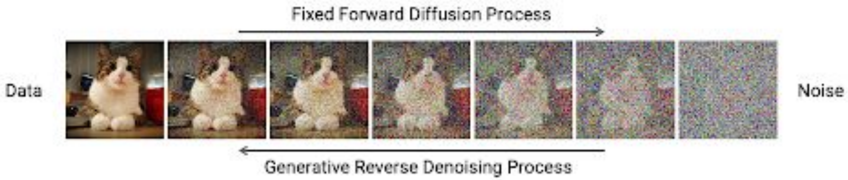


Autoregressive model:

Generative Adversarial Network:

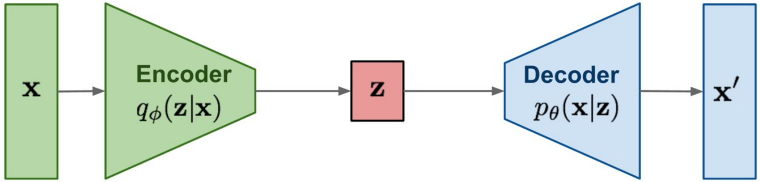


Diffusion model:



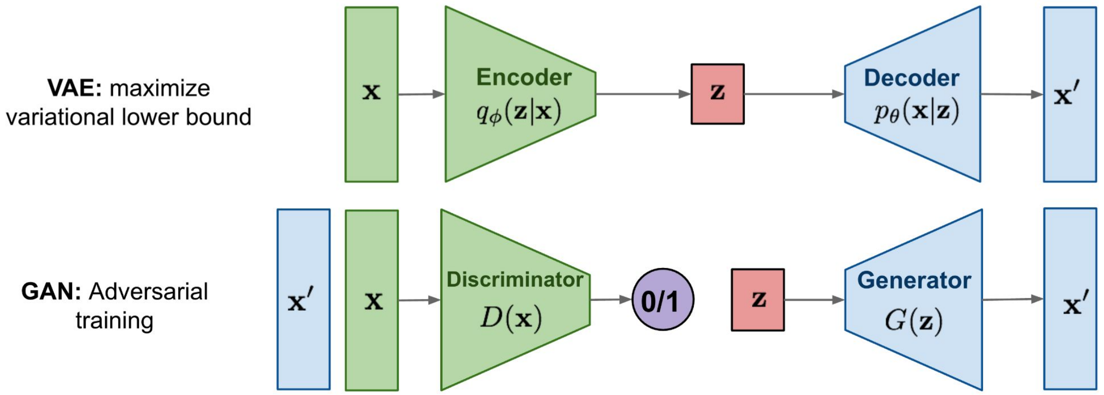
Types of generative models

VAE: maximize variational lower bound



source: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

Types of generative models



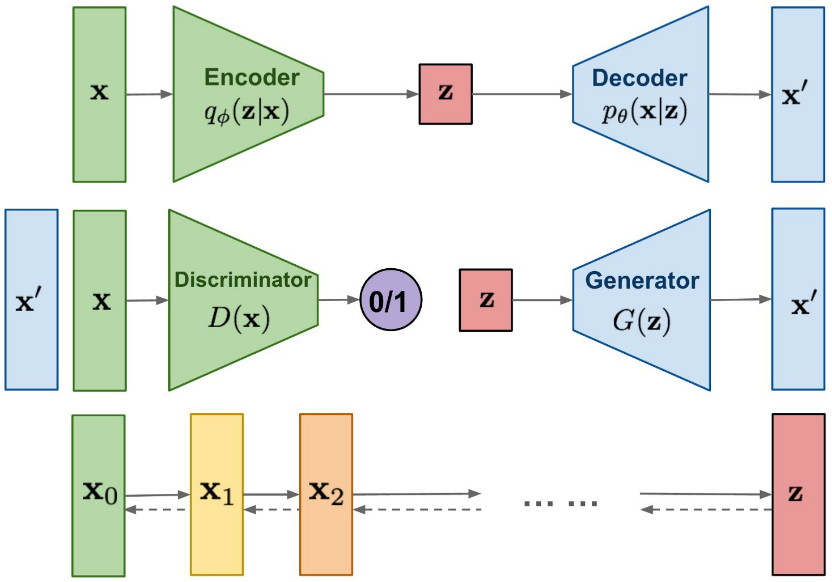
source: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

Types of generative models

VAE: maximize variational lower bound

GAN: Adversarial training

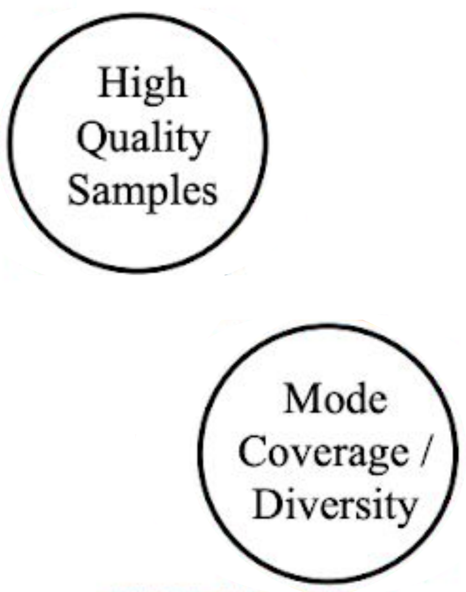
Diffusion models: Gradually add Gaussian noise and then reverse



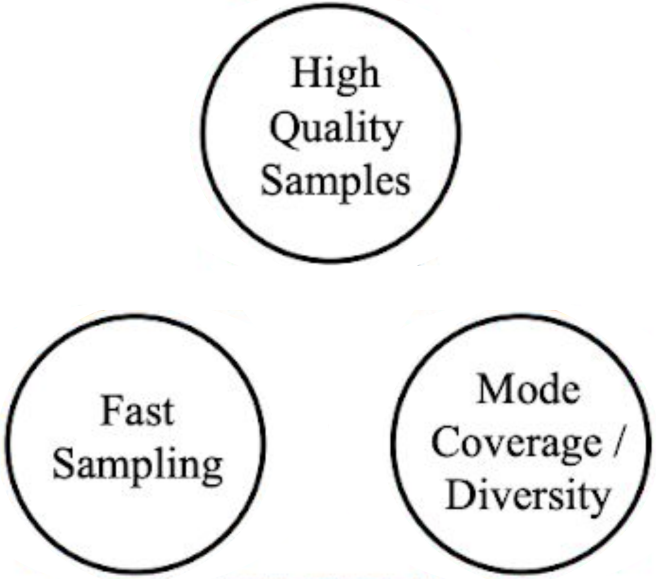
source: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

Generative learning trilemma

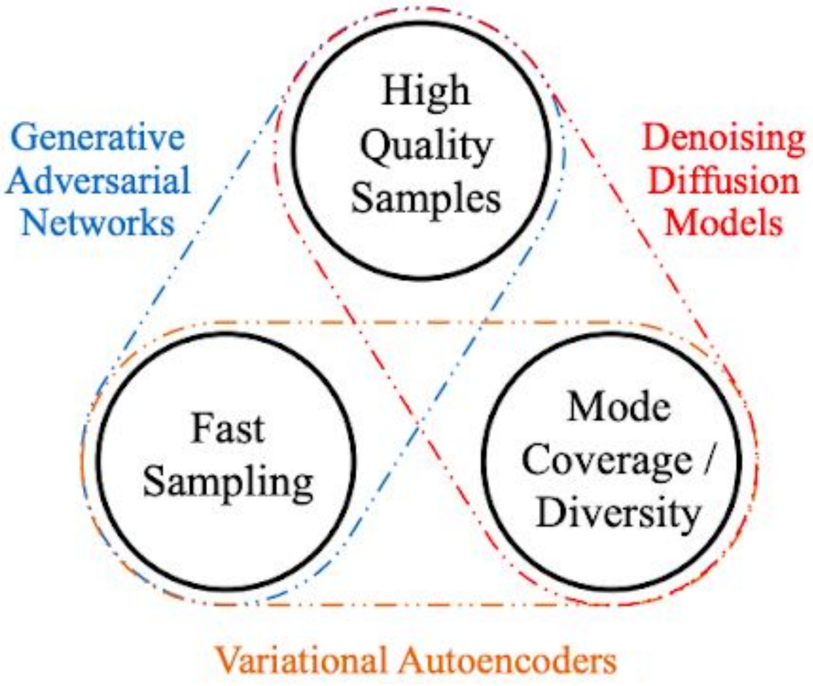
Generative learning trilemma



Generative learning trilemma



Generative learning trilemma



Generative Models Evaluation

Which of these images looks better?

Generative Models Evaluation

Which of these images looks more realistic?

Generative Models Evaluation

Which of these images appears to be more similar to the text prompt?

Prompt: The saying "BE EXCELLENT TO EACH OTHER" written on a red brick wall with a graffiti image of a green alien wearing a tuxedo. A yellow fire hydrant is on a sidewalk in the foreground.

Generative Models Evaluation

- Human-based ratings and preference judgments
- Inception Score (quality and diversity) [1]
- Frechet Inception Distance (distinguish between synthetic and generated images) [2]
- Structured Similarity Index Metric (SSIM) [3]
- CLIP score [4]

[1] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen. Improved Techniques for Training GANs. NeurIPS 2016 https://arxiv.org/abs/1606.03498
[2] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. NeurIPS 2017 https://arxiv.org/abs/1706.08500
[3] Zhou Wang; A.C. Bovik; H.R. Sheikh; E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing 2004 https://www.cns.nyu.edu/pub/eero/wang03-reprint.pdf
[4] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, Yejin Choi.CLIPScore: A Reference-free Evaluation Metric for Image Captioning. EMNLP 2021https://arxiv.org/abs/2104.08718

Inception Score (IS)

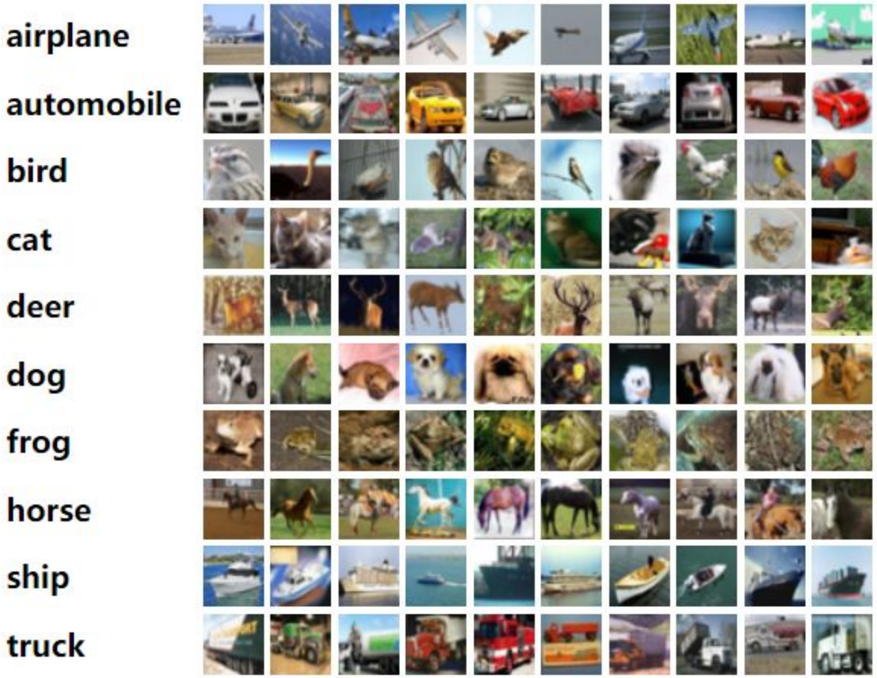
IS measures:

- the **quality** of the generated images
- their **diversity**

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen. Improved Techniques for Training GANs. NeurIPS 2016 https://arxiv.org/abs/1606.03498

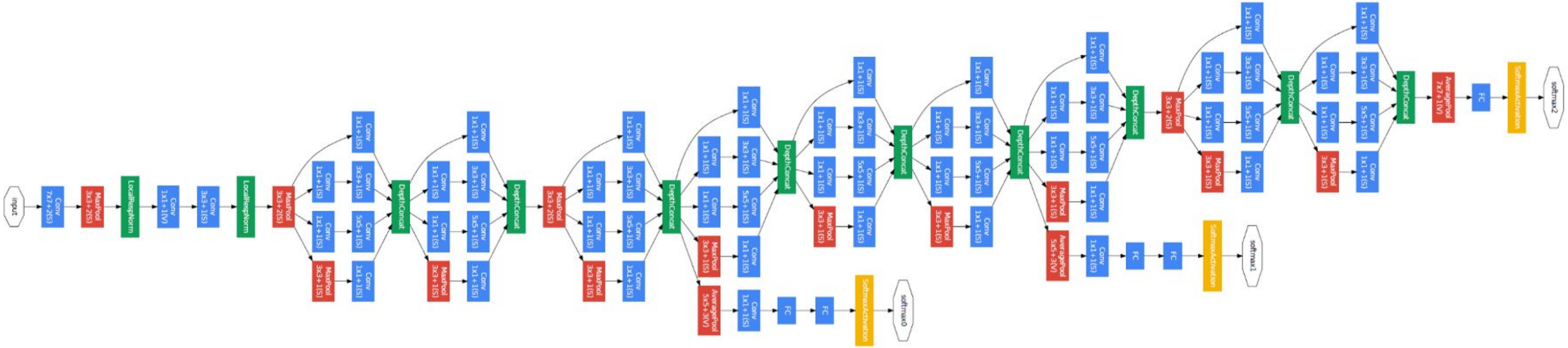
Inception Score (IS)

CIFAR10 dataset:



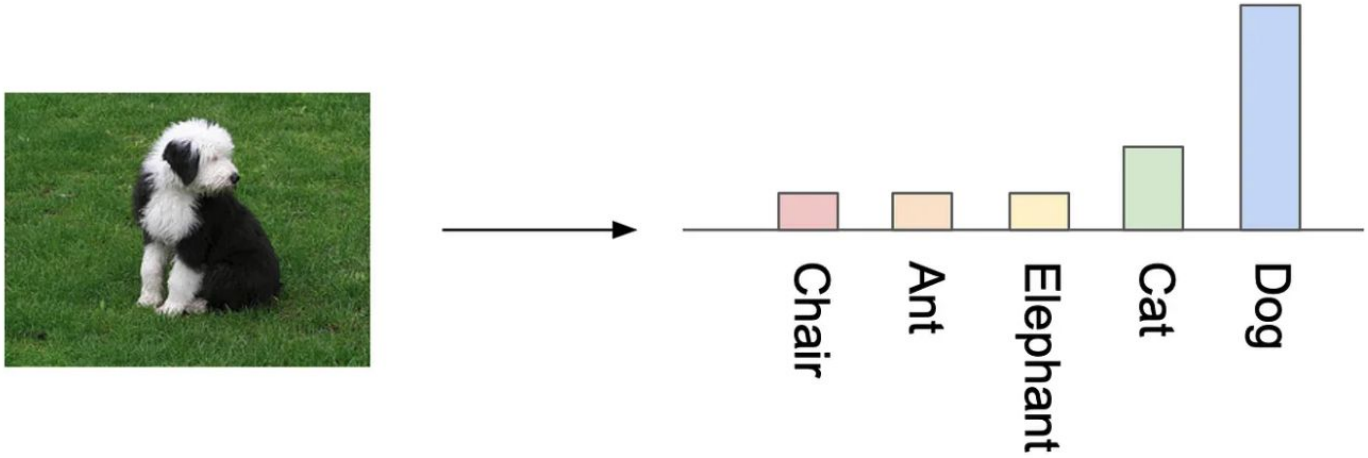
Inception Score (IS)

Generate 50000 images by the **Inception** image classifier pre-trained on CIFAR10



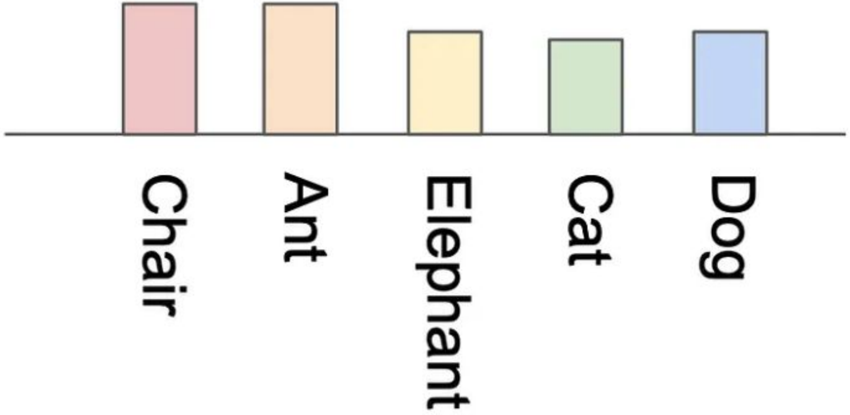
Inception Score (IS)

Generated images are fed into the Inception image classifier network pre-trained on the CIFAR10 dataset predict conditional probability p(y|x) — where y is the label and x is the generated data



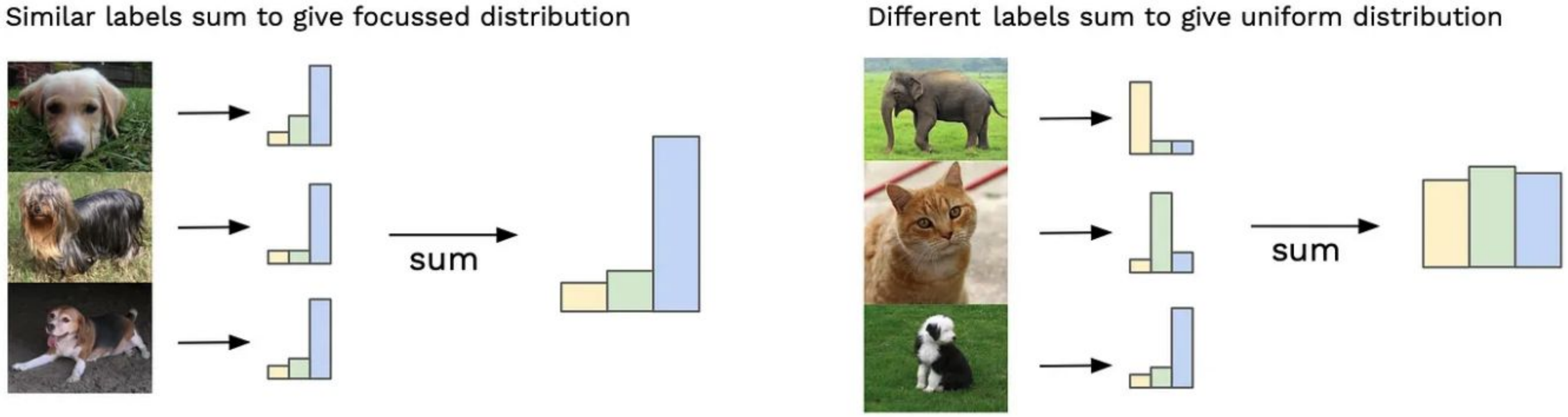
Inception Score (IS)

If the probability scores are widely distributed then the generated image is of low quality:



Inception Score (IS)

Calculate marginal probability $p(y) = \int_z p(y|x = G(z))dz$



Inception Score (IS)

- **Quality**: conditional probability **p(y|x)**
- **Diversity**: marginal probability **p(y)**

We want

- the conditional probability **p(y|x)** to be highly predictable (**low entropy**) i.e. given an image, we should know the object type easily
- the marginal probability **p(y)** to be uniform (**high entropy**)

Inception Score (IS)

Compute their KL-divergence to combine these two criteria:

$$IS(G) = \exp(E_{x \sim p_g} KL(p(y|x) || p(y)))$$

Inception Score (IS) limitations

IS is limited by what the Inception classifier can detect, which is linked to the training data

Frechet Inception Distance (FID)

FID compares the distribution of generated images with the distribution of real images

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. NeurIPS 2017 https://arxiv.org/abs/1706.08500

Frechet Inception Distance (FID)

- Uses the **Inception network** to extract features from an intermediate layer

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. NeurIPS 2017 https://arxiv.org/abs/1706.08500

Frechet Inception Distance (FID)

- Uses the Inception network to extract features from an intermediate layer
- Models data distribution for these features using a multivariate Gaussian distribution with mean μ and covariance Σ

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. NeurIPS 2017 https://arxiv.org/abs/1706.08500

Frechet Inception Distance (FID)

- Uses the Inception network to extract features from an intermediate layer
- Models data distribution for these features using a multivariate Gaussian distribution with mean μ and covariance Σ
- The FID between the real images x and generated images g is:

$$FID(x,g) = ||\mu_x - \mu_g||_2^2 + Tr(\Sigma_x + \Sigma_g - 2(\Sigma_x \Sigma_g)^{\frac{1}{2}})$$

where *Tr* sums up all the diagonal elements

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. NeurIPS 2017 https://arxiv.org/abs/1706.08500

Frechet Inception Distance (FID)

- **Lower** FID values mean **better** image quality and diversity
- FID is sensitive to mode collapse
- FID is more robust to noise than IS. If the model only generates one image per class, the distance will be high

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. NeurIPS 2017 https://arxiv.org/abs/1706.08500

Structured Similarity Index Metric (SSIM)

The Structural Similarity Index (SSIM) metric mimicks the human visual perception system which is highly capable of identifying **structural information** from a scene.
SSIM extracts 3 features from an image on a **pixel-wise** basis:

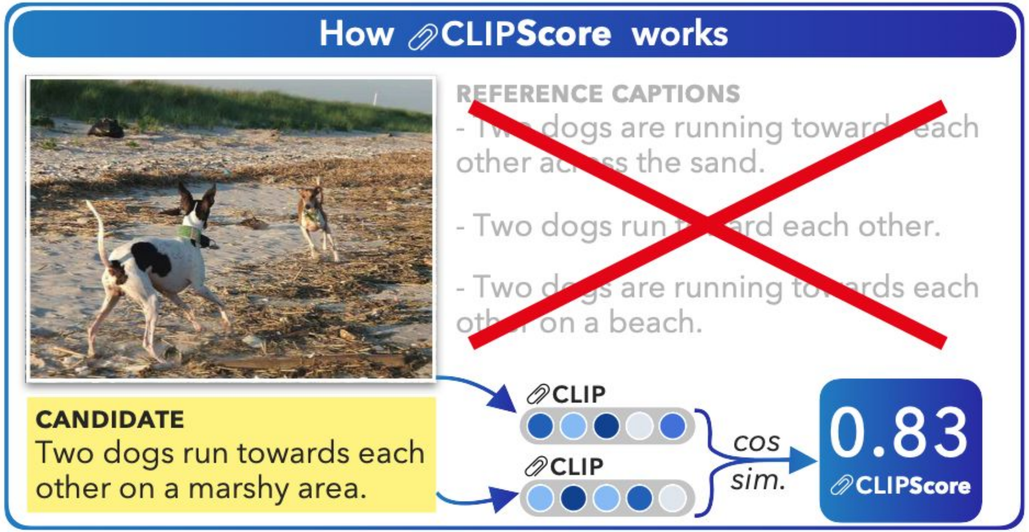
- Luminance
- Contrast
- Structure

The comparison between the two images is performed on the basis of these 3 features.

Zhou Wang; A.C. Bovik; H.R. Sheikh; E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing 2004 https://www.cns.nyu.edu/pub/eero/wang03-reprint.pdf

CLIP score

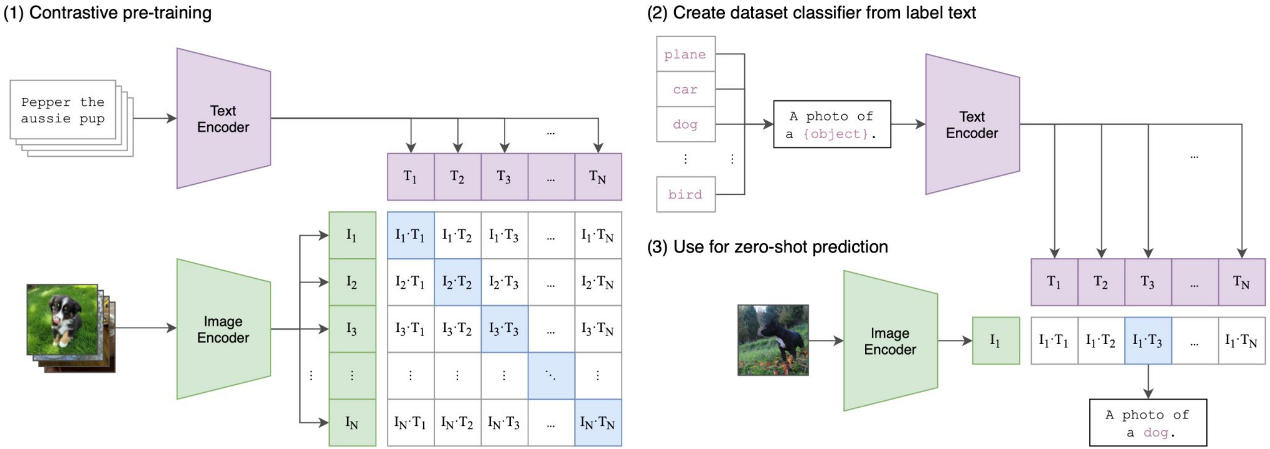
- CLIP score measures the compatibility of image-caption pairs.
- Higher CLIP scores imply higher compatibility



Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, Yejin Choi.CLIPScore: A Reference-free Evaluation Metric for Image Captioning. EMNLP 2021
https://arxiv.org/abs/2104.08718

OpenAI's CLIP

- encodes image, and text to similar embeddings
- is trained with contrastive learning, maximizing cosine similarity of corresponding image and text
- CLIP's output image embeddings contain both style and semantics
- is used in DALL-E 1, DALL-E 2, image-text classification



Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. ICML 2021 https://arxiv.org/abs/2103.00020 https://openai.com/research/clip

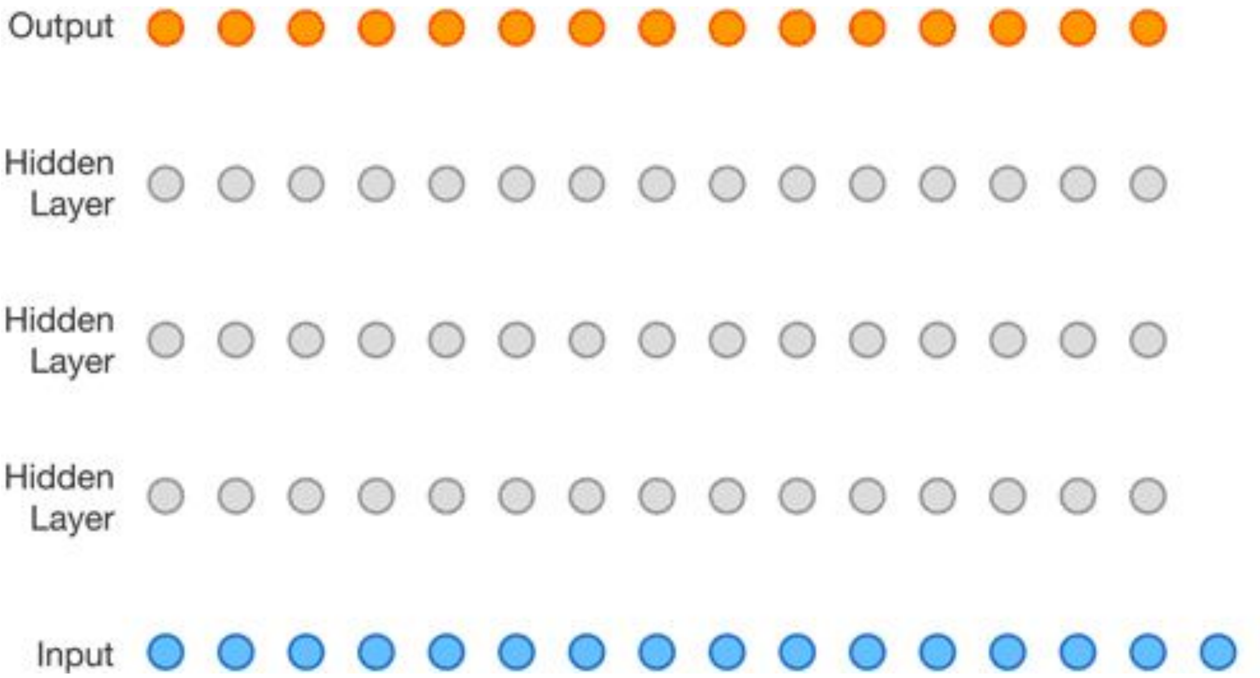
CLIP Architecture

- text and image have separate transformer encoders
- visual encoder is ViT (vision transformer)
- text encoder is GPT-2 transformer
- the fixed-length text embedding is extracted from [EOS] token position
- text token embeddings and image patch embeddings also available
- trained on 256 GPUs for 2 weeks

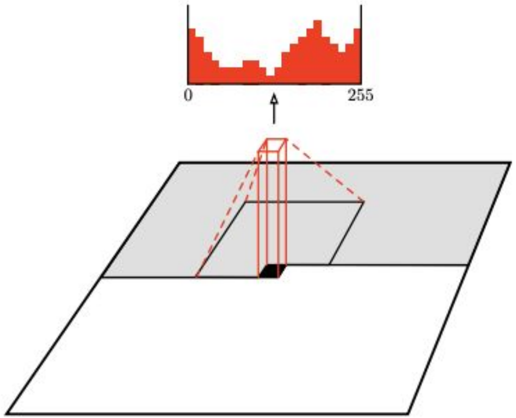
Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. ICML 2021 https://arxiv.org/abs/2103.00020 https://openai.com/research/clip

Image Generation

Autoregressive Models



Pixel Recurrent Neural Networks



Aaron van den Oord, Nal Kalchbrenner, Koray Kavukcuoglu. Pixel Recurrent Neural Networks. ICML 2016

DALL-E 1

- is introduced by OpenAI
- generates 256×256 images from text via dVAE
- autoregressively generates image tokens from textual tokens on a discrete latent space

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, Ilya Sutskever. Zero-Shot Text-to-Image Generation. PMLR 2021
https://arxiv.org/abs/2102.12092

DALL-E 1 Training

1. train encoder and decoder image of image into 32x32 grid of 8k possible code word tokens (dVAE)
2. concatenate encoded text tokens with image tokens into single array
3. train to predict next image token from the preceding tokens (autoregressive transformer)
4. discard the image encoder, keep only image decoder and next token predictor

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, Ilya Sutskever. Zero-Shot Text-to-Image Generation. PMLR 2021
https://arxiv.org/abs/2102.12092

DALL-E 1 Prediction

1. encode input text tokens
2. iteratively predict next image token from the learned codebook
3. decode the image tokens using dVAE decoder
4. select the best image using CLIP model ranker

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, Ilya Sutskever. Zero-Shot Text-to-Image Generation. PMLR 2021
https://arxiv.org/abs/2102.12092

DALL-E 1 Discrete Variational Auto-Encoder (dVAE)

- instead of copying gradients annealing (categorical reparameterization with gumbel-softmax)
- promote codebook utilization using higher KL-divergence weight
- decoder is conv2d, decoder block (4x relu + conv), upsample (tile bigger array), repeat

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, Ilya Sutskever. Zero-Shot Text-to-Image Generation. PMLR 2021
https://arxiv.org/abs/2102.12092

DALL-E 1 Results

(a) a tapir made of accordion. a tapir with the texture of an accordion.

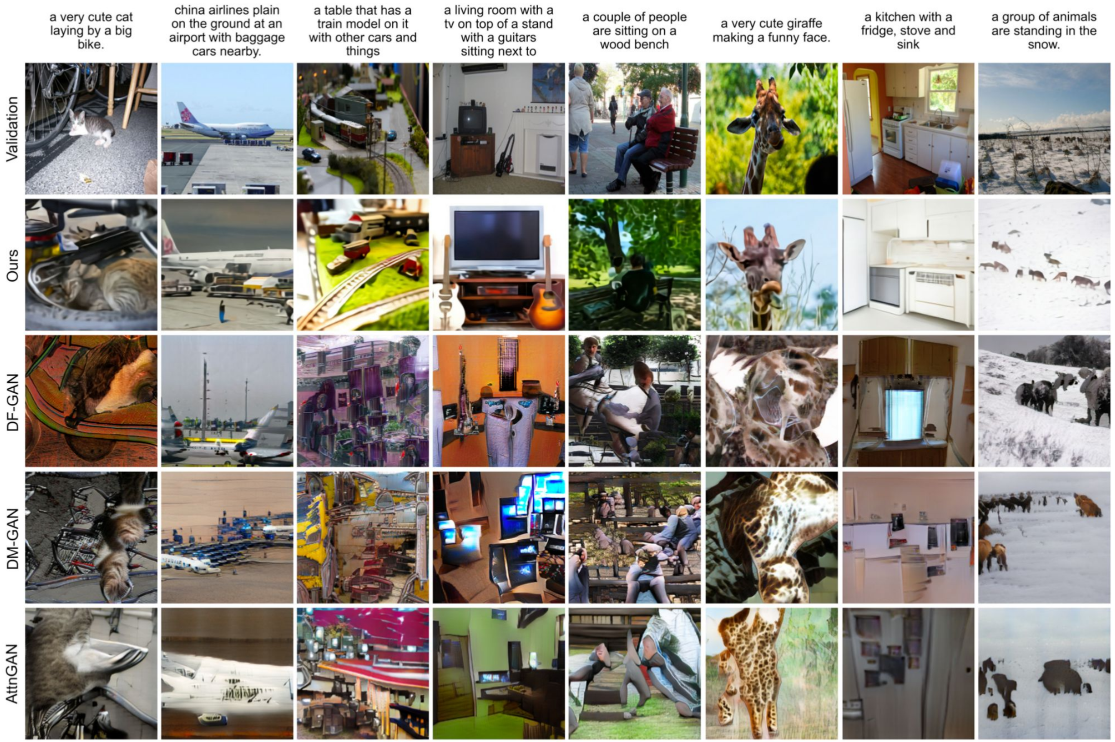
(b) an illustration of a baby hedgehog in a christmas sweater walking a dog

(c) a neon sign that reads “backprop”. a neon sign that reads “backprop”. backprop neon sign

(d) the exact same cat on the top as a sketch on the bottom

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, Ilya Sutskever. Zero-Shot Text-to-Image Generation. PMLR 2021
https://arxiv.org/abs/2102.12092

DALL-E 1 Results



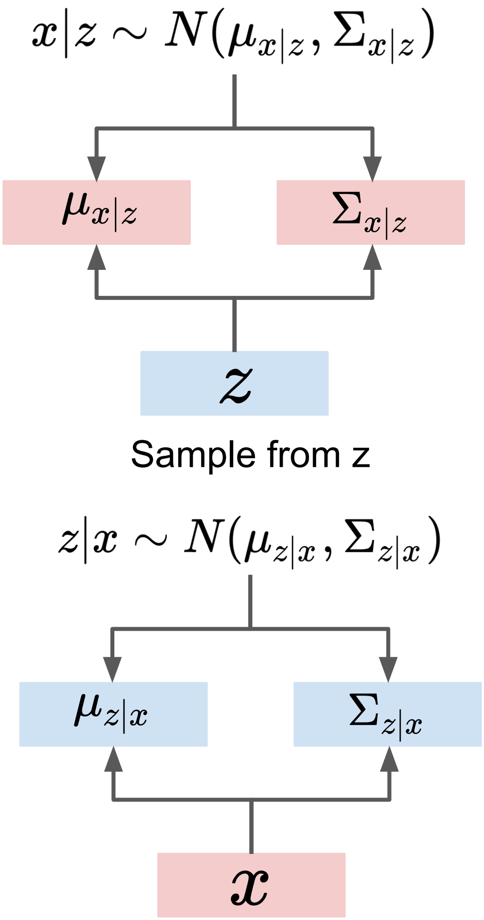
Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, Ilya Sutskever. Zero-Shot Text-to-Image Generation. PMLR 2021
https://arxiv.org/abs/2102.12092

Variational Autoencoders (VAE)

Train by maximize the **variational lower bound**.

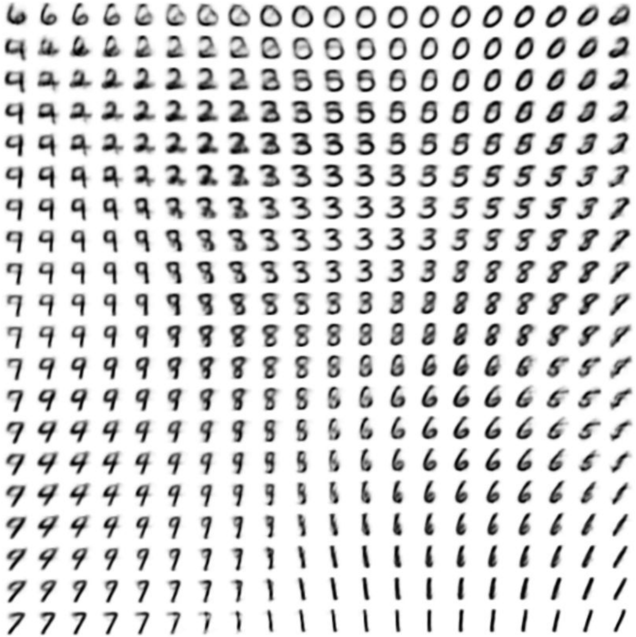
$$\boxed{E_{z\sim q_\phi(z|x)}[\log p_\Theta(x|z)]} - \boxed{KL(q_\phi(z|x), p(z))}$$

1. The input is **encoded** as distribution over the latent space
2. **Encoder output should match prior p(z)**
3. A point from the latent space is sampled from that distribution
4. The sampled point is **decoded**
5. **The reconstruction error is computed**



Variational Autoencoders (VAE)

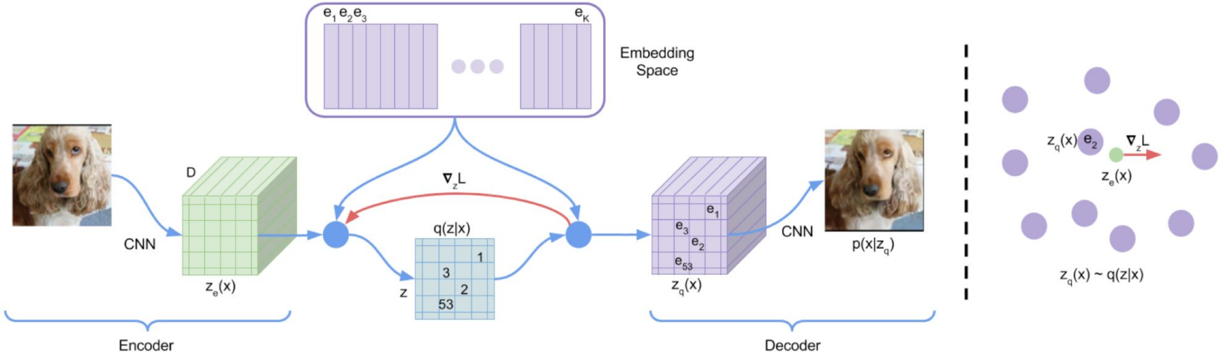
Learned data manifold for generative models with two-dimensional latent space:



Diederik P Kingma, Max Welling. Auto-Encoding Variational Bayes, ICLR 2014 https://arxiv.org/abs/1312.6114

Discrete Variational Auto-Encoder (dVAE)

- introduced in VQ-VAE 1 [1] and VQ-VAE-2 [2]
- image encoder maps to latent 32x32 grid of embeddings
- vector quantization maps to 8k code words
- decoder maps from quantized grid to the image
- copy gradients from decoder input z to the encoder output



[1] Aaron van den Oord, Oriol Vinyals, Koray Kavukcuoglu. Neural Discrete Representation Learning. NeurIPS 2017 https://arxiv.org/abs/1711.00937

[2] Ali Razavi, Aaron van den Oord, Oriol Vinyals. Generating Diverse High-Fidelity Images with VQ-VAE-2. NeurIPS 2019 https://arxiv.org/abs/1906.00446

Generative Adversarial Networks

- Setup: Assume we have data x_i drawn from distribution $p_{data}(x)$. Want to sample from p_{data}.

Generative Adversarial Networks

- Setup: Assume we have data x_i drawn from distribution $p_{data}(x)$. Want to sample from p_{data}.
- Idea: Introduce a latent variable z with simple prior $p(z)$.

Generative Adversarial Networks

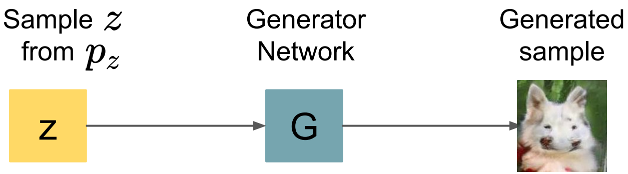
- Setup: Assume we have data x_i drawn from distribution $p_{data}(x)$. Want to sample from p_{data}.
- Idea: Introduce a latent variable z with simple prior $p(z)$.
- Sample $z \sim p(z)$ and pass to a Generator Network $x = G(z)$

Generative Adversarial Networks

- Setup: Assume we have data x_i drawn from distribution $p_{data}(x)$. Want to sample from p_{data}.
- Idea: Introduce a latent variable z with simple prior $p(z)$.
- Sample $z \sim p(z)$ and pass to a Generator Network $x = G(z)$
- Then x is a sample from the Generator distribution p_G. Want $p_G = p_{data}$

Generative Adversarial Networks

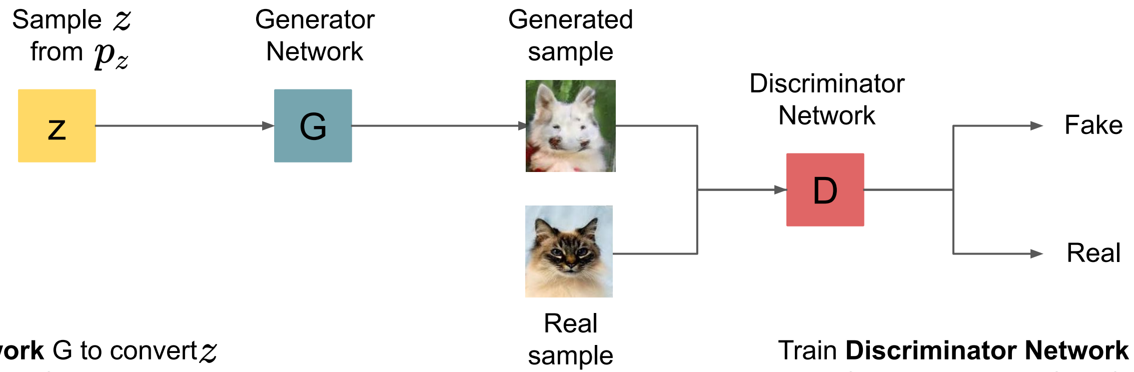
- Setup: Assume we have data x_i drawn from distribution $p_{data}(x)$. Want to sample from p_{data}.
- Idea: Introduce a latent variable z with simple prior $p(z)$.
- Sample $z \sim p(z)$ and pass to a Generator Network $x = G(z)$
- Then x is a sample from the Generator distribution p_G. Want $p_G = p_{data}$



Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Nets. Advances in Neural Information Processing Systems (NeurIPS) 2014.

Generative Adversarial Networks

- Setup: Assume we have data x_i drawn from distribution $p_{data}(x)$. Want to sample from p_{data}.
- Idea: Introduce a latent variable z with simple prior $p(z)$.
- Sample $z \sim p(z)$ and pass to a Generator Network $x = G(z)$
- Then x is a sample from the Generator distribution p_G. Want $p_G = p_{data}$



Train **Generator Network** G to convert z into fake data x sampled from p_G

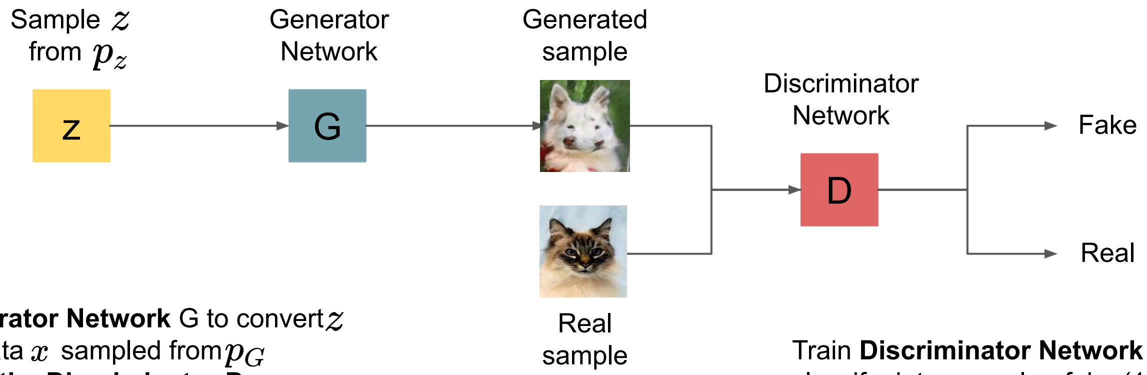
Train **Discriminator Network** D to classify data as real or fake (1/0)

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Nets. Advances in Neural Information Processing Systems (NeurIPS) 2014.

Generative Adversarial Networks: Training Objective

Jointly train generator G and discriminator D with a **minimax game**

$$\min_{G} \max_{D} \left(E_{x\sim p_{data}}[\log D(x)] + E_{z\sim p(z)}[\log(1 - D(G(z)))]\right)$$



Train **Generator Network** G to convert z into fake data x sampled from p_G **by fooling the Discriminator D**

Train **Discriminator Network** D to classify data as real or fake (1/0)

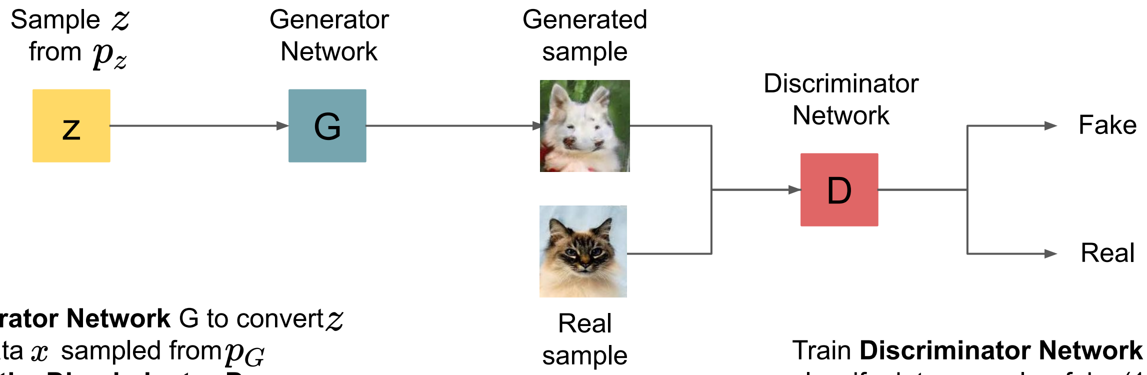
Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Nets. Advances in Neural Information Processing Systems (NeurIPS) 2014.

Generative Adversarial Networks: Training Objective

Jointly train generator G and discriminator D with a **minimax game**

Discriminator wants D(x)=1 for real data

$$\min_{\mathbf{G}} \max_{\mathbf{D}} \left(E_{x \sim p_{data}} [\log \mathbf{D}(x)] + E_{\mathbf{z} \sim p(\mathbf{z})} [\log(1 - \mathbf{D}(\mathbf{G}(\mathbf{z})))] \right)$$



Train **Generator Network** G to convert z into fake data x sampled from p_G **by fooling the Discriminator D**

Train **Discriminator Network** D to classify data as real or fake (1/0)

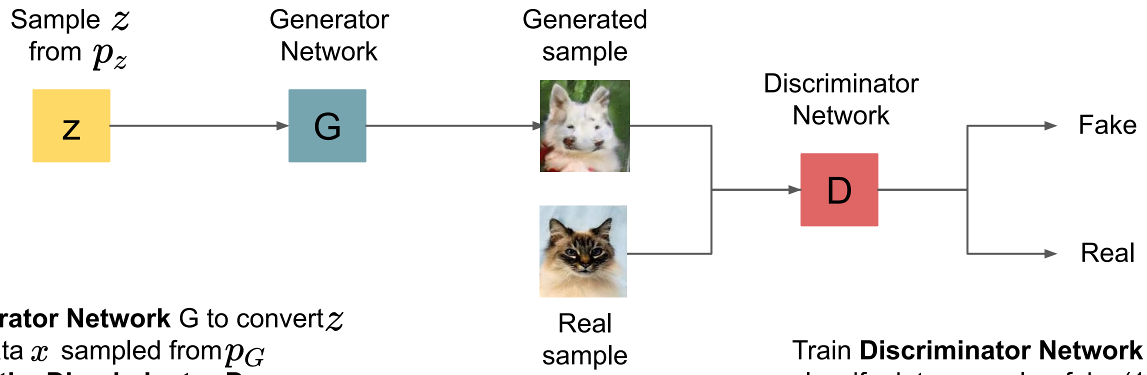
Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Nets. Advances in Neural Information Processing Systems (NeurIPS) 2014.

Generative Adversarial Networks: Training Objective

Jointly train generator G and discriminator D with a **minimax game**

Discriminator wants D(x)=0 for fake data

$$\min_{G} \max_{D} \left(E_{x \sim p_{data}} [\log D(x)] + E_{z \sim p(z)} [\log (1 - D(G(z)))] \right)$$



Train **Generator Network** G to convert z into fake data x sampled from p_G **by fooling the Discriminator D**

Train **Discriminator Network** D to classify data as real or fake (1/0)

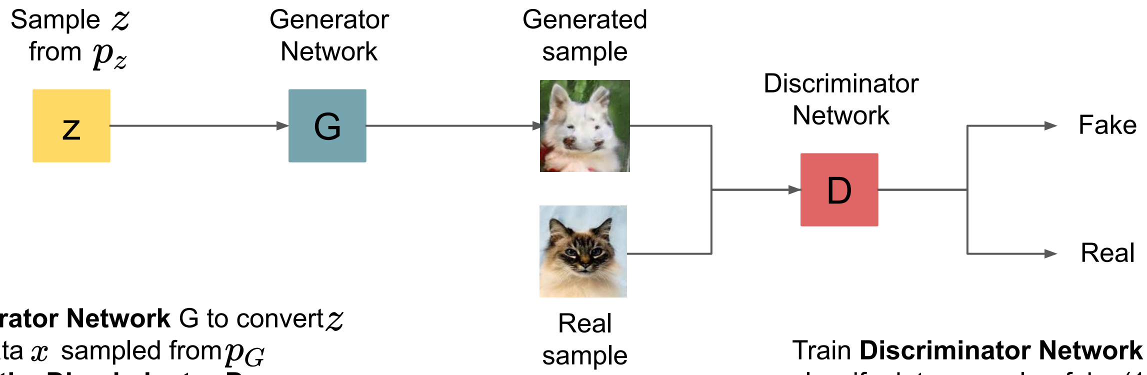
Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Nets. Advances in Neural Information Processing Systems (NeurIPS) 2014.

Generative Adversarial Networks: Training Objective

Jointly train generator G and discriminator D with a **minimax game**

$$\min_{\mathbf{G}} \max_{\mathbf{D}} \left(E_{x \sim p_{data}} [\log \mathbf{D}(x)] + E_{\mathbf{z} \sim p(\mathbf{z})} [\log(1 - \mathbf{D}(\mathbf{G}(\mathbf{z})))] \right)$$

Generator wants D(x)=1 for fake data



Train **Generator Network** G to convert z into fake data x sampled from p_G **by fooling the Discriminator D**

Train **Discriminator Network** D to classify data as real or fake (1/0)

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Nets. Advances in Neural Information Processing Systems (NeurIPS) 2014.

Generative Adversarial Networks: Training Objective

Jointly train generator G and discriminator D with a **minimax game**

$$\min_{\mathbf{G}} \max_{\mathbf{D}} \left(E_{x \sim p_{data}} [\log \mathbf{D}(x)] + E_{\mathbf{z} \sim p(\mathbf{z})} [\log(1 - \mathbf{D}(\mathbf{G}(\mathbf{z})))] \right)$$

$$= \min_{\mathbf{G}} \max_{\mathbf{D}} \mathbf{V}(\mathbf{G}, \mathbf{D})$$

Train G and D using alternating gradient updates:

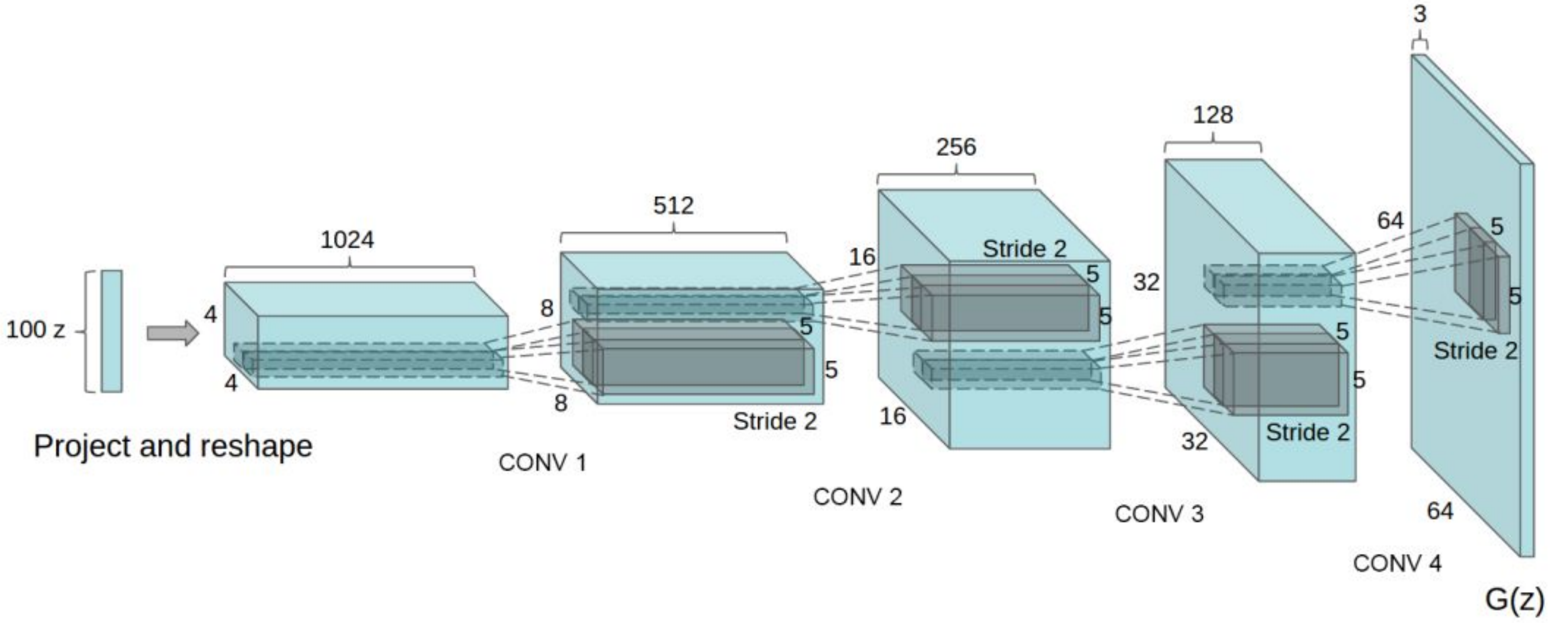
1. Update $\mathbf{D} = \mathbf{D} + \alpha_{\mathbf{D}} \frac{\delta \mathbf{V}}{\delta \mathbf{D}}$
2. Update $\mathbf{G} = \mathbf{G} - \alpha_{\mathbf{G}} \frac{\delta \mathbf{V}}{\delta \mathbf{G}}$

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Nets. Advances in Neural Information Processing Systems (NeurIPS) 2014.

Generative Adversarial Networks: results

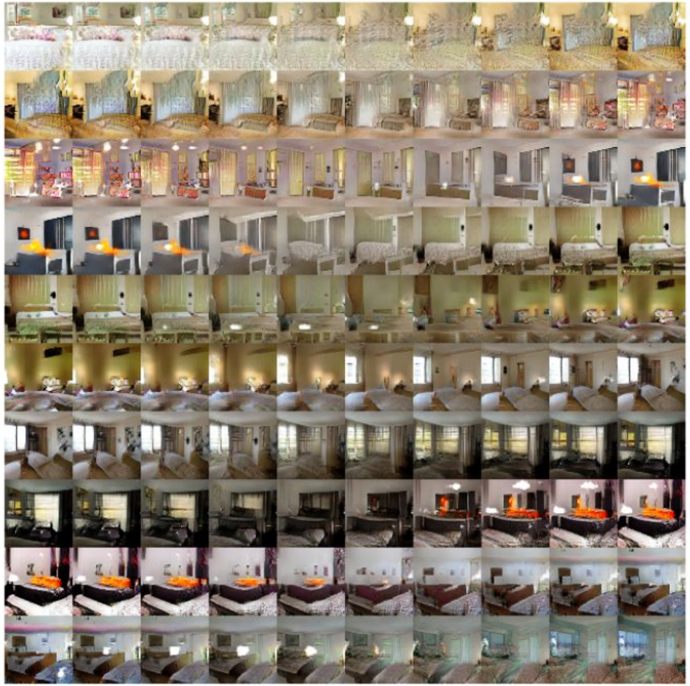
Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Nets. Advances in Neural Information Processing Systems (NeurIPS) 2014.

Generative Adversarial Networks: DC-GAN



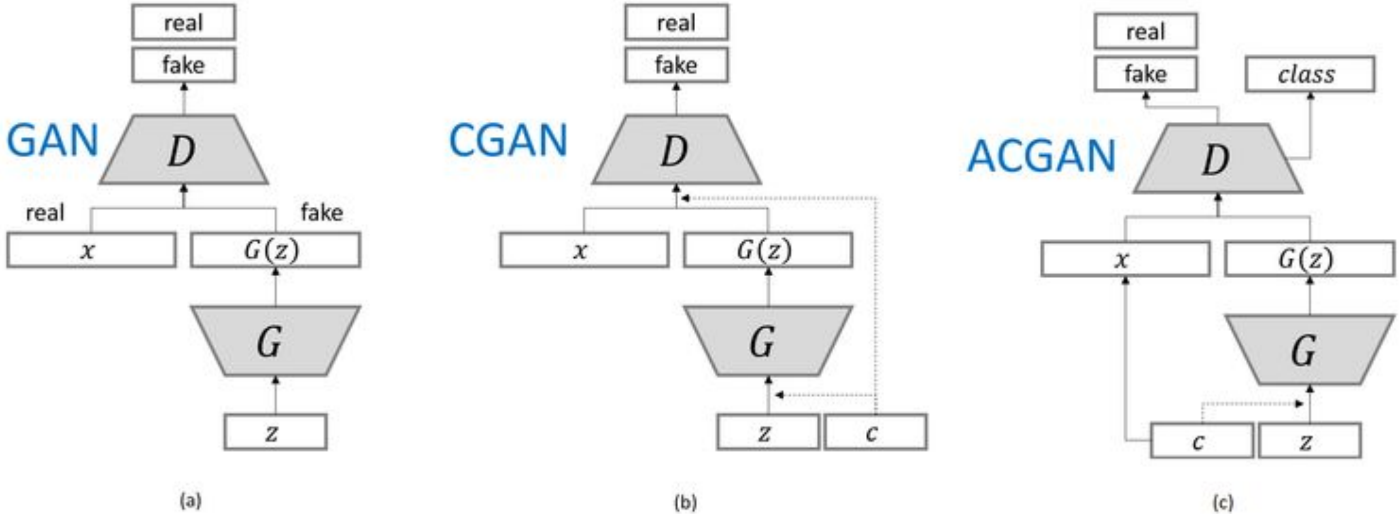
Alec Radford, Luke Metz, Soumith Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. ICLR 2016

Generative Adversarial Networks: Interpolation



Alec Radford, Luke Metz, Soumith Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. ICLR 2016

Conditional GANs



[b] Mehdi Mirza, Simon Osindero. Conditional Generative Adversarial Nets. 2014

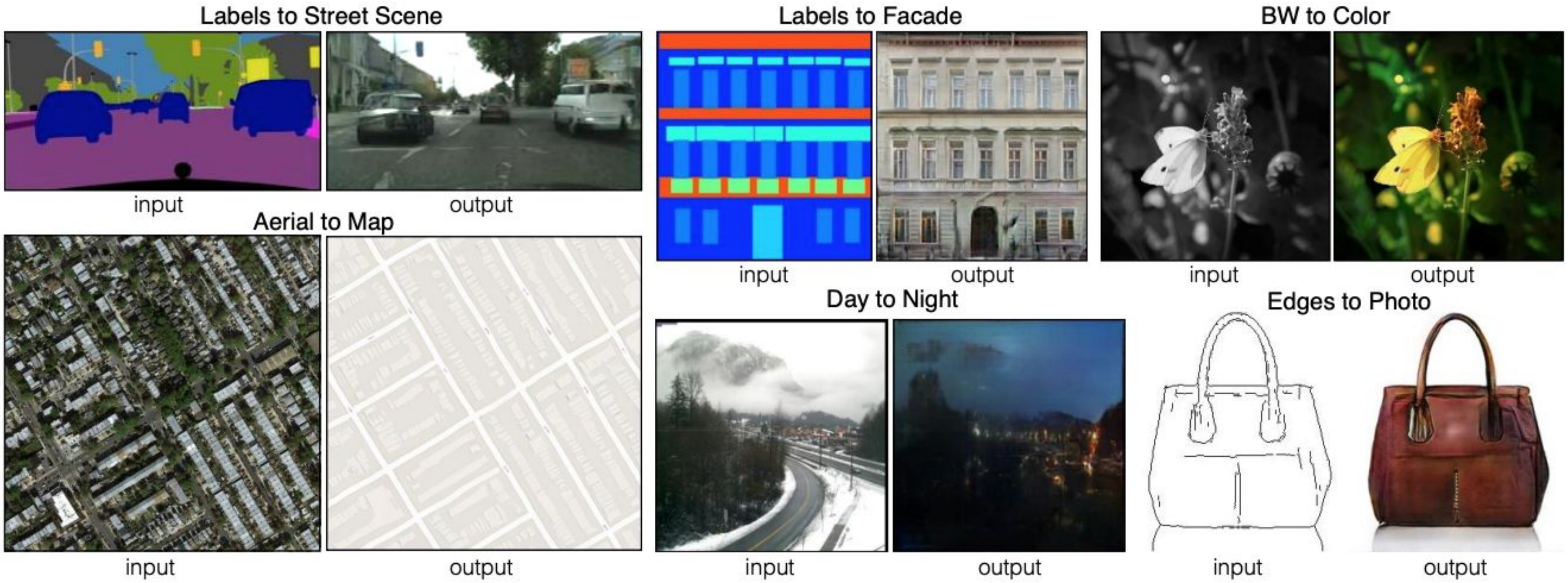
[c] Augustus Odena, Christopher Olah, Jonathon Shlens. Conditional Image Synthesis With Auxiliary Classifier GANs. ICML 2016

Conditional GANs

monarch butterfly | goldfinch | daisy | redshank | grey whale

Augustus Odena, Christopher Olah, Jonathon Shlens. Conditional Image Synthesis With Auxiliary Classifier GANs. ICML 2016

Image-to-Image Translation: Pix2Pix

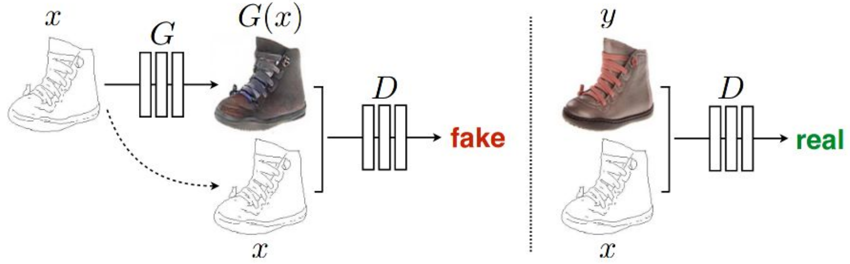


Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. CVPR 2017 https://arxiv.org/abs/1611.07004

Image-to-Image Translation: Pix2Pix

Objective:

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G)$$



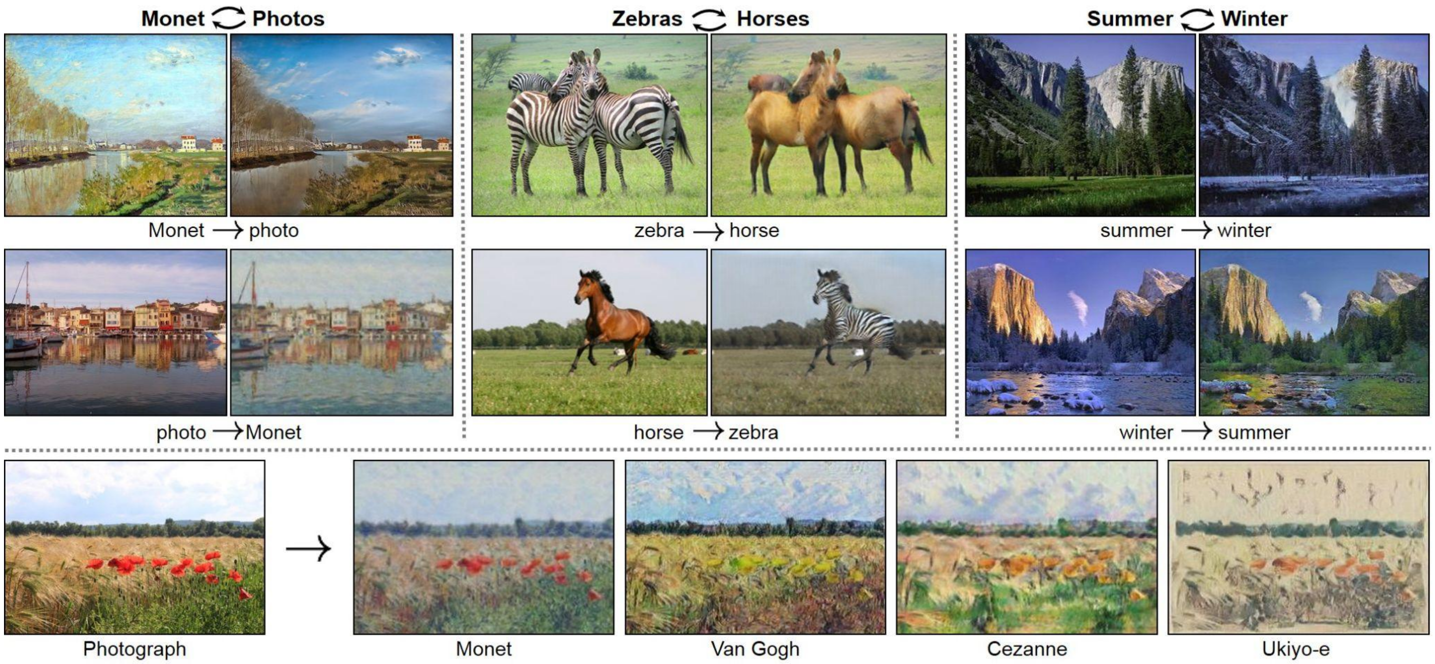
where

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z))]$$

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1]$$

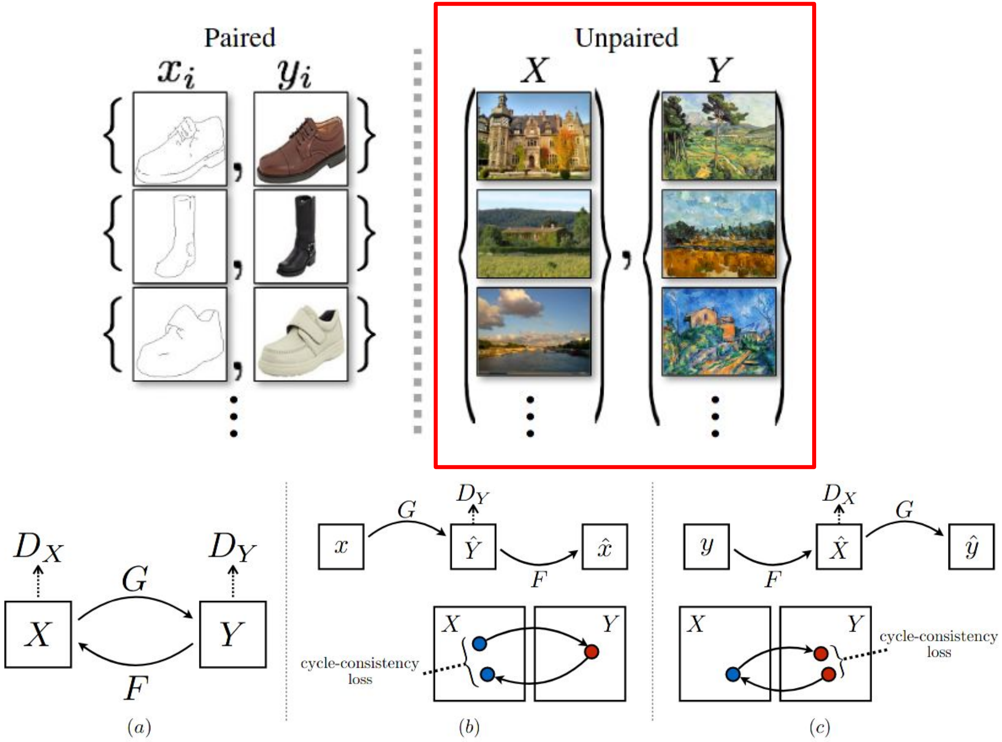
Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. CVPR 2017 https://arxiv.org/abs/1611.07004

Unpaired Image-to-Image Translation: CycleGAN



Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. ICCV 2017 https://arxiv.org/abs/1703.10593

Unpaired Image-to-Image Translation: CycleGAN



Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. ICCV 2017 https://arxiv.org/abs/1703.10593

Unpaired Image-to-Image Translation: CycleGAN

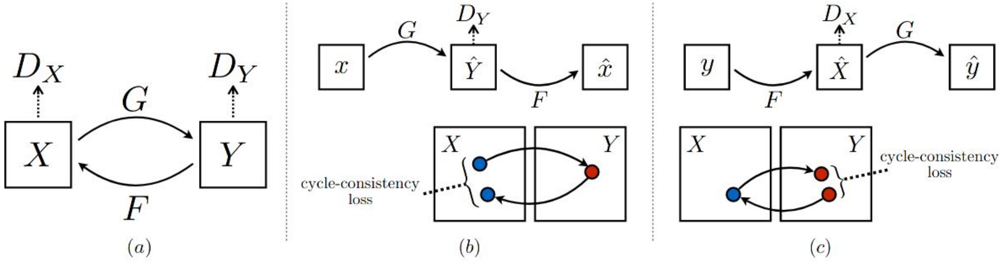
Objective:

$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) =& \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) \\ &+ \lambda \mathcal{L}_{\text{cyc}}(G, F),\end{aligned}$$

where

$$\begin{aligned}\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) &= \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log D_Y(y)] \\ &+ \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x))],\end{aligned}$$

$$\begin{aligned}\mathcal{L}_{\text{cyc}}(G, F) &= \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] \\ &+ \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1].\end{aligned}$$

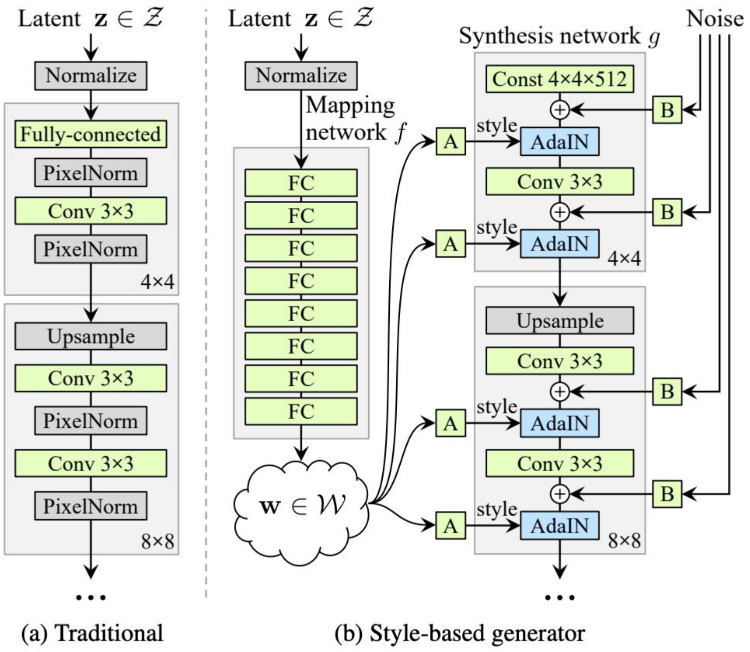


Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. ICCV 2017 https://arxiv.org/abs/1703.10593

Unpaired Image-to-Image Translation: CycleGAN

Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. ICCV 2017 https://arxiv.org/abs/1703.10593

StyleGAN



Tero Karras, Samuli Laine, Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. CVPR 2019

StyleGAN

Tero Karras, Samuli Laine, Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. CVPR 2019

Outline

- Image Generation
    - Diffusion Models: DALL-E 2, Latent Diffusion
- Video Generation
    - Phenaki
    - Everybody Dance Now
    - Make-A-Video

Denoising Diffusion Models.

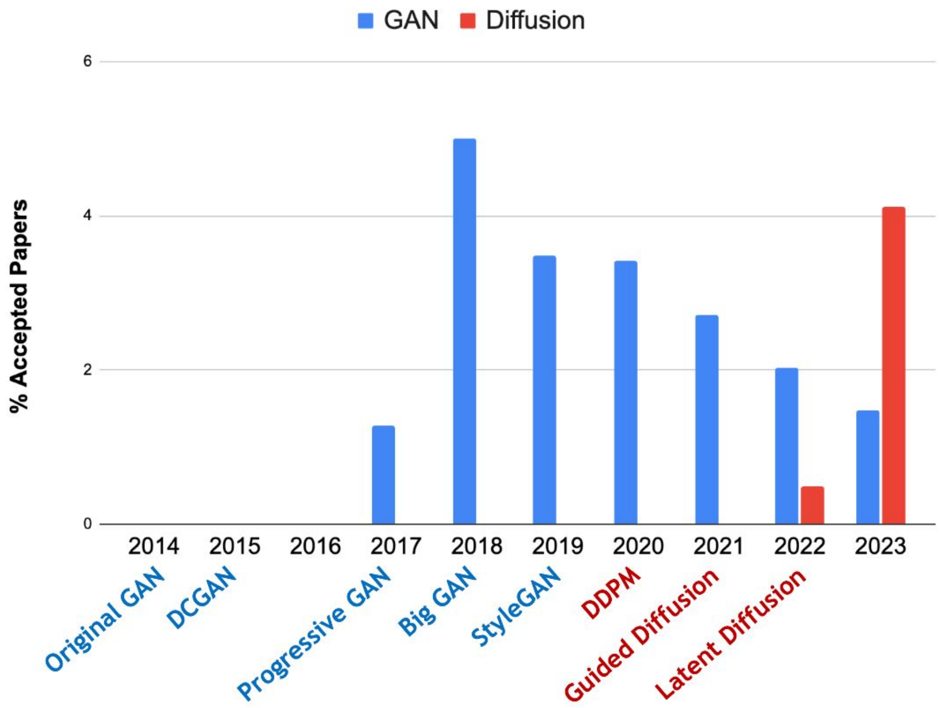
Slide Credits:

Denoising Diffusion Models:

A Generative Learning Big Bang

https://cvpr2023-tutorial-diffusion-models.github.io

We May Not Know Cosmology, But We Know CVPR



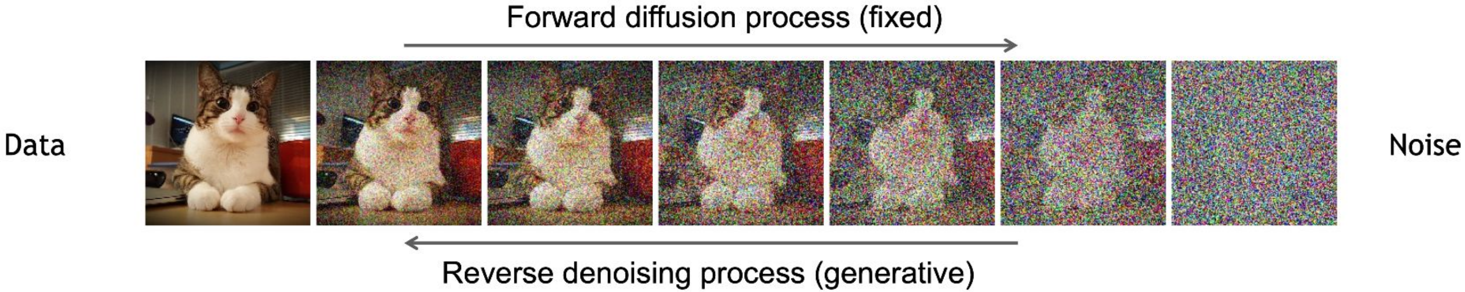
*Disclaimer: We rely on paper titles for counting the number of papers in each topic. Our statistics are likely to be biased.

Denoising Diffusion Models

Learning to generate by denoising

Denoising diffusion models consist of two processes:

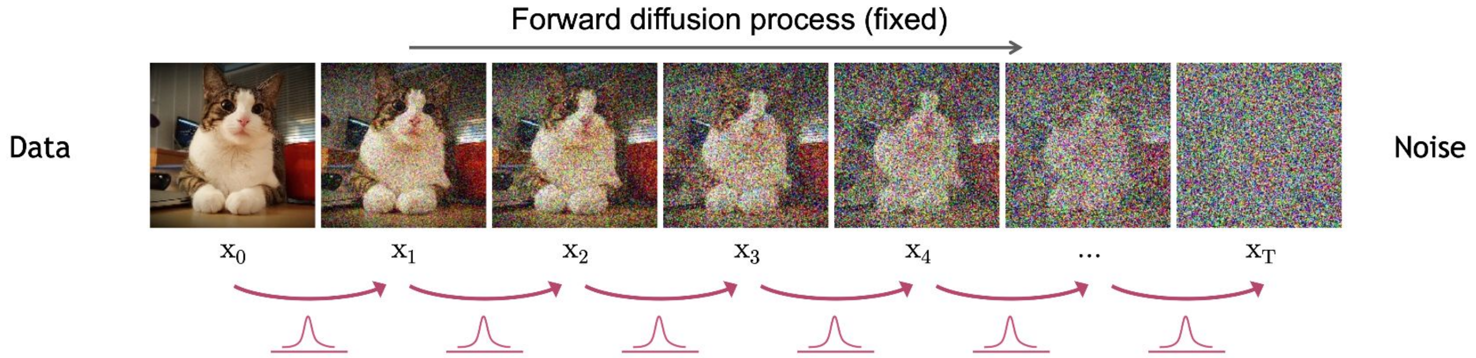
- Forward diffusion process that gradually adds noise to input
- Reverse denoising process that learns to generate data by denoising



Sohl-Dickstein et al., Deep Unsupervised Learning using Nonequilibrium Thermodynamics, ICML 2015
Ho et al., Denoising Diffusion Probabilistic Models, NeurIPS 2020
Song et al., Score-Based Generative Modeling through Stochastic Differential Equations, ICLR 2021

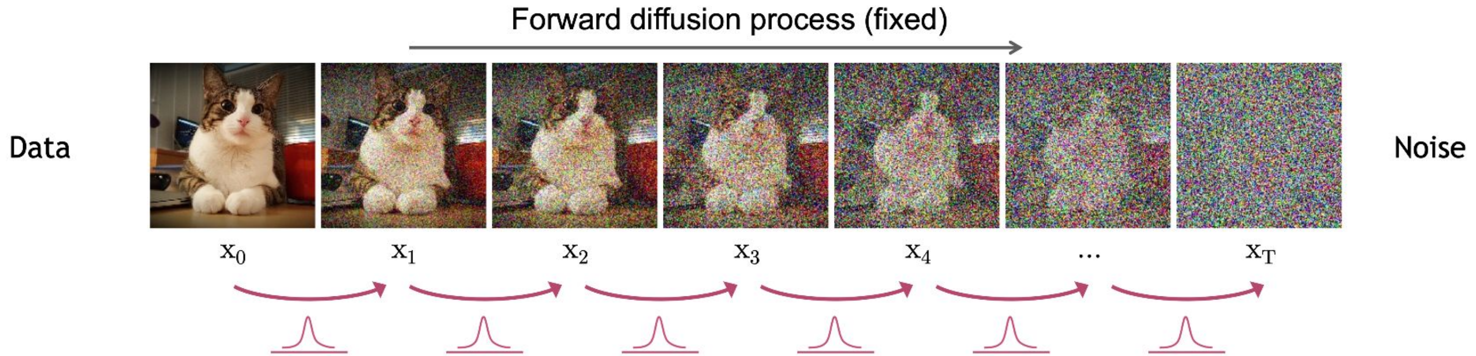
Forward Diffusion Process

The formal definition of the forward process in T steps:



Forward Diffusion Process

The formal definition of the forward process in T steps:

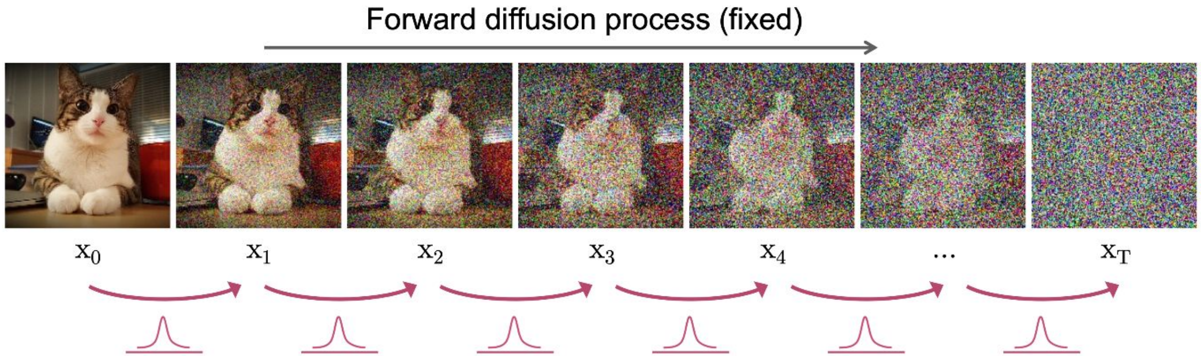


$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x_t}; \sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t\mathbf{I})$$

Forward Diffusion Process

The formal definition of the forward process in T steps:

Data

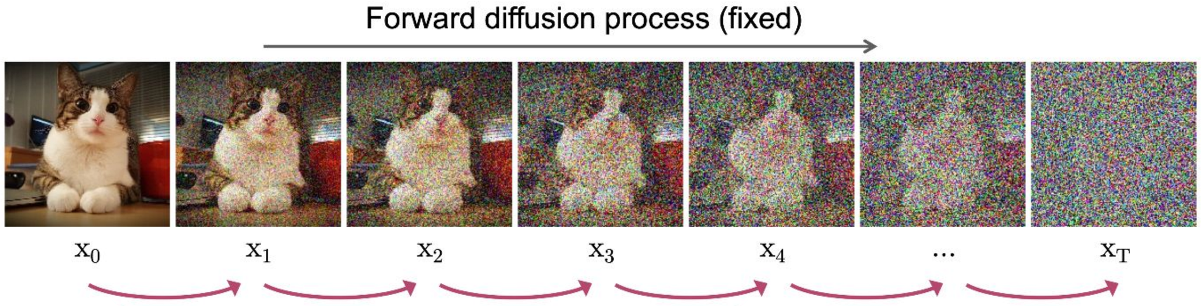


Noise

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x_t}; \sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t\mathbf{I}) \quad \Rightarrow \quad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \quad \text{(joint)}$$

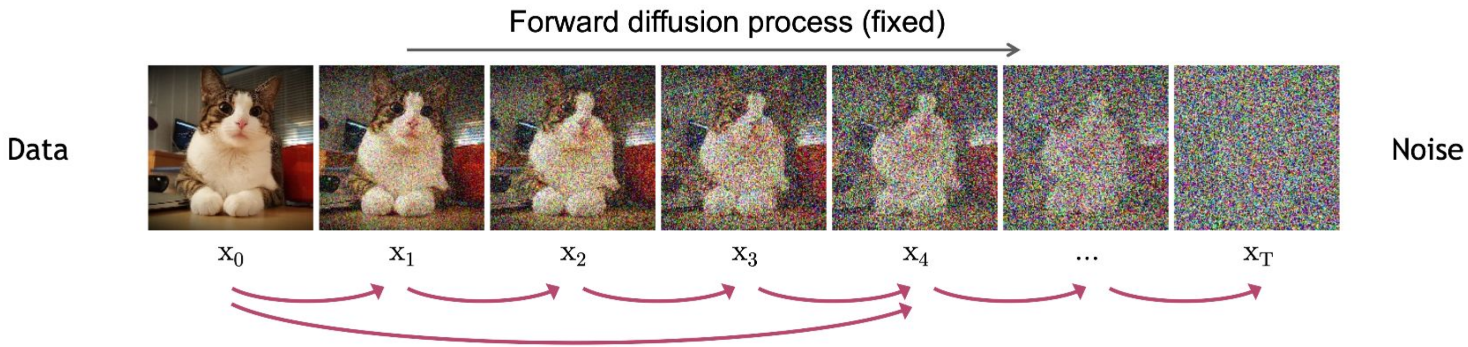
Diffusion Kernel

Data

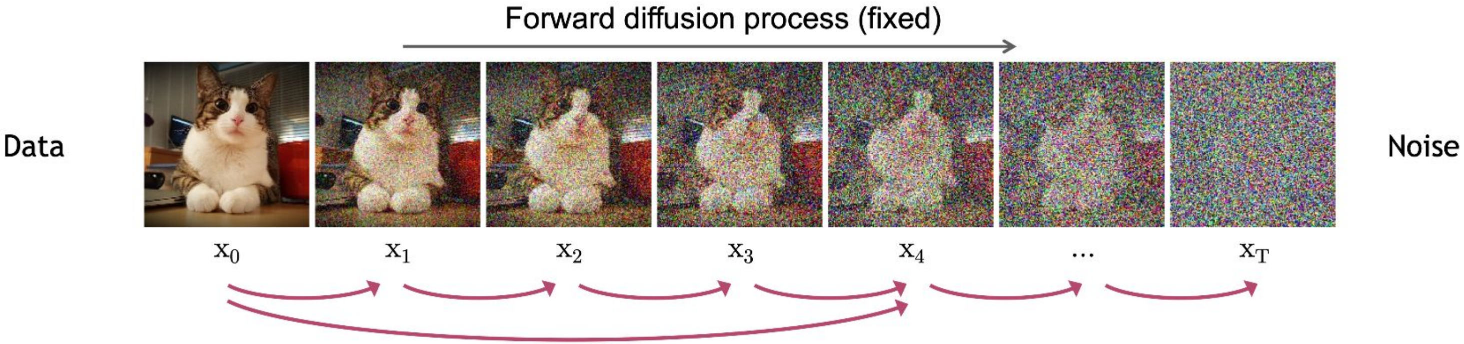


Noise

Diffusion Kernel

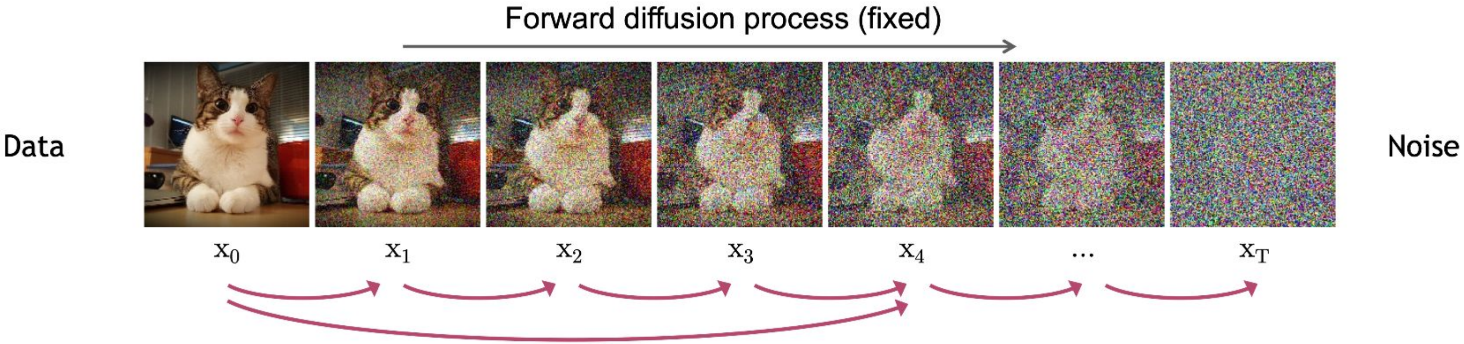


Diffusion Kernel



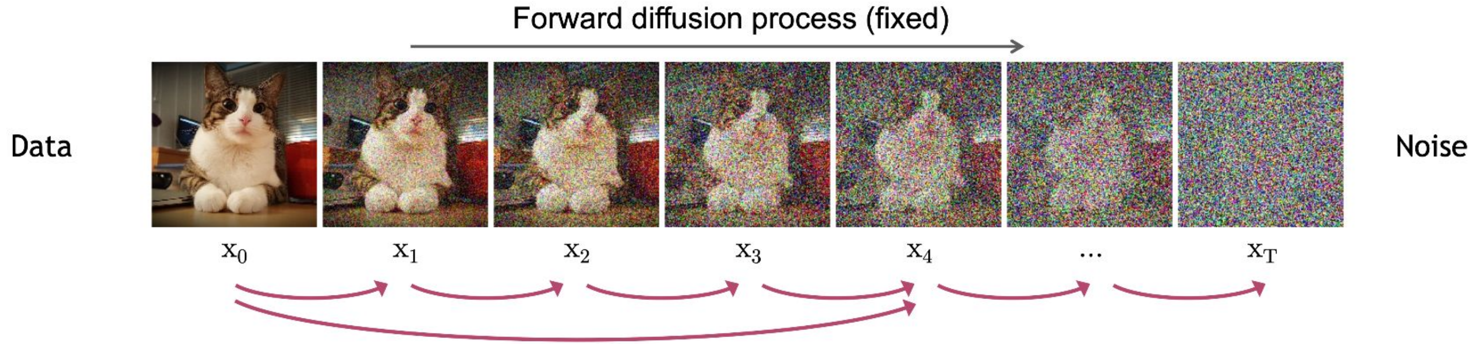
Define $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$

Diffusion Kernel



Define $\bar{\alpha}_t = \prod_{s=1}^{t} (1 - \beta_s)$ ➡ $q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I}))$ (Diffusion Kernel)

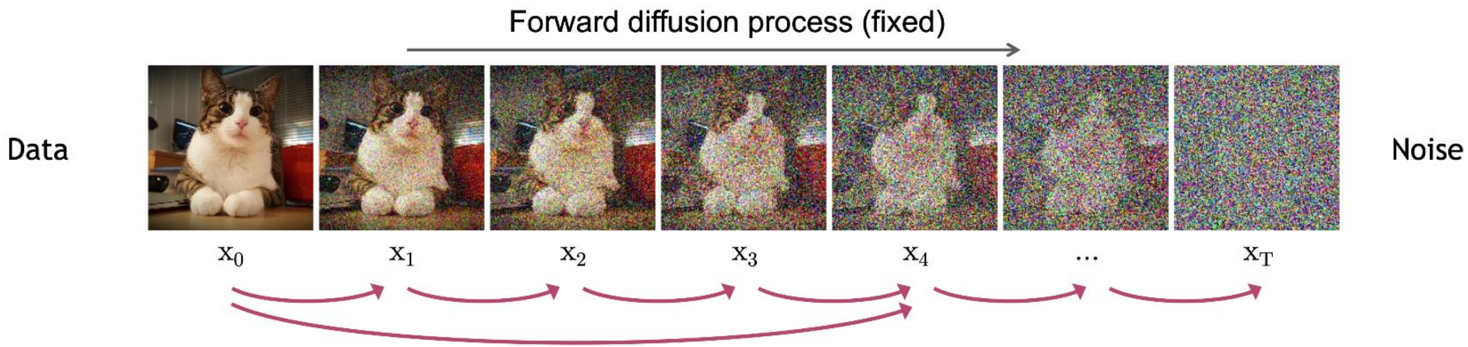
Diffusion Kernel



Define $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$ ➡ $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}))$ (Diffusion Kernel)

For sampling: $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\,\epsilon$ where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

Diffusion Kernel



Define $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$ ➡ $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}))$ (Diffusion Kernel)

For sampling: $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\,\epsilon$ where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

β_t values schedule (i.e., the noise schedule) is designed such that $\bar{\alpha}_T \rightarrow 0$ and $q(\mathbf{x}_T|\mathbf{x}_0) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}))$

What happens to a distribution in the forward diffusion?

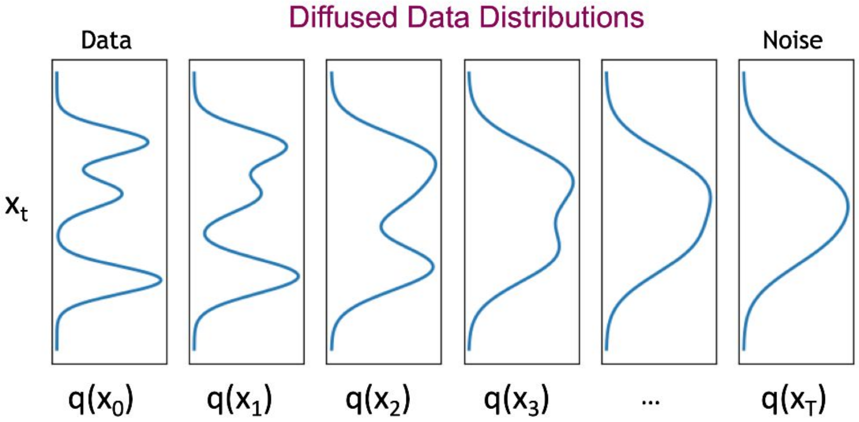
So far, we discussed the diffusion kernel $q(\mathbf{x}_t|\mathbf{x}_0)$ but what about $q(\mathbf{x}_t)$?

$$\underbrace{q(\mathbf{x}_t)}_{\substack{\text{Diffused} \\ \text{data dist.}}} = \int \underbrace{q(\mathbf{x}_0, \mathbf{x}_t)}_{\substack{\text{Joint} \\ \text{dist.}}} \, d\mathbf{x}_0 = \int \underbrace{q(\mathbf{x}_0)}_{\substack{\text{Input} \\ \text{data dist.}}} \, \underbrace{q(\mathbf{x}_t|\mathbf{x}_0)}_{\substack{\text{Diffusion} \\ \text{kernel}}} \, d\mathbf{x}_0$$

What happens to a distribution in the forward diffusion?

So far, we discussed the diffusion kernel $q(\mathbf{x}_t|\mathbf{x}_0)$ but what about $q(\mathbf{x}_t)$?

$$\underbrace{q(\mathbf{x}_t)}_{\text{Diffused data dist.}} = \int \underbrace{q(\mathbf{x}_0, \mathbf{x}_t)}_{\text{Joint dist.}} \, d\mathbf{x}_0 = \int \underbrace{q(\mathbf{x}_0)}_{\text{Input data dist.}} \, \underbrace{q(\mathbf{x}_t|\mathbf{x}_0)}_{\text{Diffusion kernel}} \, d\mathbf{x}_0$$

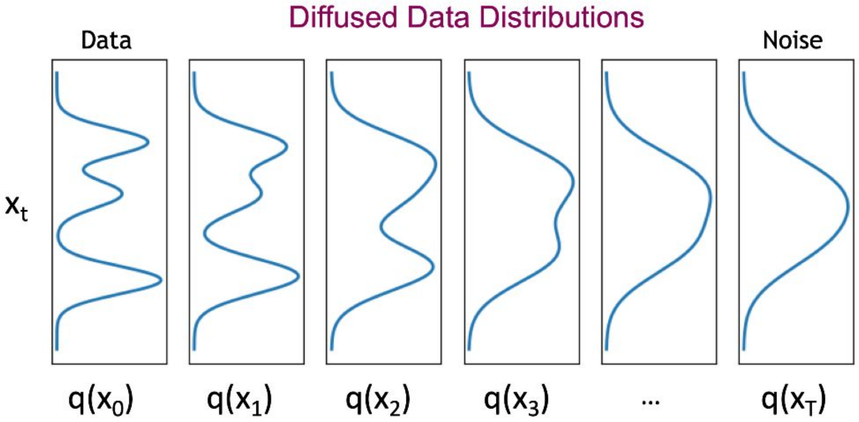


What happens to a distribution in the forward diffusion?

So far, we discussed the diffusion kernel $q(\mathbf{x}_t|\mathbf{x}_0)$ but what about $q(\mathbf{x}_t)$?

$$\underbrace{q(\mathbf{x}_t)}_{\substack{\text{Diffused} \\ \text{data dist.}}} = \int \underbrace{q(\mathbf{x}_0, \mathbf{x}_t)}_{\substack{\text{Joint} \\ \text{dist.}}}\, d\mathbf{x}_0 = \int \underbrace{q(\mathbf{x}_0)}_{\substack{\text{Input} \\ \text{data dist.}}}\, \underbrace{q(\mathbf{x}_t|\mathbf{x}_0)}_{\substack{\text{Diffusion} \\ \text{kernel}}}\, d\mathbf{x}_0$$

The diffusion kernel is Gaussian convolution.

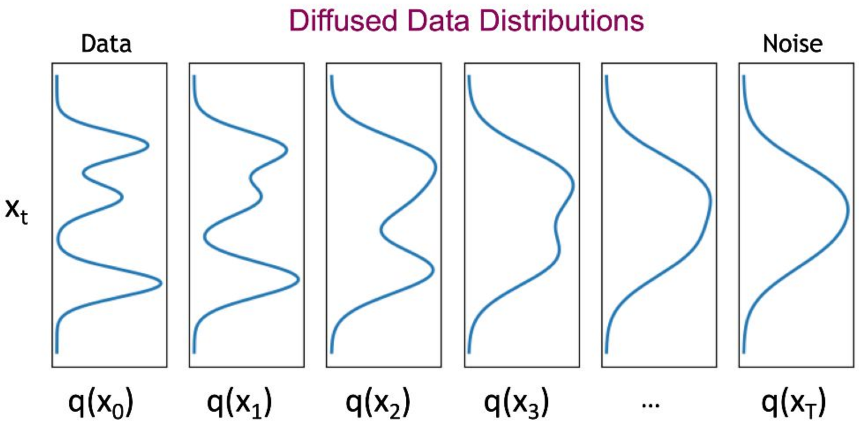


What happens to a distribution in the forward diffusion?

So far, we discussed the diffusion kernel $q(\mathbf{x}_t|\mathbf{x}_0)$ but what about $q(\mathbf{x}_t)$?

$$\underbrace{q(\mathbf{x}_t)}_{\text{Diffused data dist.}} = \int \underbrace{q(\mathbf{x}_0, \mathbf{x}_t)}_{\text{Joint dist.}} \, d\mathbf{x}_0 = \int \underbrace{q(\mathbf{x}_0)}_{\text{Input data dist.}} \, \underbrace{q(\mathbf{x}_t|\mathbf{x}_0)}_{\text{Diffusion kernel}} \, d\mathbf{x}_0$$

The diffusion kernel is Gaussian convolution.



We can sample $\mathbf{x}_t \sim q(\mathbf{x}_t)$ by first sampling $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ and then sampling $\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0)$ (i.e., ancestral sampling).

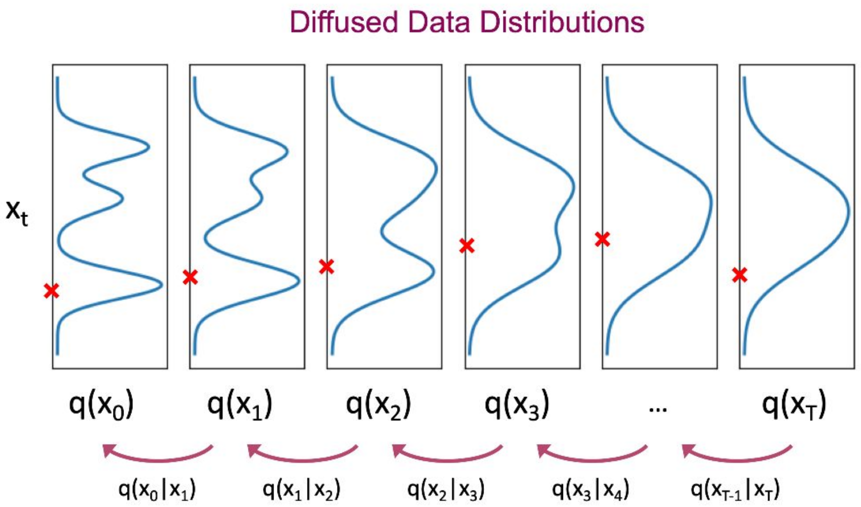
Generative Learning by Denoising

Recall, that the diffusion parameters are designed such that $q(\mathbf{x}_T) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}))$

Generation:

Sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$

Iteratively sample $\mathbf{x}_{t-1} \sim \underbrace{q(\mathbf{x}_{t-1}|\mathbf{x}_t)}_{\text{True Denoising Dist.}}$



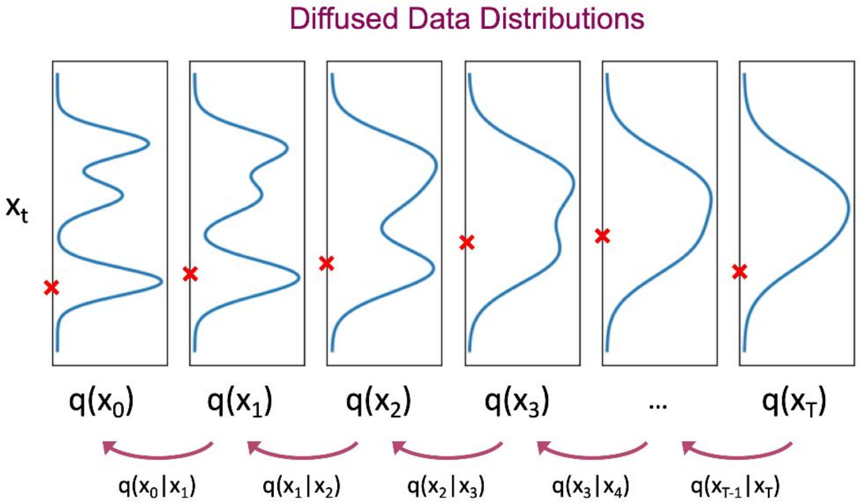
Generative Learning by Denoising

Recall, that the diffusion parameters are designed such that $q(\mathbf{x}_T) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}))$

Generation:

Sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$

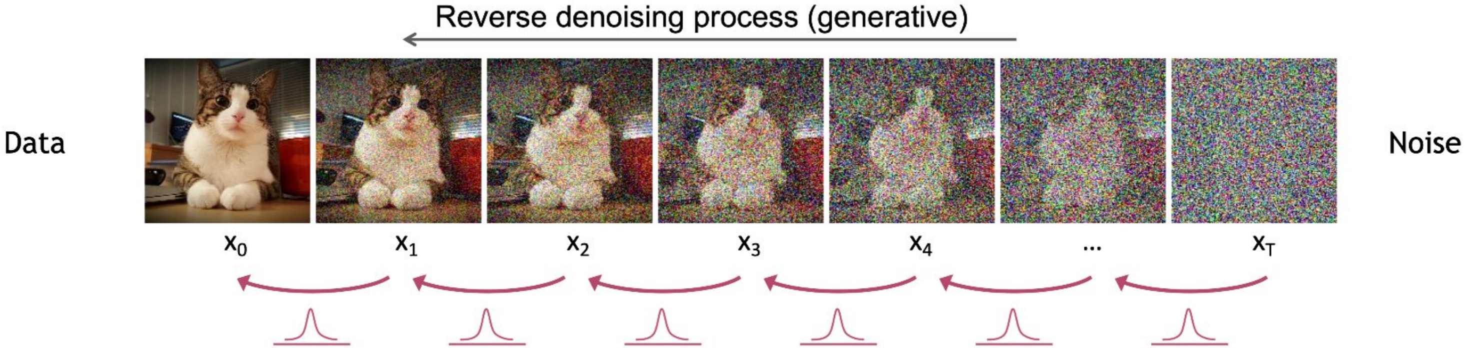
Iteratively sample $\mathbf{x}_{t-1} \sim \underbrace{q(\mathbf{x}_{t-1}|\mathbf{x}_t)}_{\text{True Denoising Dist.}}$



Can we approximate $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$? Yes, we can use a Normal distribution if β_t is small in each forward diffusion step.

Reverse Denoising Process

Formal definition of forward and reverse processes in T steps:

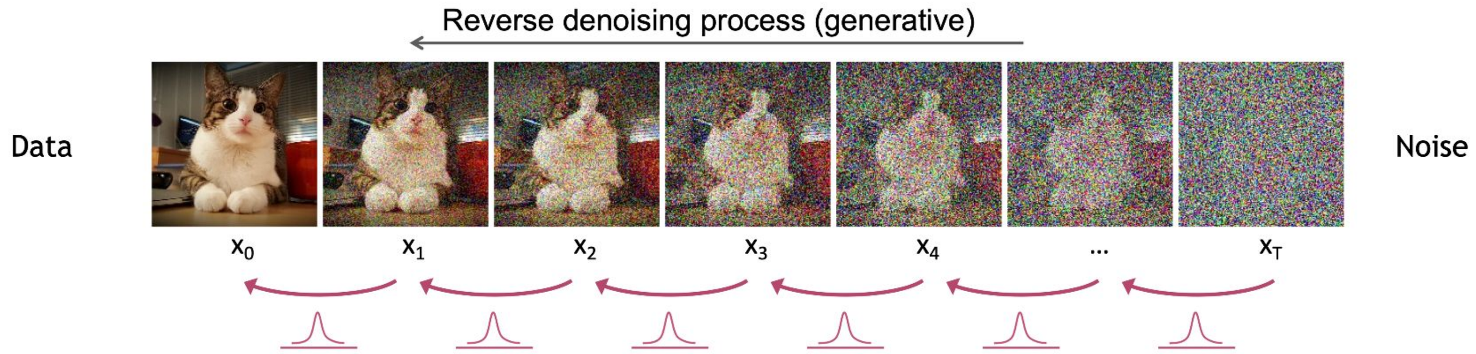


$$p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_t^2\mathbf{I})$$

Reverse Denoising Process

Formal definition of forward and reverse processes in T steps:

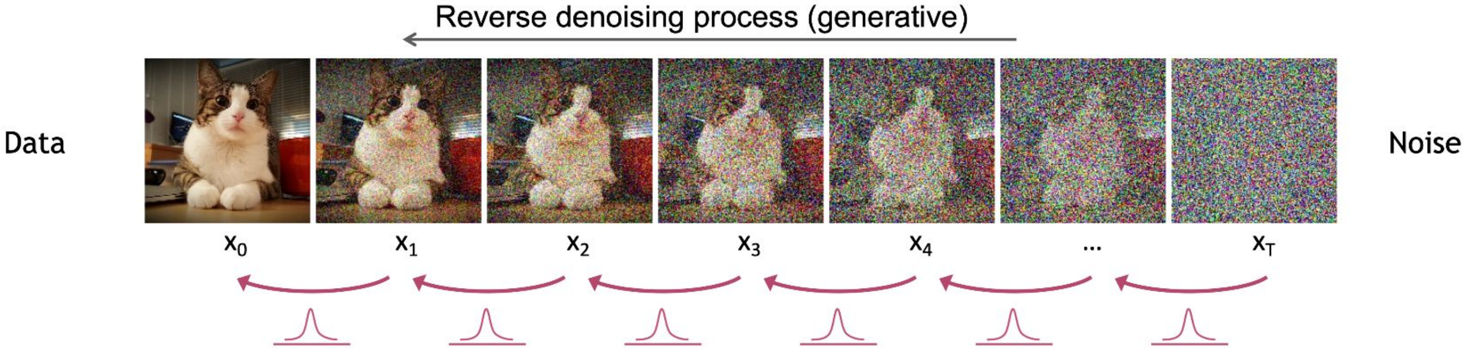


$$p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \underbrace{\mu_\theta(\mathbf{x}_t, t)}_{\substack{\text{Trainable network} \\ \text{(U-net, Denoising Autoencoder)}}}, \sigma_t^2\mathbf{I})$$

Reverse Denoising Process

Formal definition of forward and reverse processes in T steps:



$$p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \underbrace{\mu_\theta(\mathbf{x}_t, t)}_{\text{Trainable network (U-net, Denoising Autoencoder)}}, \sigma_t^2\mathbf{I})$$

$$\Rightarrow \quad p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$$

Learning Denoising Model

Variational upper bound

For training, we can form variational upper bound that is commonly used for training variational autoencoders:

$$\mathbb{E}_{q(\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] =: L$$

Learning Denoising Model

Variational upper bound

For training, we can form variational upper bound that is commonly used for training variational autoencoders:

$$\mathbb{E}_{q(\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] =: L$$

Recall that $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\ \mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\ \epsilon$. Ho et al. NeurIPS 2020 parameterized the mean of denoising model via:

$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\ \epsilon_\theta(\mathbf{x}_t, t)\right)$$

Learning Denoising Model

Variational upper bound

For training, we can form variational upper bound that is commonly used for training variational autoencoders:

$$\mathbb{E}_{q(\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] =: L$$

Recall that $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\ \mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\ \epsilon$. Ho et al. NeurIPS 2020 parameterized the mean of denoising model via:

$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\ \epsilon_\theta(\mathbf{x}_t, t)\right)$$

Using a few simple arithmetic operations, we can write down the variational objective as:

$$L = \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), t \sim \mathcal{U}\{1,T\}, \epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I})}\left[\lambda_t ||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\ \mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\ \epsilon, t)||^2\right]$$

Learning Denoising Model

Variational upper bound

For training, we can form variational upper bound that is commonly used for training variational autoencoders:

$$\mathbb{E}_{q(\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] =: L$$

Recall that $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\ \mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\ \epsilon$. Ho et al. NeurIPS 2020 parameterized the mean of denoising model via:

$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\ \epsilon_\theta(\mathbf{x}_t, t)\right)$$

Using a few simple arithmetic operations, we can write down the variational objective as:

$$L = \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0), t \sim \mathcal{U}\{1,T\}, \epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I})}\left[\lambda_t ||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\ \mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\ \epsilon, t)||^2\right]$$

Ho et al. NeurIPS 2020 observe that simply setting λ_t to 1 for all *t* works best in practice.

Summary

Training and Sample Generation

Algorithm 1 Training

1: **repeat**
2: $\quad \mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $\quad t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
4: $\quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: $\quad$ Take gradient descent step on

$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boxed{\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}}, t) \right\|^2$$

6: **until** converged

Summary

Training and Sample Generation

Algorithm 1 Training

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

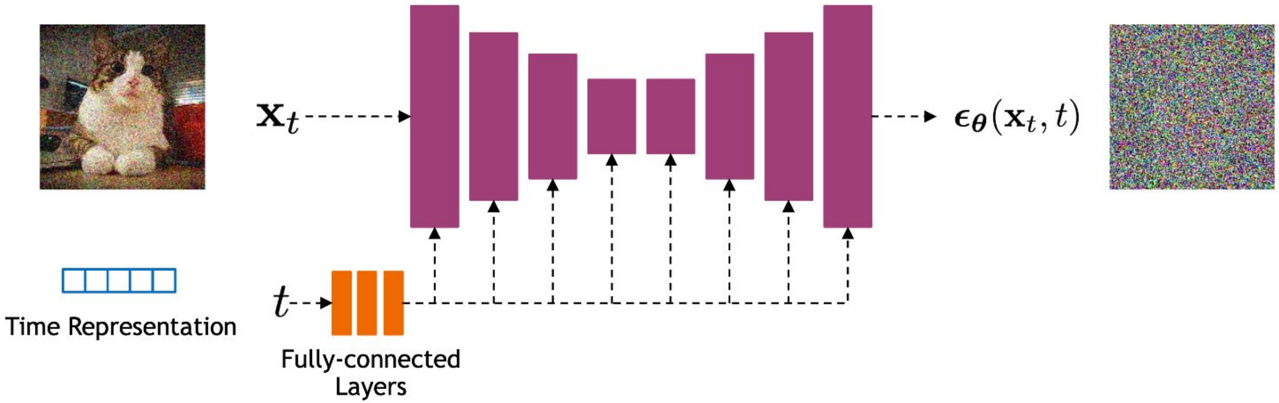
Algorithm 2 Sampling

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

Implementation Considerations

Network Architectures

Diffusion models often use U-Net architectures with ResNet blocks and self-attention layers to represent $\epsilon_\theta(\mathbf{x}_t, t)$

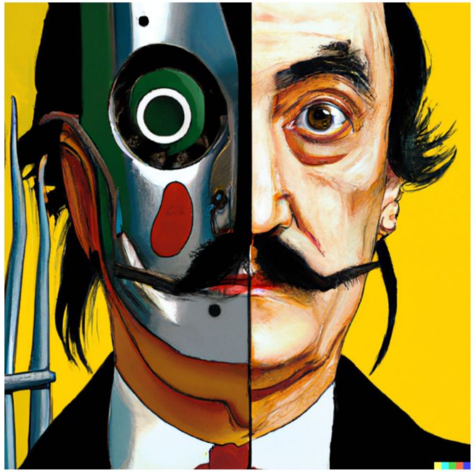


Time representation: sinusoidal positional embeddings or random Fourier features.

Time features are fed to the residual blocks using either simple spatial addition or using adaptive group normalization layers. (see Dharivwal and Nichol NeurIPS 2021)

DALL-E 2

- is introduced by OpenAI
- generates 1024 x 1024 images from text using diffusion models.



vibrant portrait painting of Salvador Dalí with a robotic half face

a shiba inu wearing a beret and black turtleneck

a close up of a handpalm with leaves growing from it

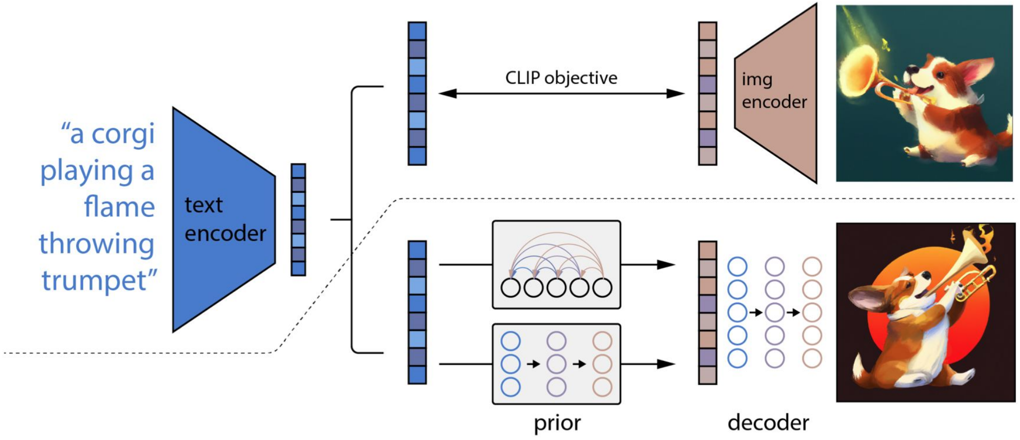
Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, Mark Chen.

Hierarchical Text-Conditional Image Generation with CLIP Latents. 2022 https://arxiv.org/abs/2204.06125

https://openai.com/product/dall-e-2

DALL-E 2

1. generates a CLIP text embedding for text caption
2. “prior” network generates CLIP image embedding from text embedding
3. diffusion decoder generates image from the image embedding



Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, Mark Chen.

Hierarchical Text-Conditional Image Generation with CLIP Latents. 2022 https://arxiv.org/abs/2204.06125

https://openai.com/product/dall-e-2

DALL-E 2 training

Can vary images while preserving style and semantics in the embeddings

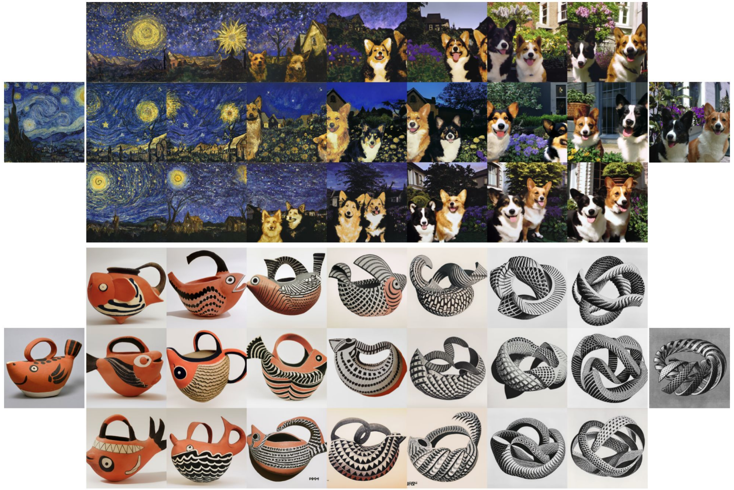


Figure 4: Variations between two images by interpolating their CLIP image embedding and then decoding with a diffusion model. We fix the decoder seed across each row. The intermediate variations naturally blend the content and style from both input images.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, Mark Chen.

Hierarchical Text-Conditional Image Generation with CLIP Latents. 2022 https://arxiv.org/abs/2204.06125

https://openai.com/product/dall-e-2

DALL-E 2 Limitations

A family dining for Christmas

A hyperrealistic man's face smiling with a pair of sunglasses

A green-eyed woman with freckles showing happines

A green-eyed woman with freckles showing disgust

Midjourney

Capture the essence of a young footballer wearing the Paris Saint-Germain cinematic uniform through an extreme close-up portrait, focusing on the intricate details of her face and expressions, very detailed, Octane Render, cinematic, digital art

animated blonde blue eyes love tattoo soft male guy enhanced by ai smile defined --q 2 --s 750

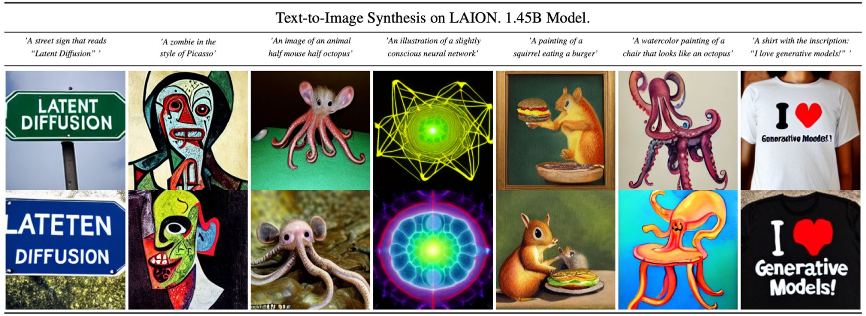
surprised and happy man realistic

Midjourney

eiffel tower in space with an alien on top

model and robot in fashion show, public, cyberpunk,small robot ,red road , cyberpunk,8k rendering

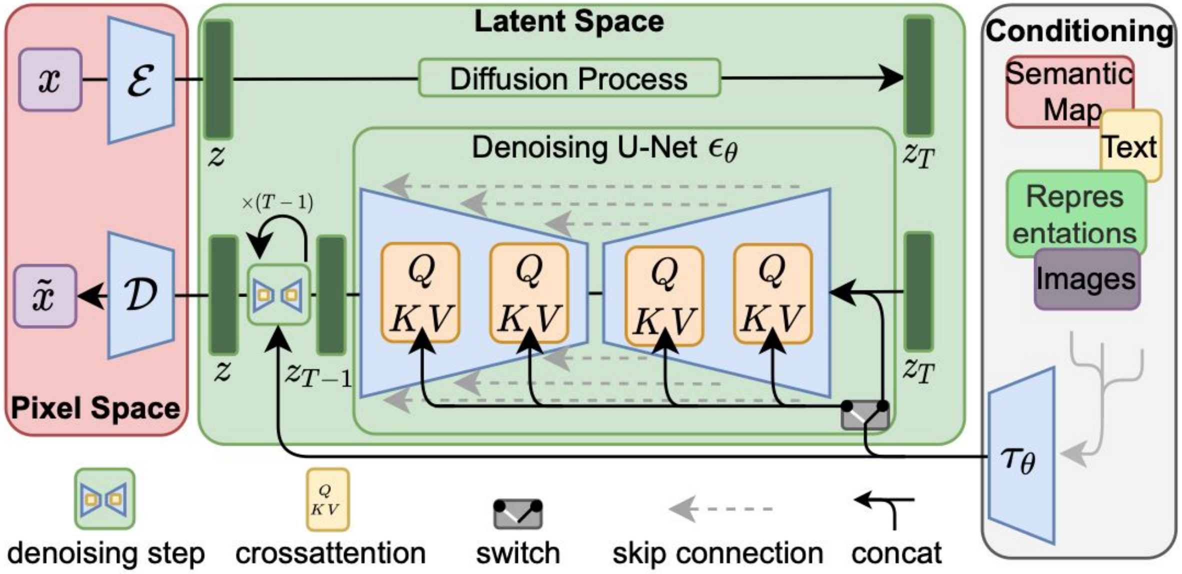
Latent Diffusion Models



Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, Björn Ommer.

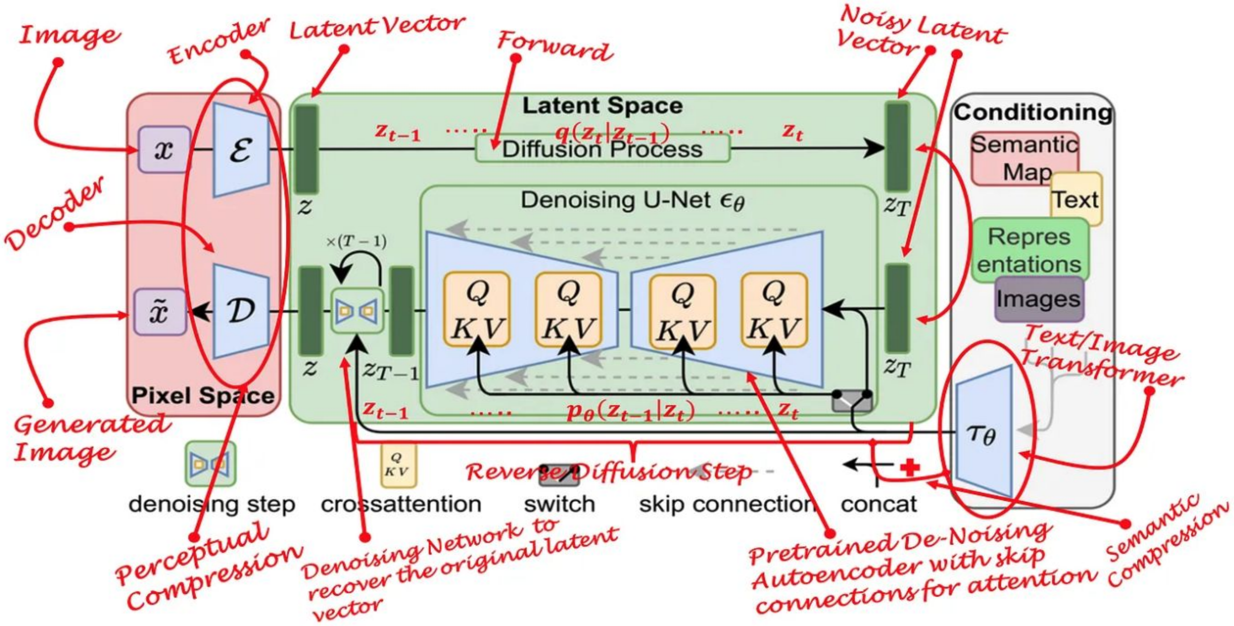
High-Resolution Image Synthesis with Latent Diffusion Models. 2022 https://arxiv.org/abs/2112.10752

Latent Diffusion Models



Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. 2022 https://arxiv.org/abs/2112.10752

Latent Diffusion Models



Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. 2022 https://arxiv.org/abs/2112.10752

Latent Diffusion Models

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, Björn Ommer.

High-Resolution Image Synthesis with Latent Diffusion Models. 2022 https://arxiv.org/abs/2112.10752

Stable Diffusion

Prompt: realistic smiling woman

https://stablediffusionweb.com

Video Generation

Phenaki

The water is magical

Prompts used:
A photorealistic teddy bear is swimming in the ocean at San Francisco
The teddy bear goes under water
The teddy bear keeps swimming under the water with colorful fishes
A panda bear is swimming under water

Chilling on the beach

Prompts used:
A teddy bear diving in the ocean
A teddy bear emerges from the water
A teddy bear walks on the beach
Camera zooms out to the teddy bear in the campfire by the beach

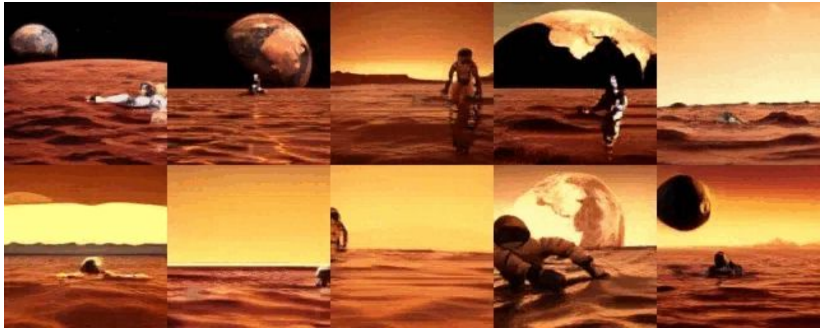
Fireworks on the spacewalk

Prompts used:
Side view of an astronaut is walking through a puddle on mars
The astronaut is dancing on mars
The astronaut walks his dog on mars
The astronaut and his dog watch fireworks

Ruben Villegas, Mohammad Babaeizadeh, Pieter-Jan Kindermans, Hernan Moraldo, Han Zhang, Mohammad Taghi Saffar, Santiago Castro, Julius Kunze, Dumitru Erhan. Phenaki: Variable Length Video Generation From Open Domain Textual Description. 2022 https://arxiv.org/abs/2210.02399
https://phenaki.video

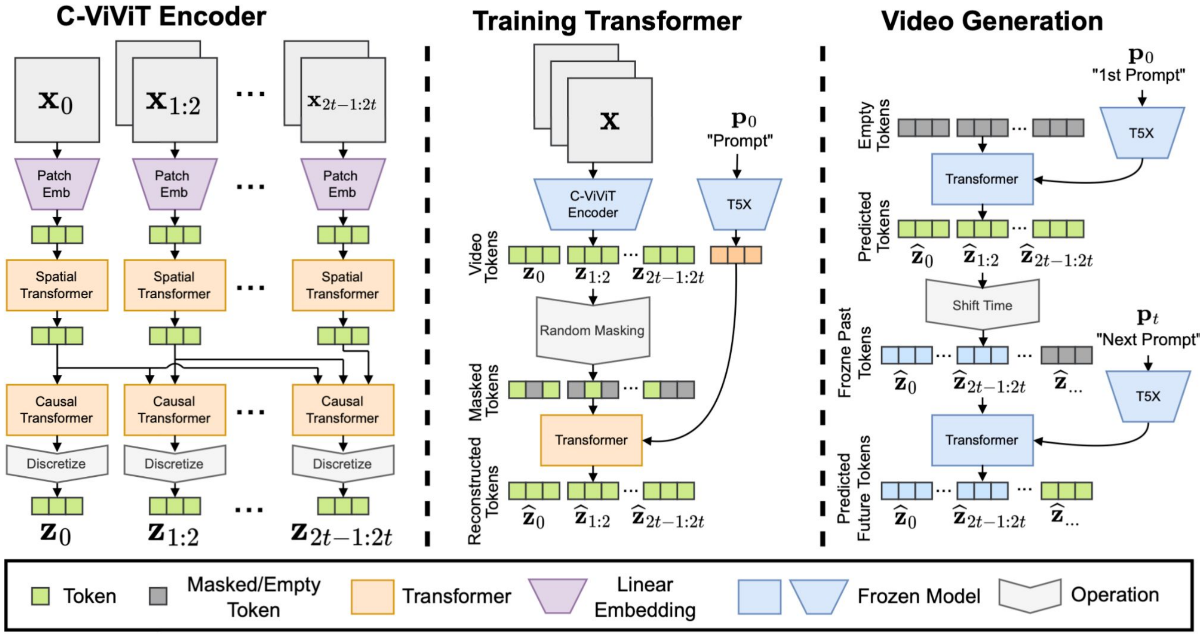
Phenaki






Ruben Villegas, Mohammad Babaeizadeh, Pieter-Jan Kindermans, Hernan Moraldo, Han Zhang, Mohammad Taghi Saffar, Santiago Castro, Julius Kunze, Dumitru Erhan. Phenaki: Variable Length Video Generation From Open Domain Textual Description. 2022 https://arxiv.org/abs/2210.02399
https://phenaki.video

Phenaki

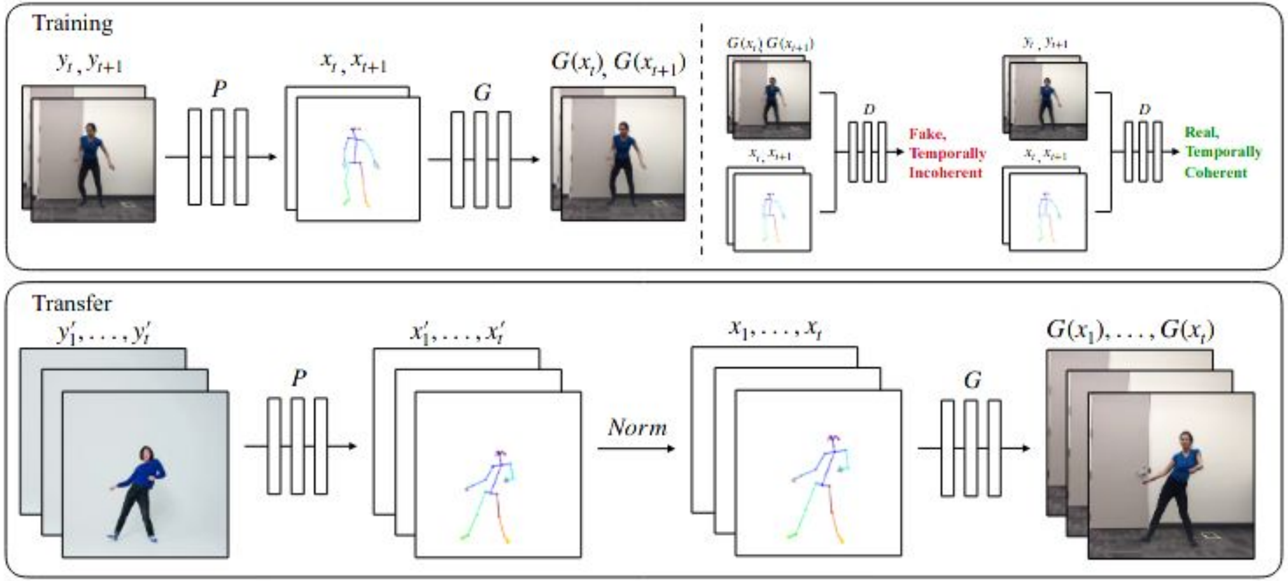


Ruben Villegas, Mohammad Babaeizadeh, Pieter-Jan Kindermans, Hernan Moraldo, Han Zhang, Mohammad Taghi Saffar, Santiago Castro, Julius Kunze, Dumitru Erhan. Phenaki: Variable Length Video Generation From Open Domain Textual Description. 2022 https://arxiv.org/abs/2210.02399
https://phenaki.video

Everybody Dance Now

Caroline Chan, Shiry Ginosar, Tinghui Zhou, Alexei A. Efros. Everybody Dance Now. ICCV 2019

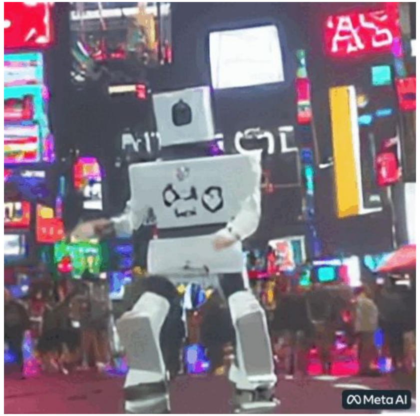
Everybody Dance Now



Caroline Chan, Shiry Ginosar, Tinghui Zhou, Alexei A. Efros. Everybody Dance Now. ICCV 2019

Make-A-Video

A teddy bear painting a portrait



Robot dancing in times square

Cat watching TV with a remote in hand

A fluffy baby sloth with an orange knitted hat trying to figure out a laptop close up highly detailed studio lighting screen reflecting in its eye

Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, Yaniv Taigman.

Make-A-Video: Text-to-Video Generation without Text-Video Data. 2022 https://arxiv.org/abs/2209.14792

https://makeavideo.studio

Make-A-Video architecture

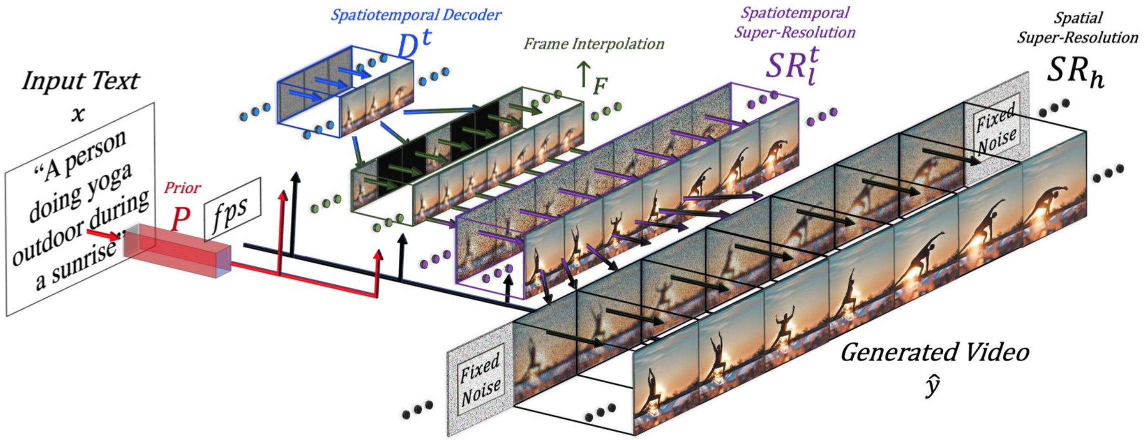


Figure 2: **Make-A-Video high-level architecture.** Given input text x translated by the prior P into an image embedding, and a desired frame rate fps, the decoder D^t generates 16 64×64 frames, which are then interpolated to a higher frame rate by $\uparrow_F$, and increased in resolution to 256×256 by SR_l^t and 768×768 by SR_h, resulting in a high-spatiotemporal-resolution generated video $\hat{y}$.

Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, Yaniv Taigman.

Make-A-Video: Text-to-Video Generation without Text-Video Data. 2022 https://arxiv.org/abs/2209.14792

https://makeavideo.studio

Credits to

https://cvpr2023-tutorial-diffusion-models.github.io

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

https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

https://developer.nvidia.com/blog/improving-diffusion-models-as-an-alternative-to-gans-part-1/