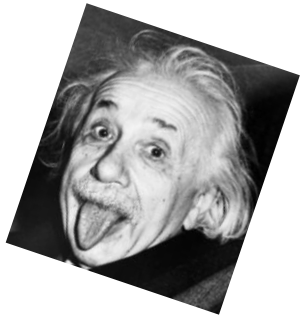
# (some) Generative Models (which are not necessarily GANs)

December 23rd, 2021

Niv Haim

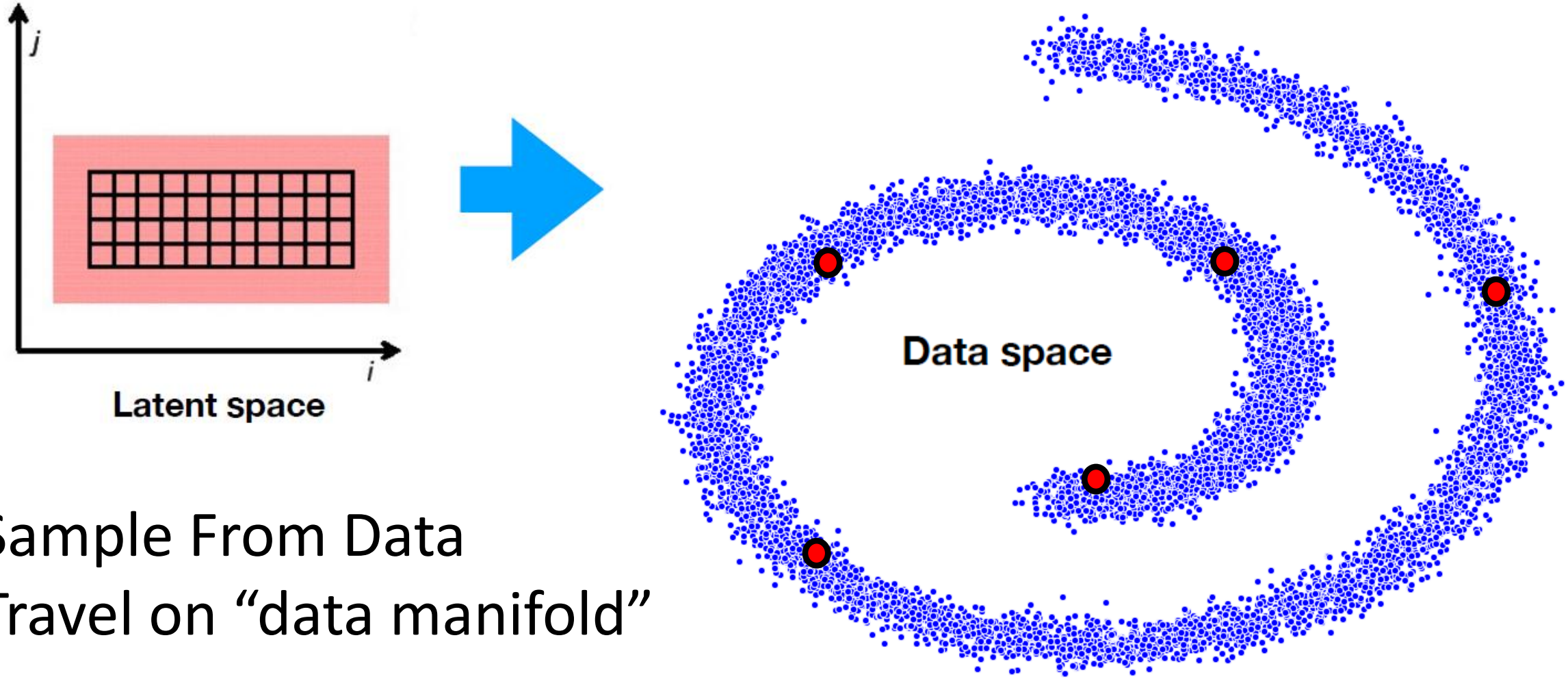
# Outline

- Autoencoders (AE)
- Variational Autoencoders (VAE)
- Vector Quantized VAE (VQ-VAE)
- Diffusion Models



# Objective - Reminder

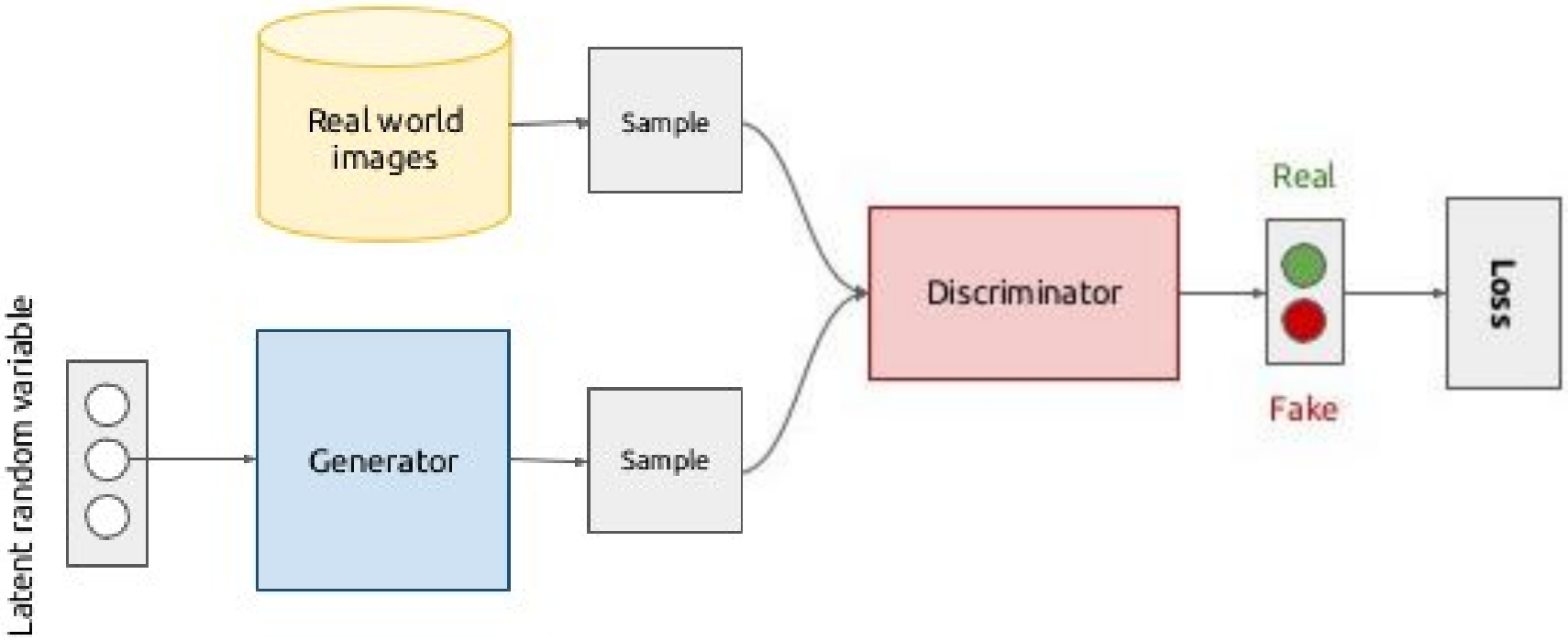
**Goal**: map simple (known) distribution to the data distribution




Sample From Data

Travel on “data manifold”

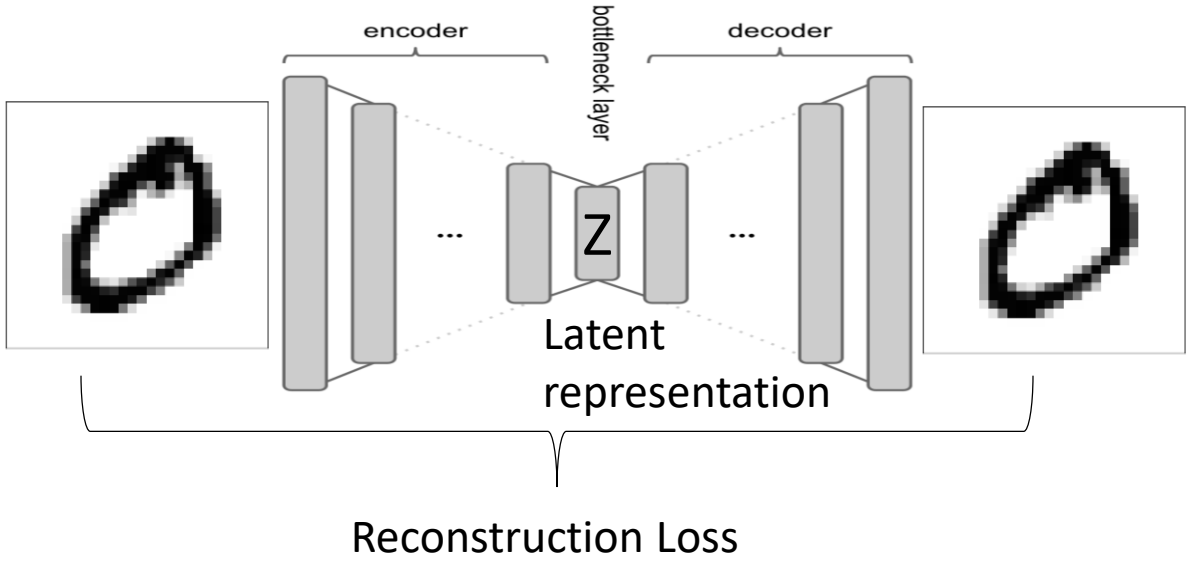
# GANs - Reminder




$$\mathcal{L}_{GAN} = \min_G \max_D \mathbb{E}_{x \sim p_{data}}[log\, D(x)] + \mathbb{E}_{z \sim p_z}[log(1 - D(G(z)))]$$

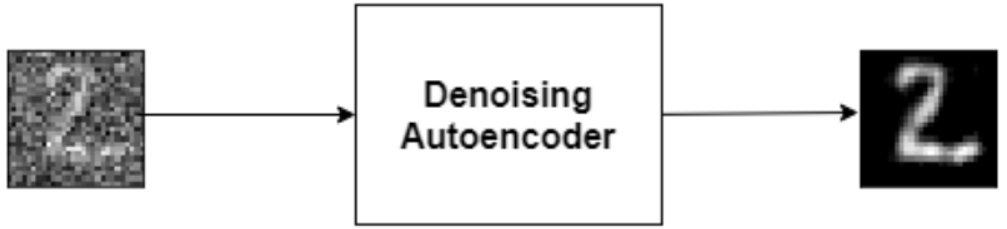
Source: Assaf Shocher's slides

# Autoencoders




Dimensionality Reduction

Denoising Autoencoder




source: https://theailearner.com/2018/11/10/denoising-autoencoders

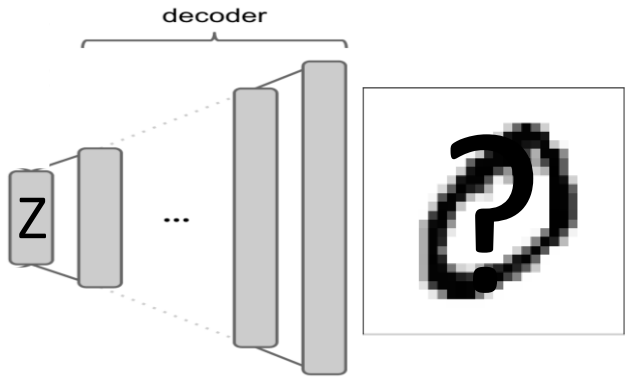
"Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion" [Vincent et al, 2010]

# - Generative model?

$Z_1 = E(img1)$

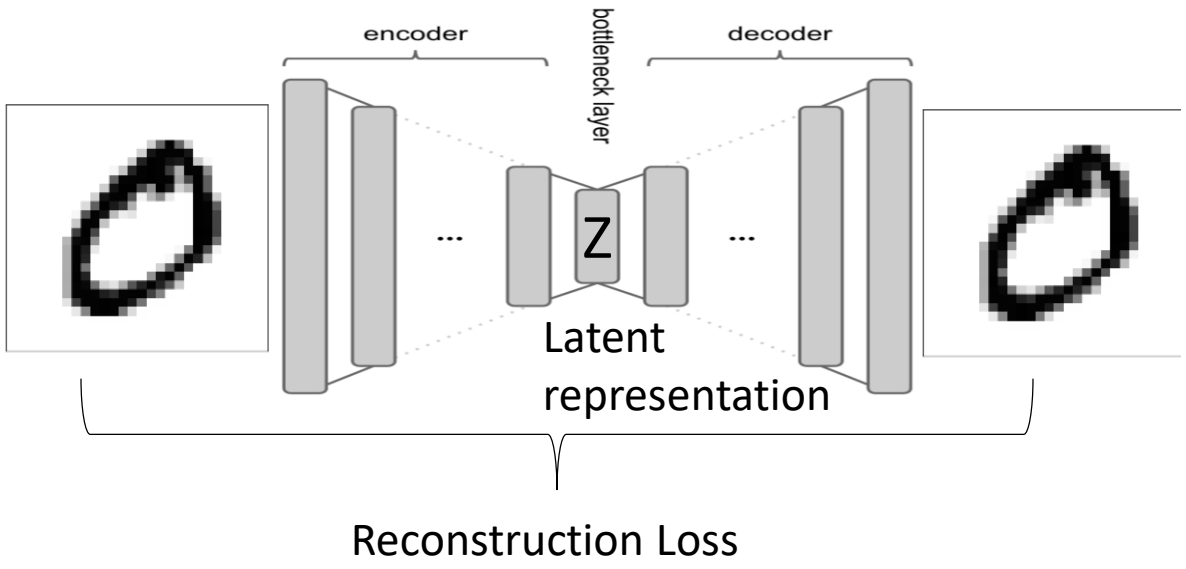
$Z_3 = Z_1 + ½ Z_2$

$Z_2 = E(img2)$




What happens in Latent Space?

# Autoencoders

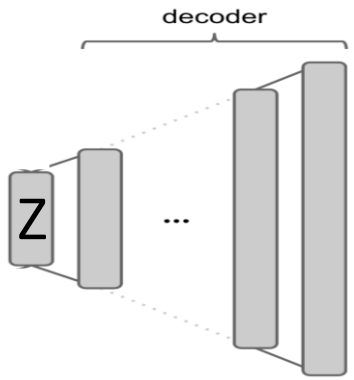



# - Generative model?

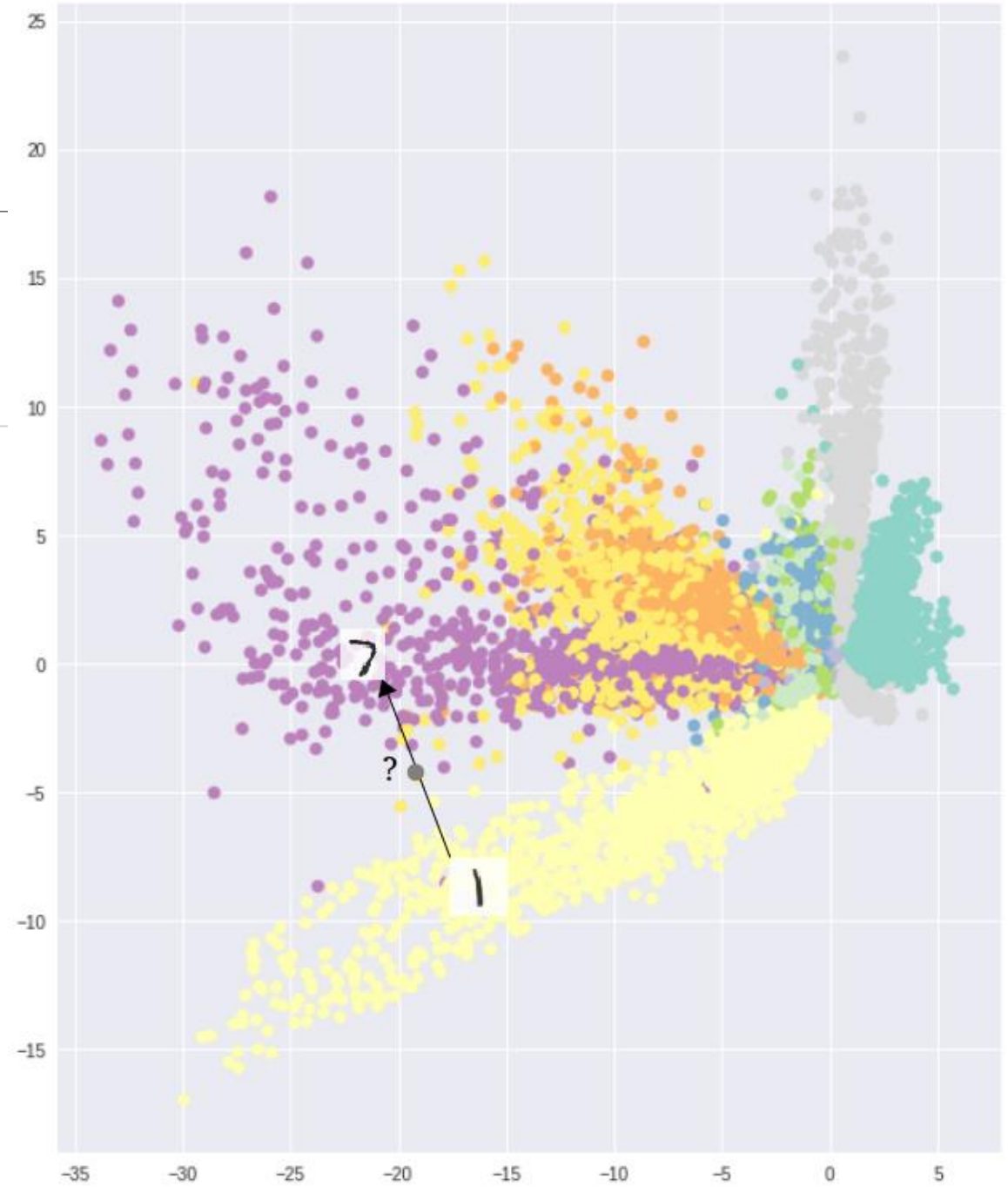
$Z_1 = E(img1)$

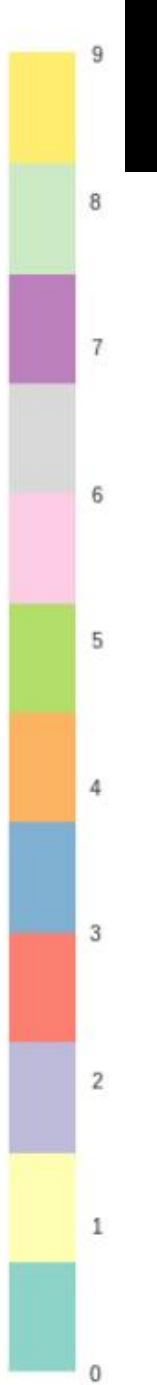
$Z_3 = Z_1 + ½ Z_2$

$Z_2 = E(img2)$

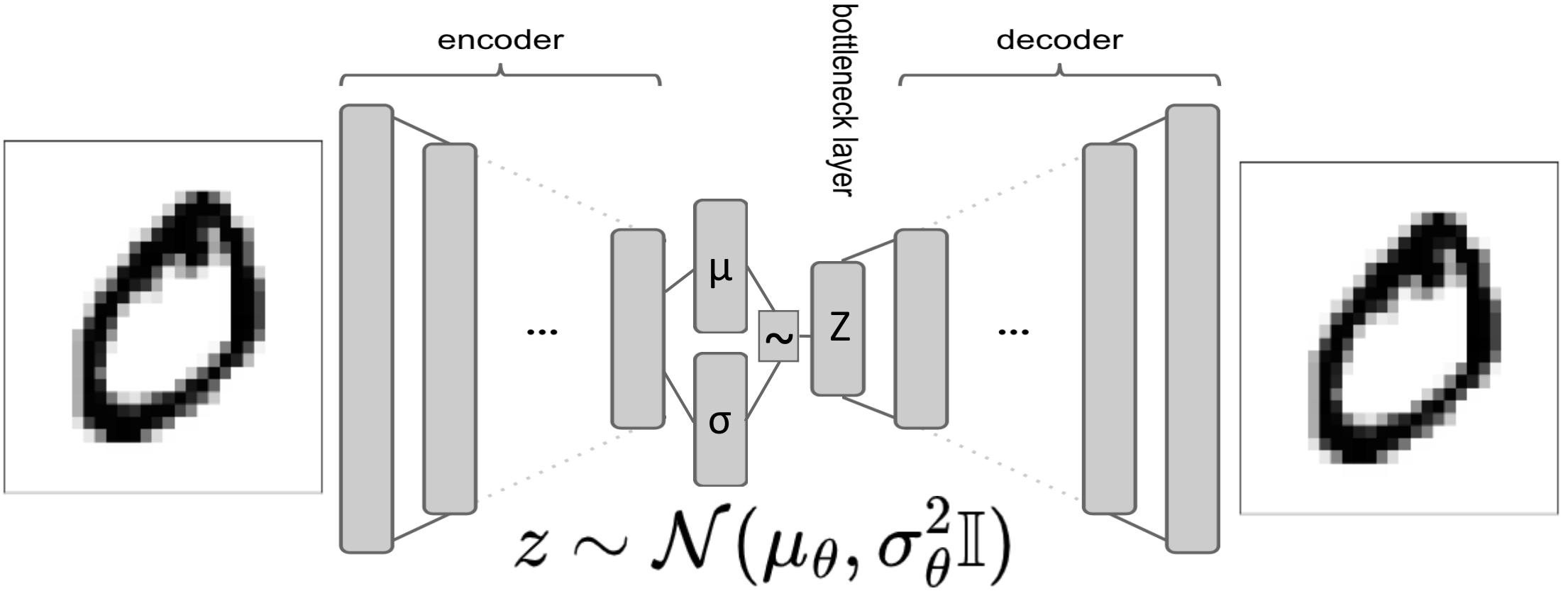



?



# Variational Autoencoders (VAE)




$$z \sim \mathcal{N}(\mu_\theta, \sigma_\theta^2 \mathbb{I})$$

"Reparametrization Trick":

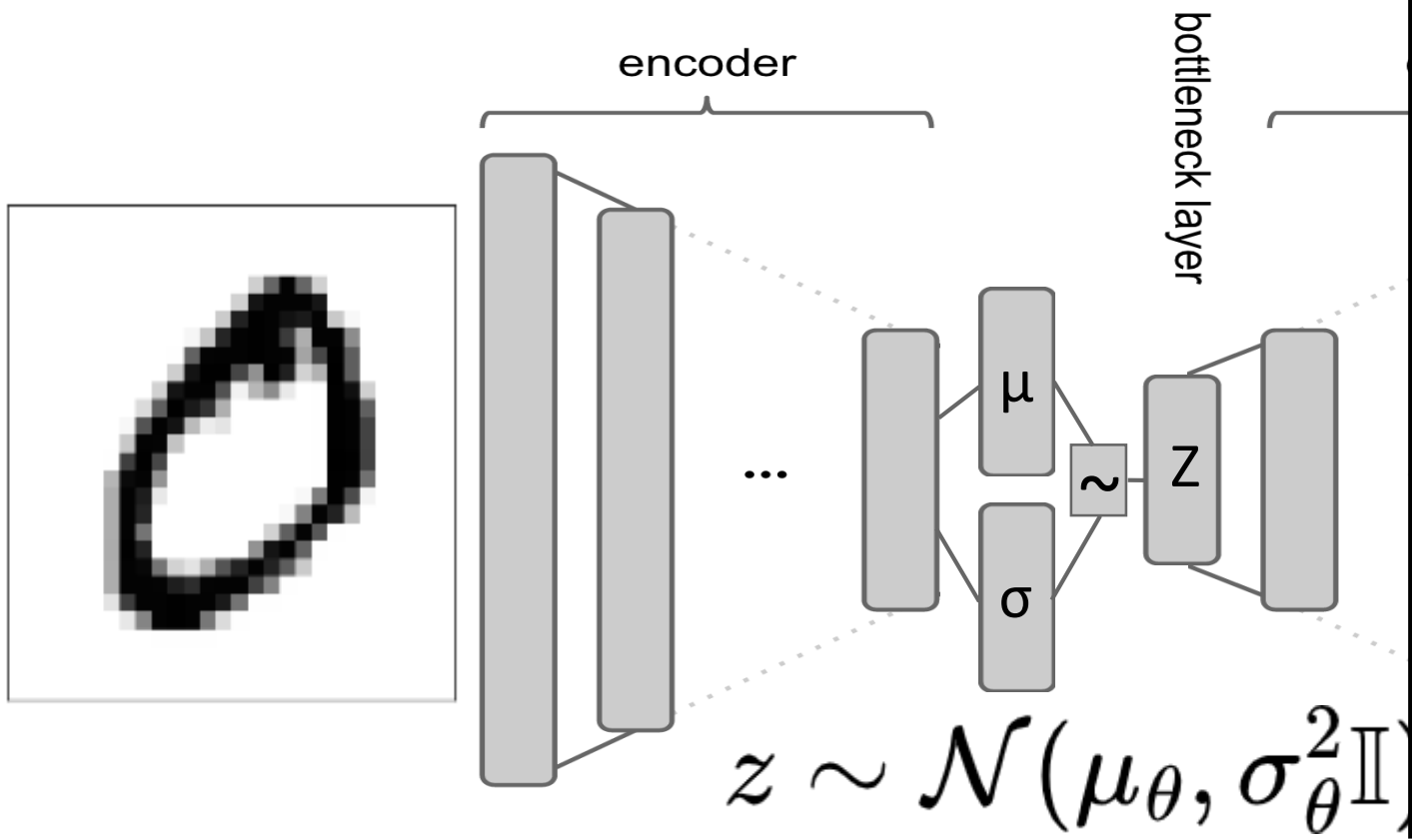
$$z_\theta = \mu_\theta + \sigma_\theta \cdot \mathcal{N}(0, \mathbb{I})$$

torch.randn_like(std)

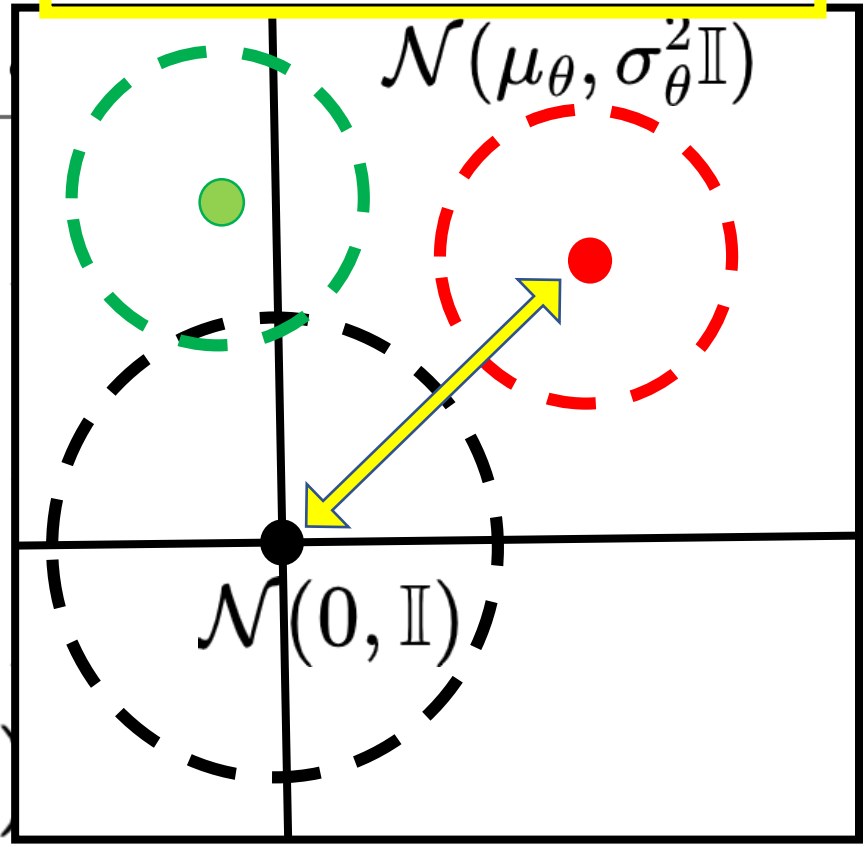


"Auto-Encoding Variational Bayes" [Kingma&Welling 14']

# Variational Autoencoders (VAE)




KL Between 2 Gaussians




Encourage $p(z) \sim \mathcal{N}(0,1)$:

$$\sum_{i=1}^{n} \sigma_i^2 + \mu_i^2 - \log(\sigma_i) - 1$$

"Reparametrization Trick":

$$z_\theta = \mu_\theta + \sigma_\theta \cdot \mathcal{N}(0, \mathbb{I})$$

torch.randn_like(std)

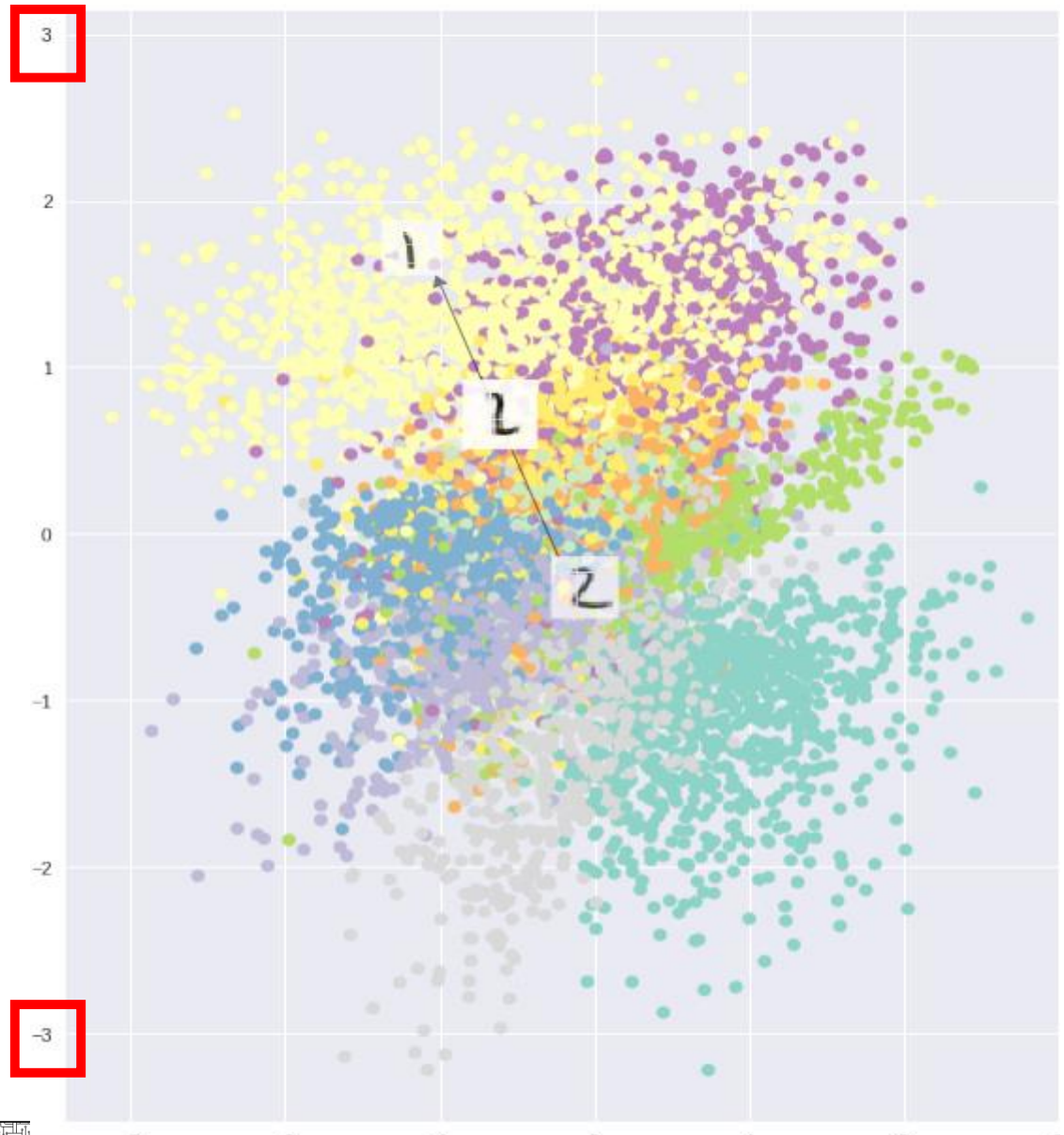


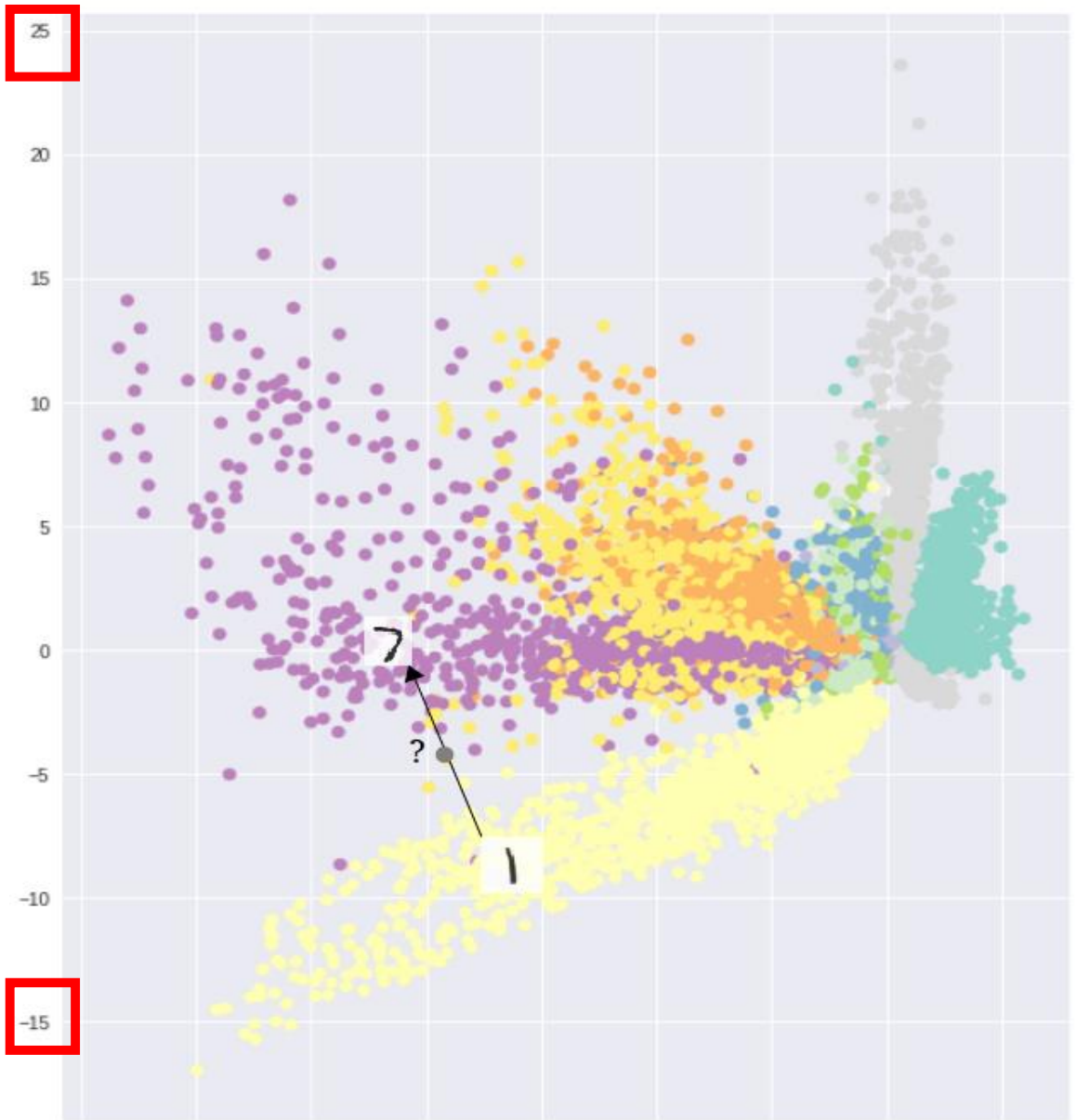
"Auto-Encoding Variational Bayes" [Kingma&Welling 14']

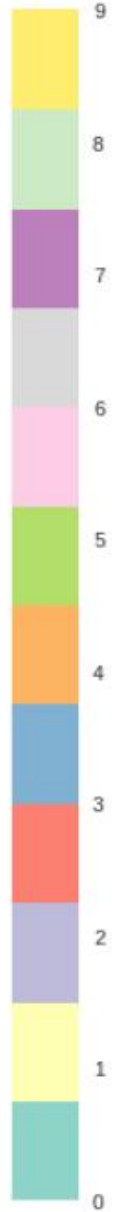
# VAE

# AE

**Also check out the scale!**

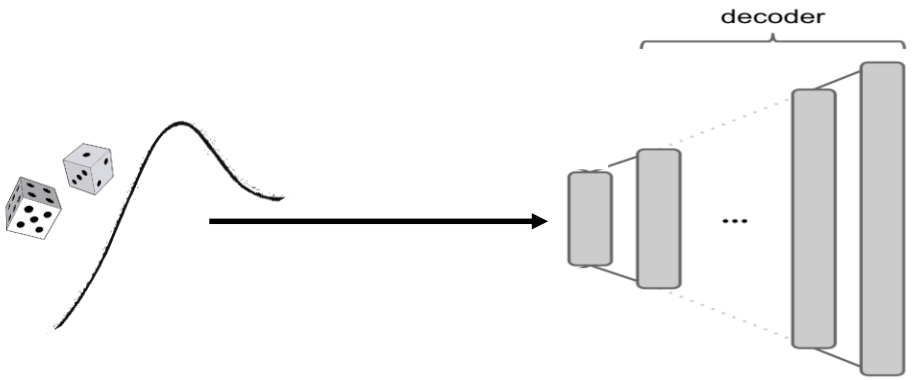


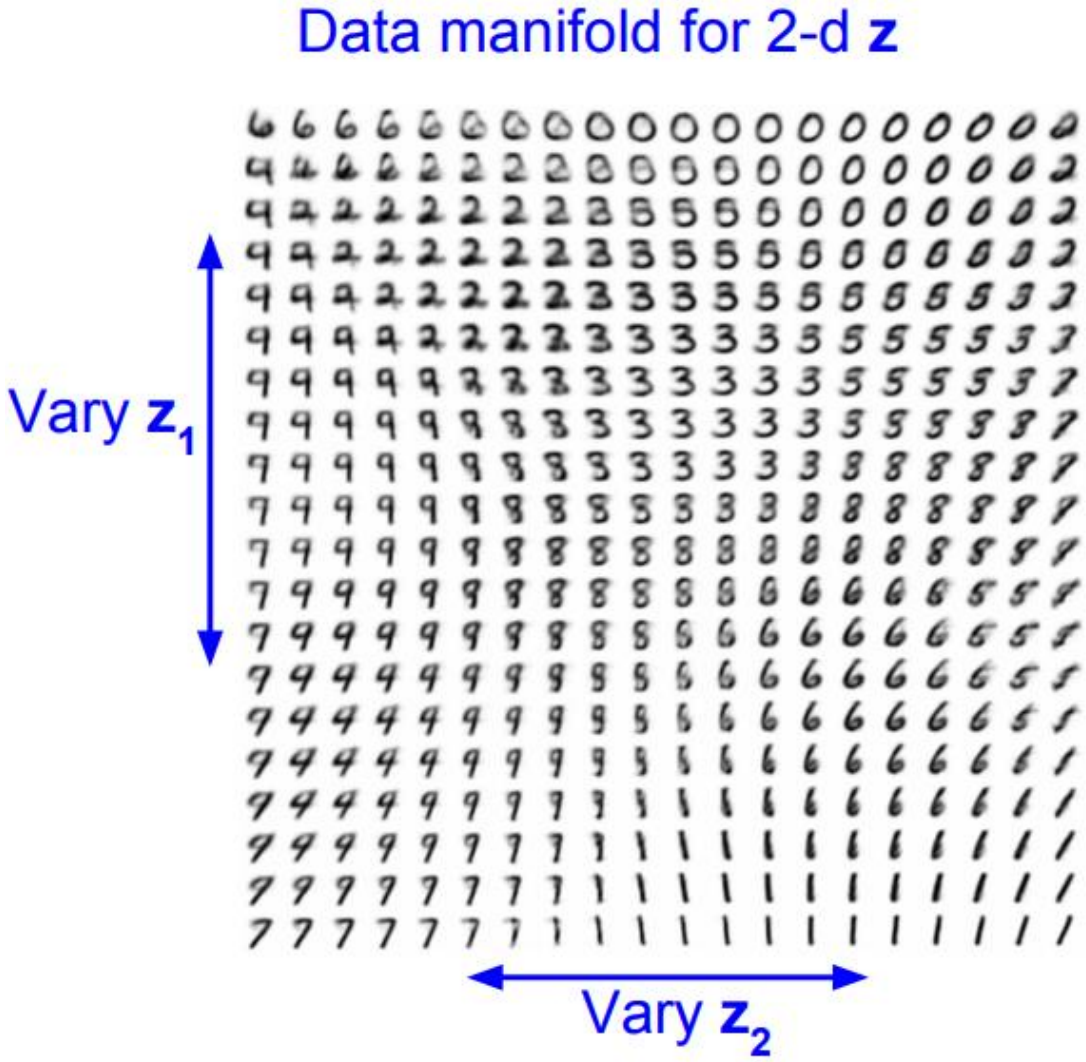




https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf

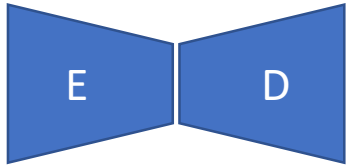
# Generate Data




From VAE paper

# Probabilistic Interpretation



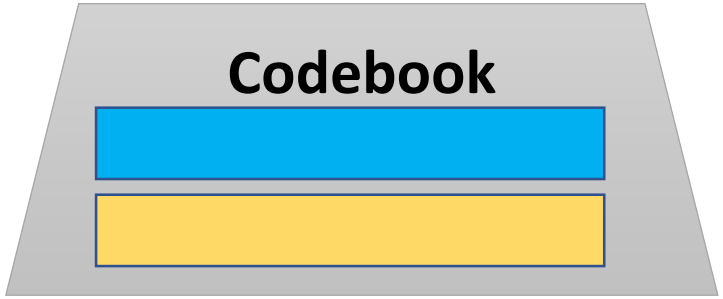

Reconstruction
+ Latent Prior

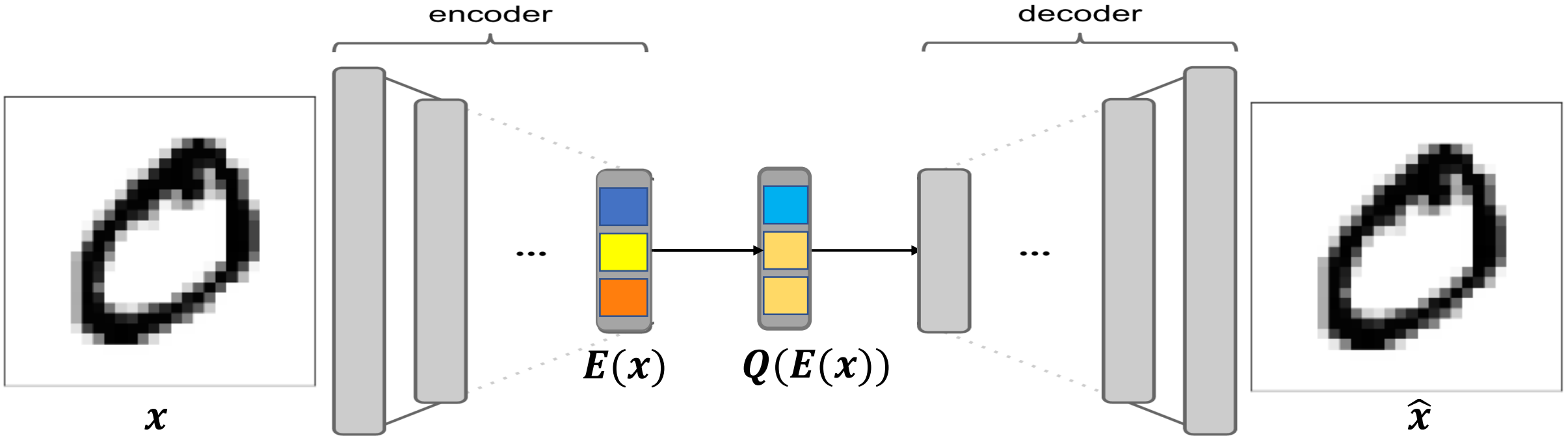
$$\log p_\theta(x^{(i)}) =$$

VAEs minimize a lower bound of the (minus) log likelihood

Slide credit: Stanford cs231n

# Vector-Quantized (VQ) VAE



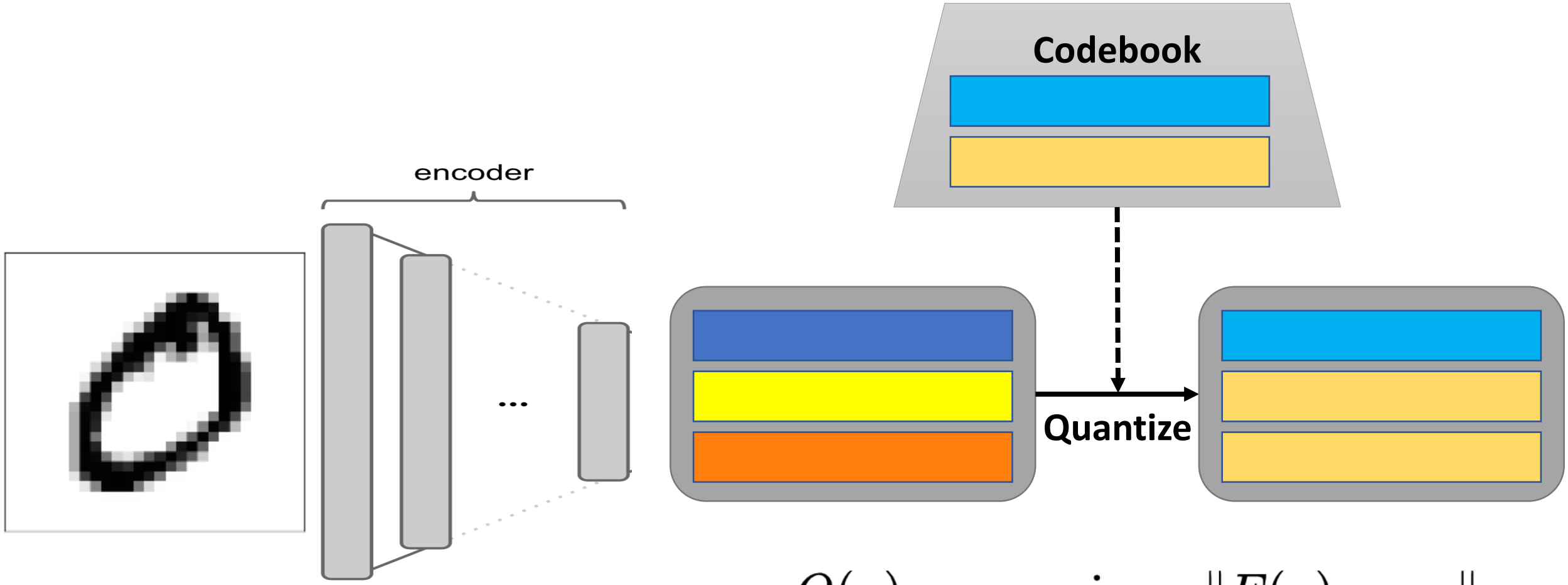




Quantization / Discretization

Source: Assaf Shocher's slides

"Neural Discrete Representation Learning " [van den Oord et al., 2017]

# Vector-Quantized (VQ) VAE

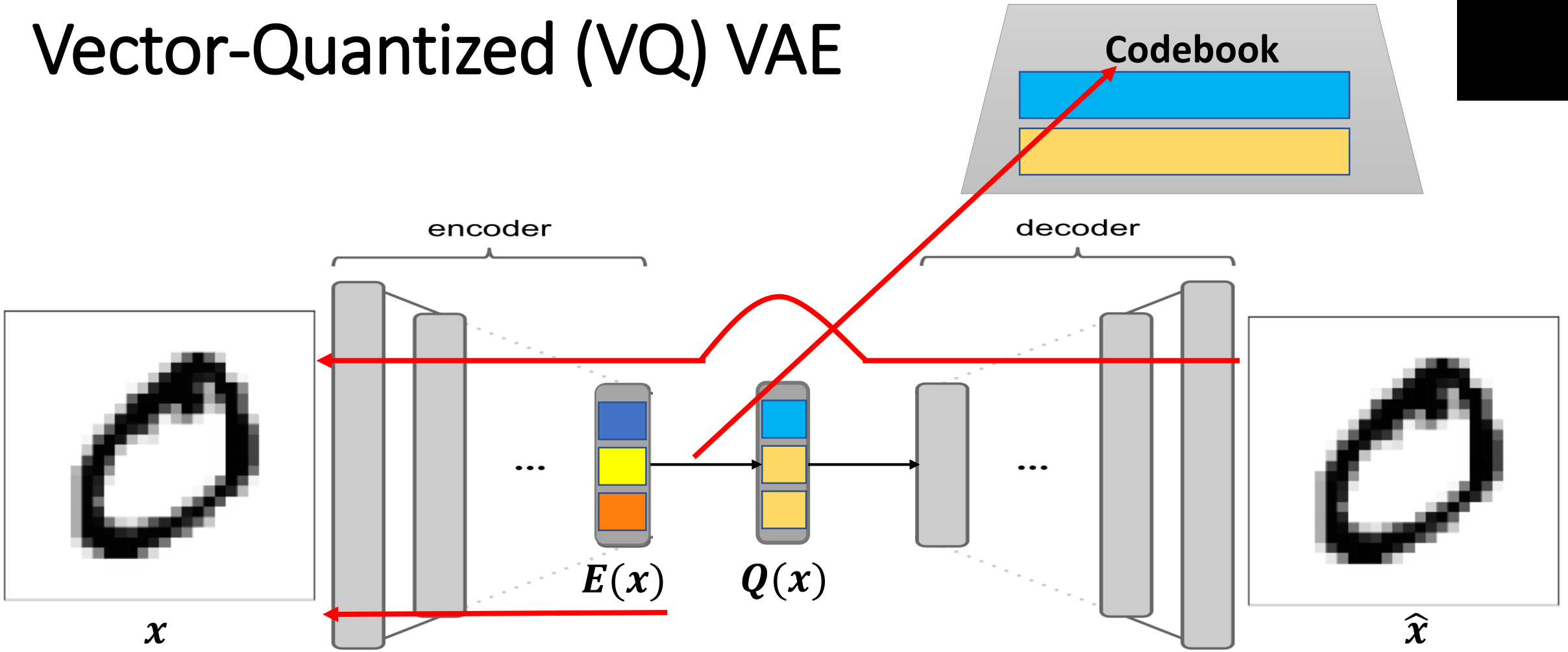



$$Q(x)_i = \min_{z_j \in \text{Codebook}} \|E(x)_i - z_j\|$$

Source: Assaf Shocher's slides

"Neural Discrete Representation Learning " [van den Oord et al., 2017]

# Vector-Quantized (VQ) VAE




$$\mathcal{L}_{rec} = \|\hat{x} - x\|_2^2 \qquad \mathcal{L}_{commit} = \|E(x) - sg(Q(x))\|_2^2 \qquad \mathcal{L}_{codebook} = \|sg(E(x)) - Q(x)\|_2^2$$

Quantization is non-differentiable!

$sg$ = stop-gradient

Source: Assaf Shocher's slides

"Neural Discrete Representation Learning " [van den Oord et al., 2017]

# VQ-VAE Reconstructions

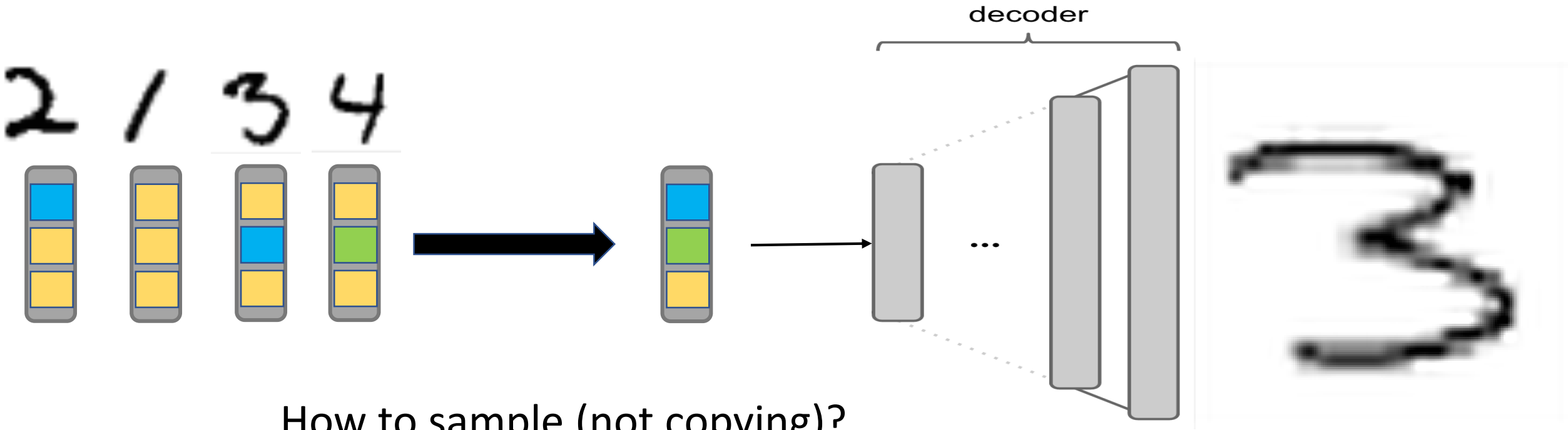
Real

Reconstructed

"Neural Discrete Representation Learning " [van den Oord et al., 2017]

# Sampling New Instances

How to sample new “sentence”?



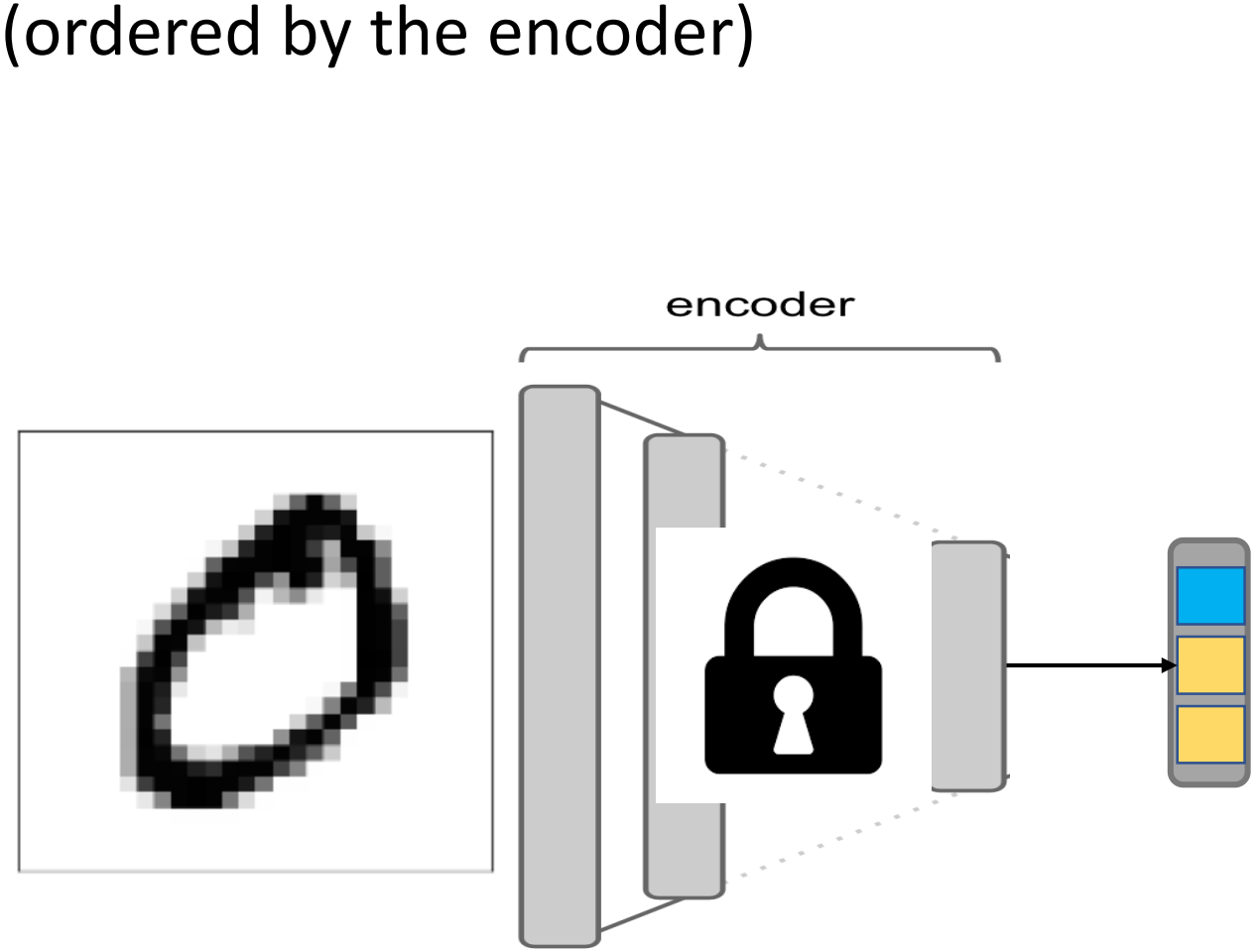

How to sample (not copying)?

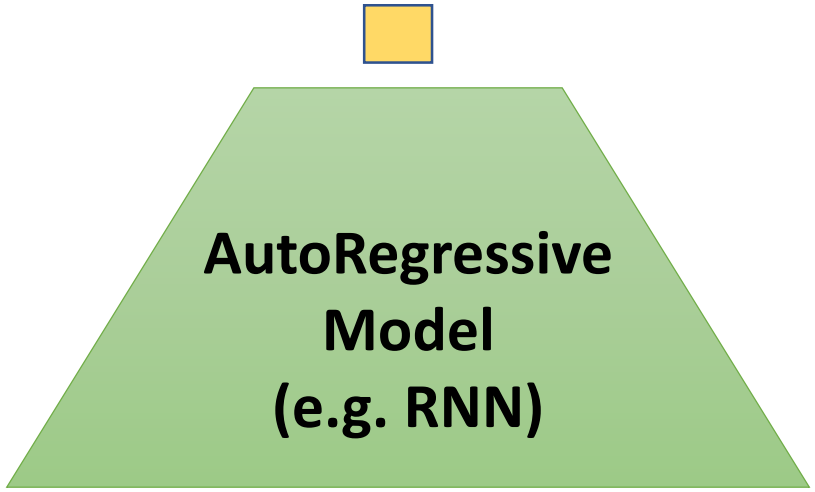
Train an autoregressive model to predict “words”

“Neural Discrete Representation Learning “ [van den Oord et al., 2017]

# Sampling New Instances

Training an autoregressive
on codebook elements
(ordered by the encoder)

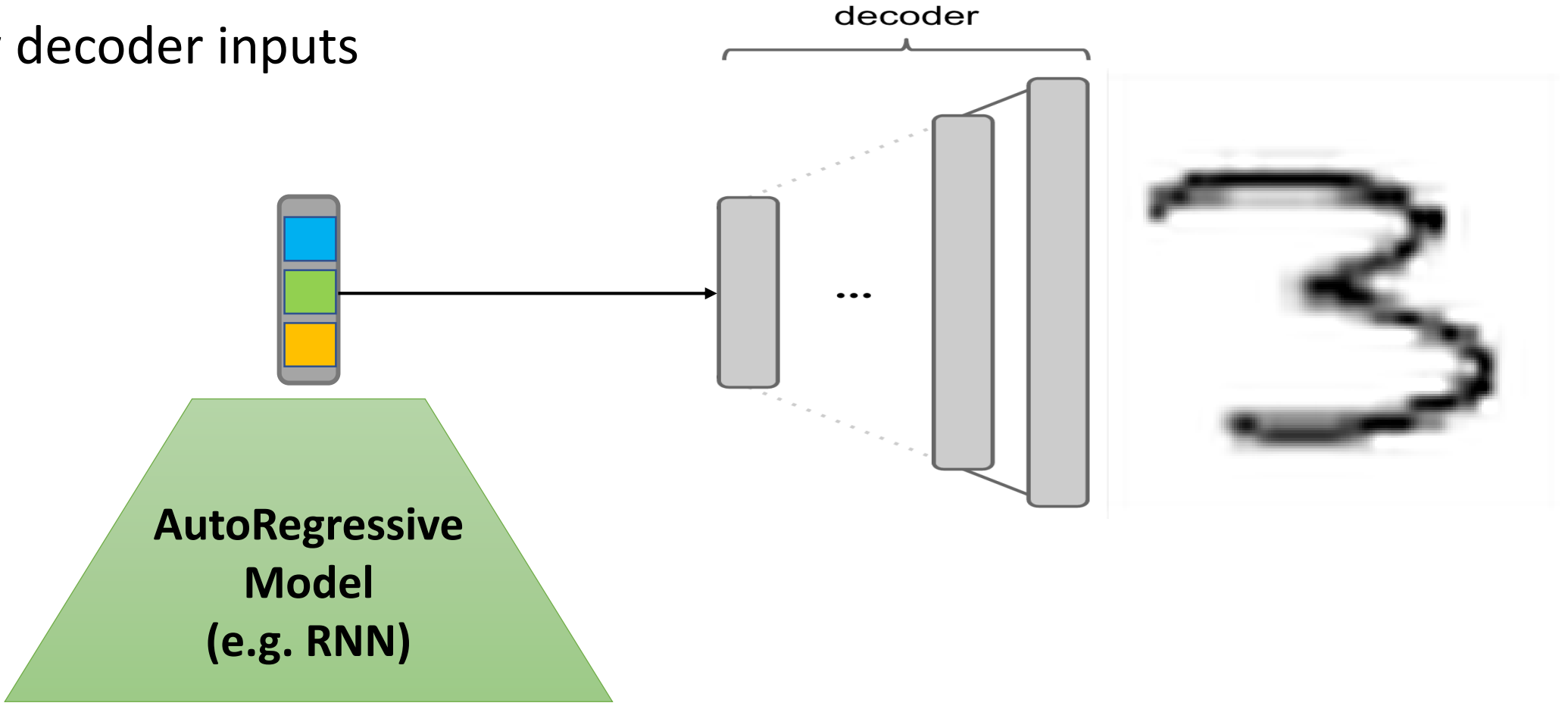



"Neural Discrete Representation Learning " [van den Oord et al., 2017]

# Sampling New Instances

Once trained, use it to
generate new decoder inputs

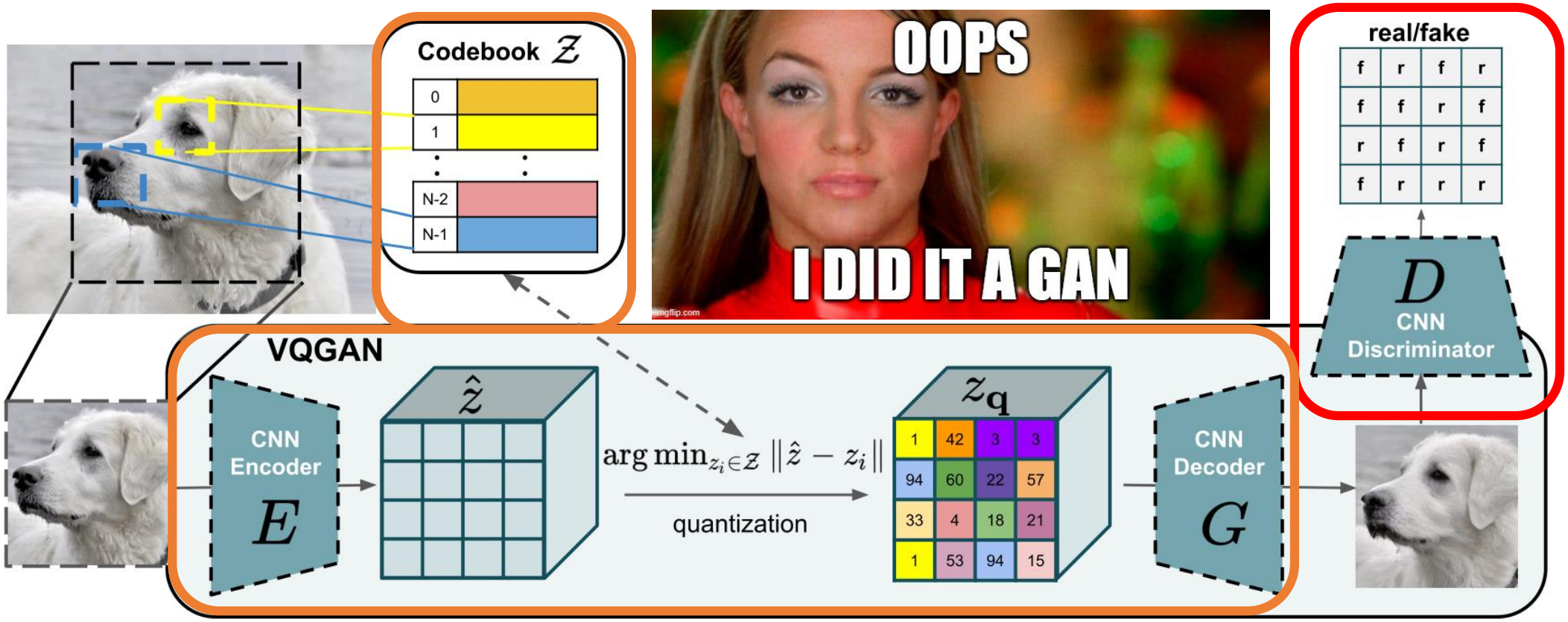



"Neural Discrete Representation Learning " [van den Oord et al., 2017]

# Sampling New Instances - Results

"Neural Discrete Representation Learning " [van den Oord et al., 2017]

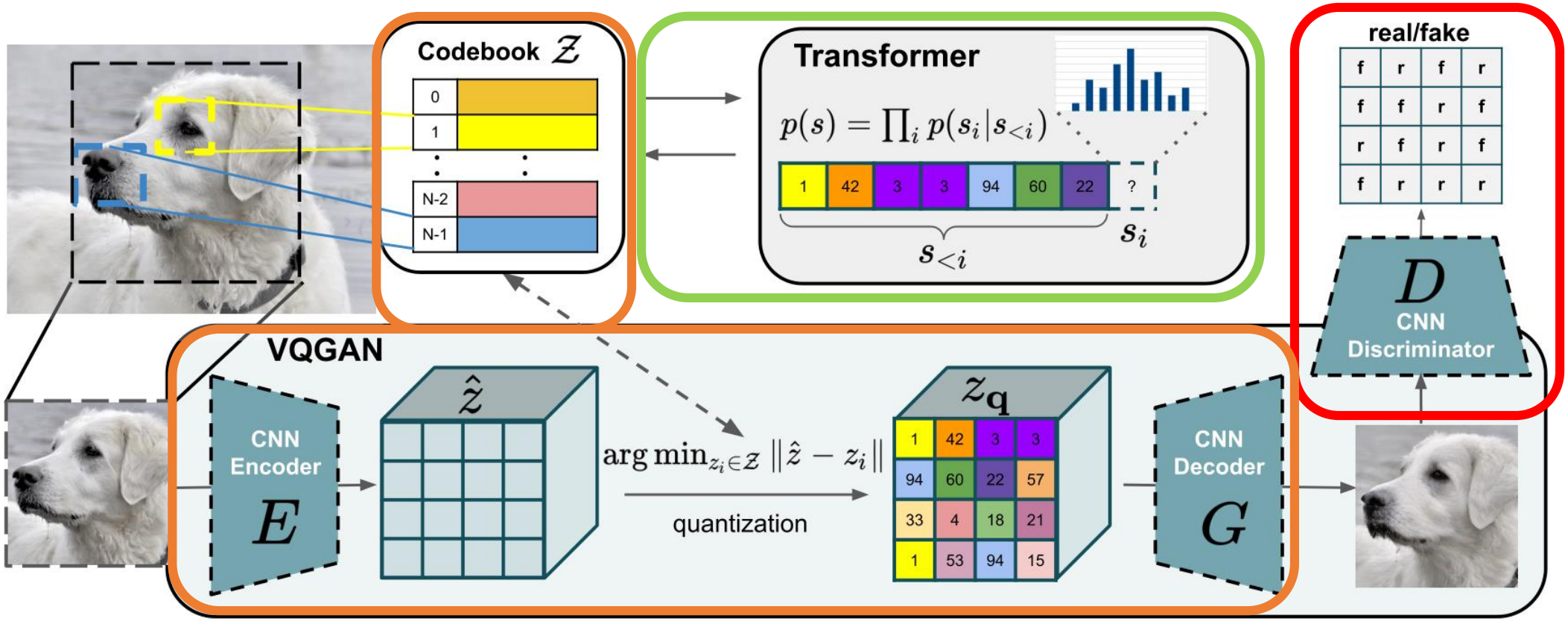
# VQGAN (Taming Transformers)




VQ-VAE + GAN

Taming Transformers for High-Resolution Image Synthesis [Esser et al. 2020]

# VQGAN (Taming Transformers)

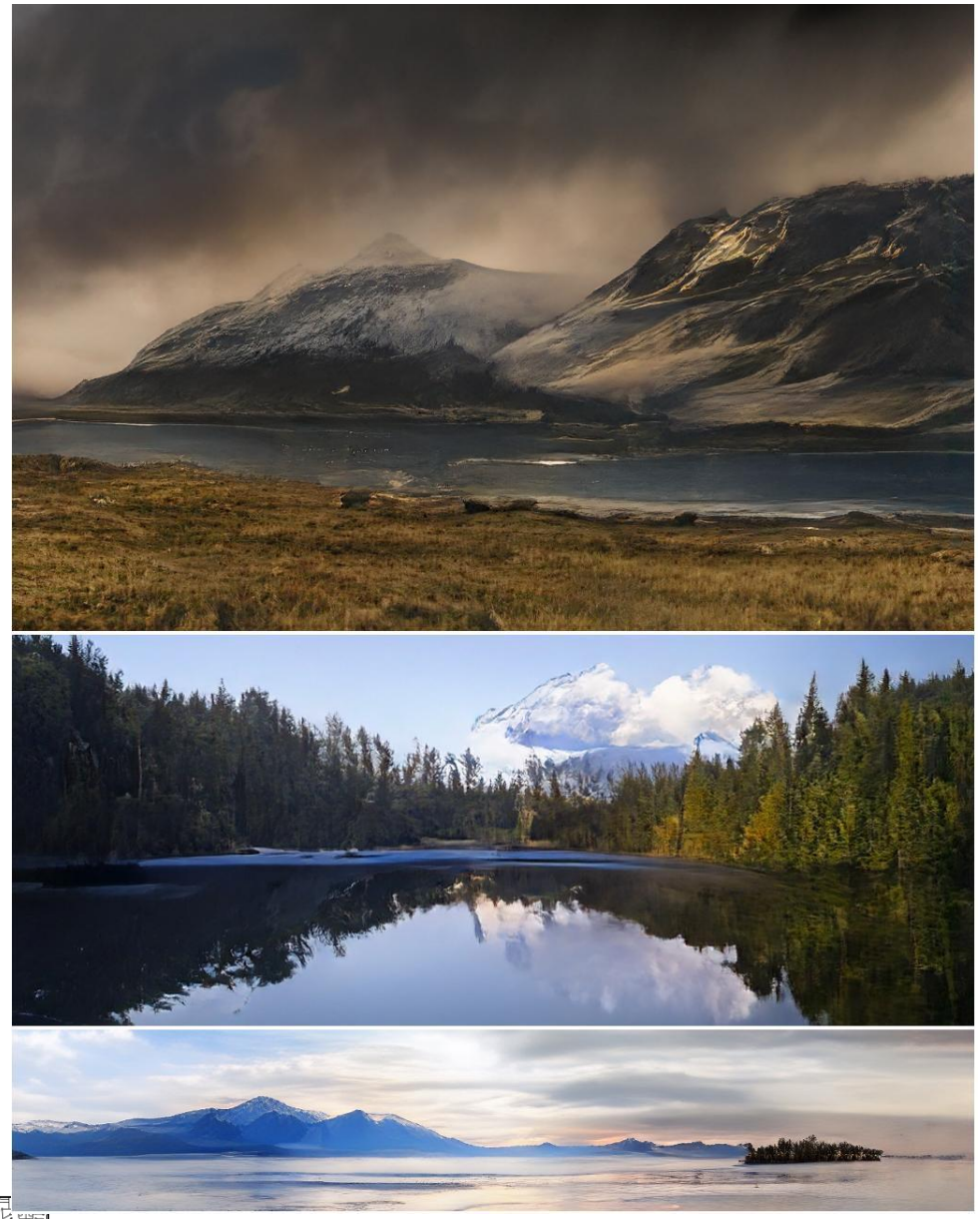


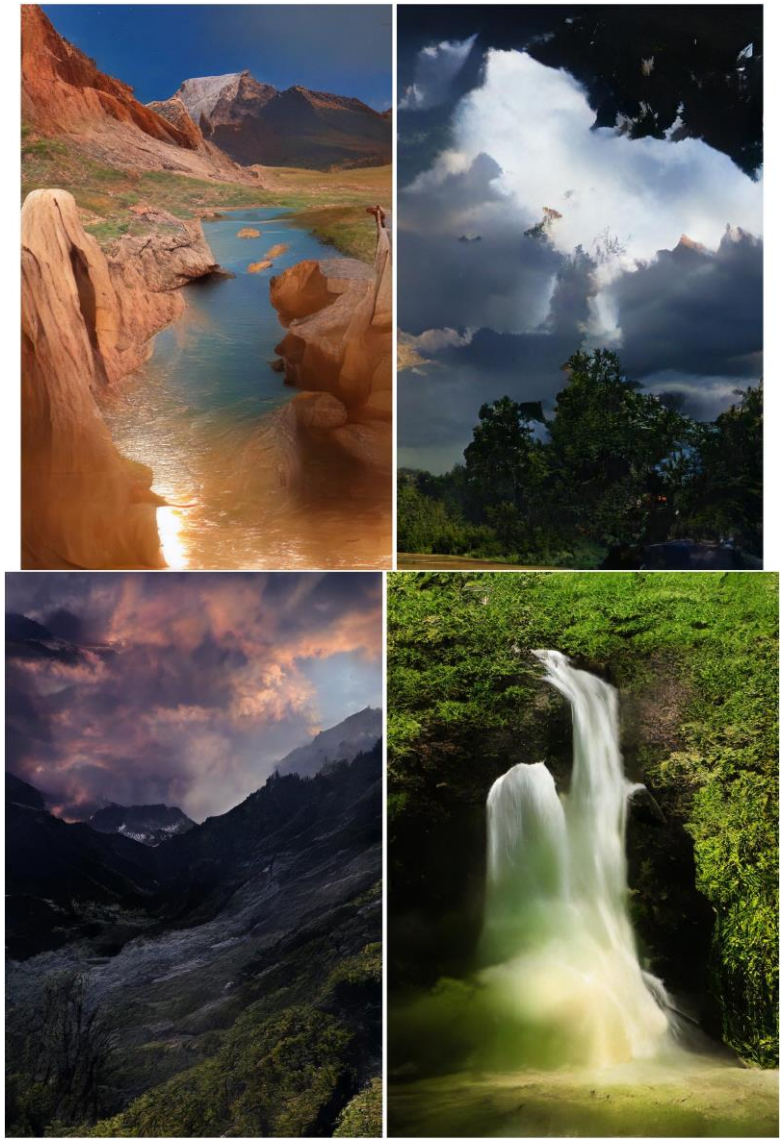

VQ-VAE + GAN

Transformer

Taming Transformers for High-Resolution Image Synthesis [Esser et al. 2020]

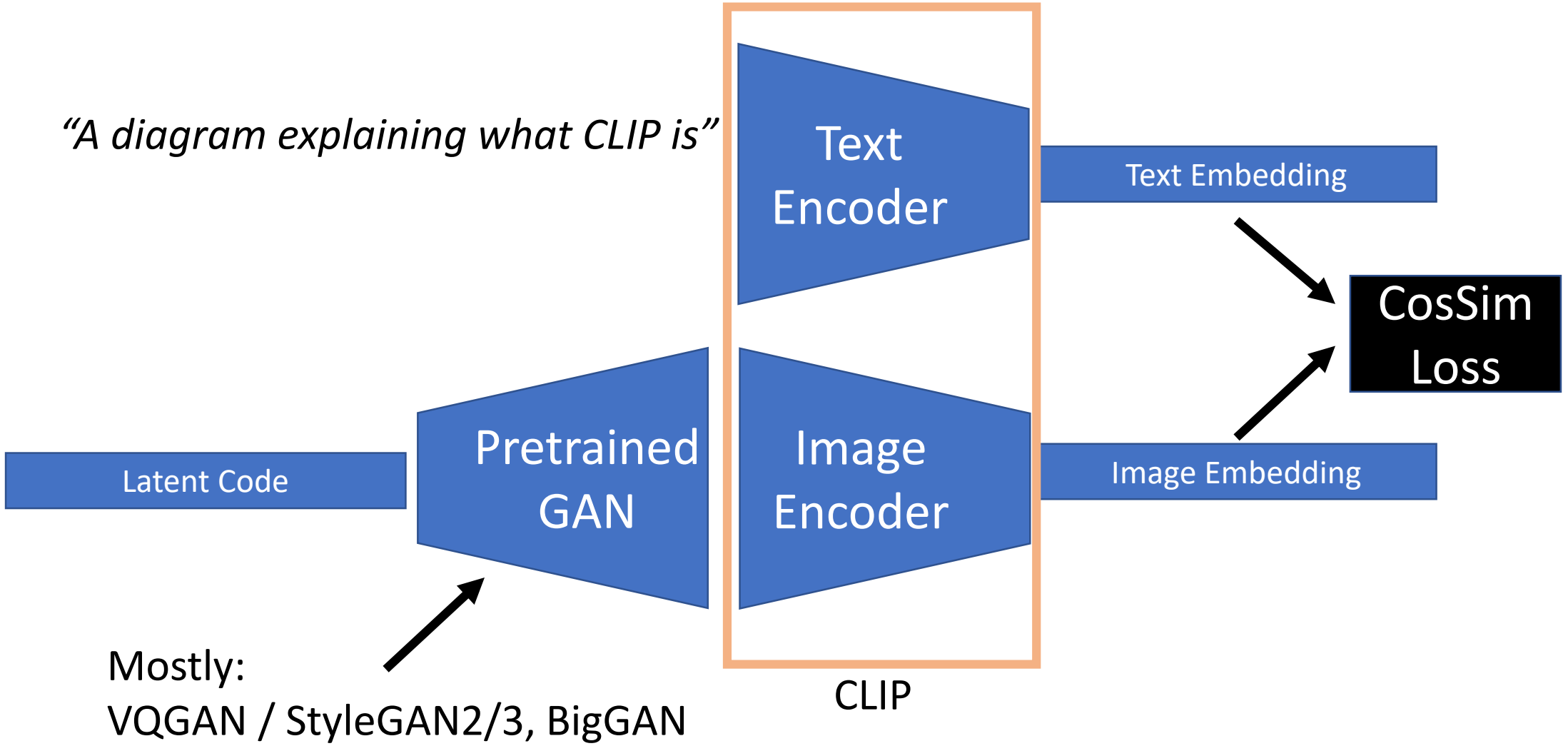
# VQGAN (Taming Transformers)





Taming Transformers for High-Resolution Image Synthesis [Esser et al. 2020]

# CLIP: Contrastive Langauage-Image Pre-Training




Learning Transferable Visual Models From Natural Language Supervision, Radford et al., 2021

# CLIP: Contrastive Langauage-Image Pre-Training



3985932

Learning Transferable Visual Models From Natural Language Supervision, Radford et al., 2021

# (~)VQ-VAE + Transformers: DALL-E

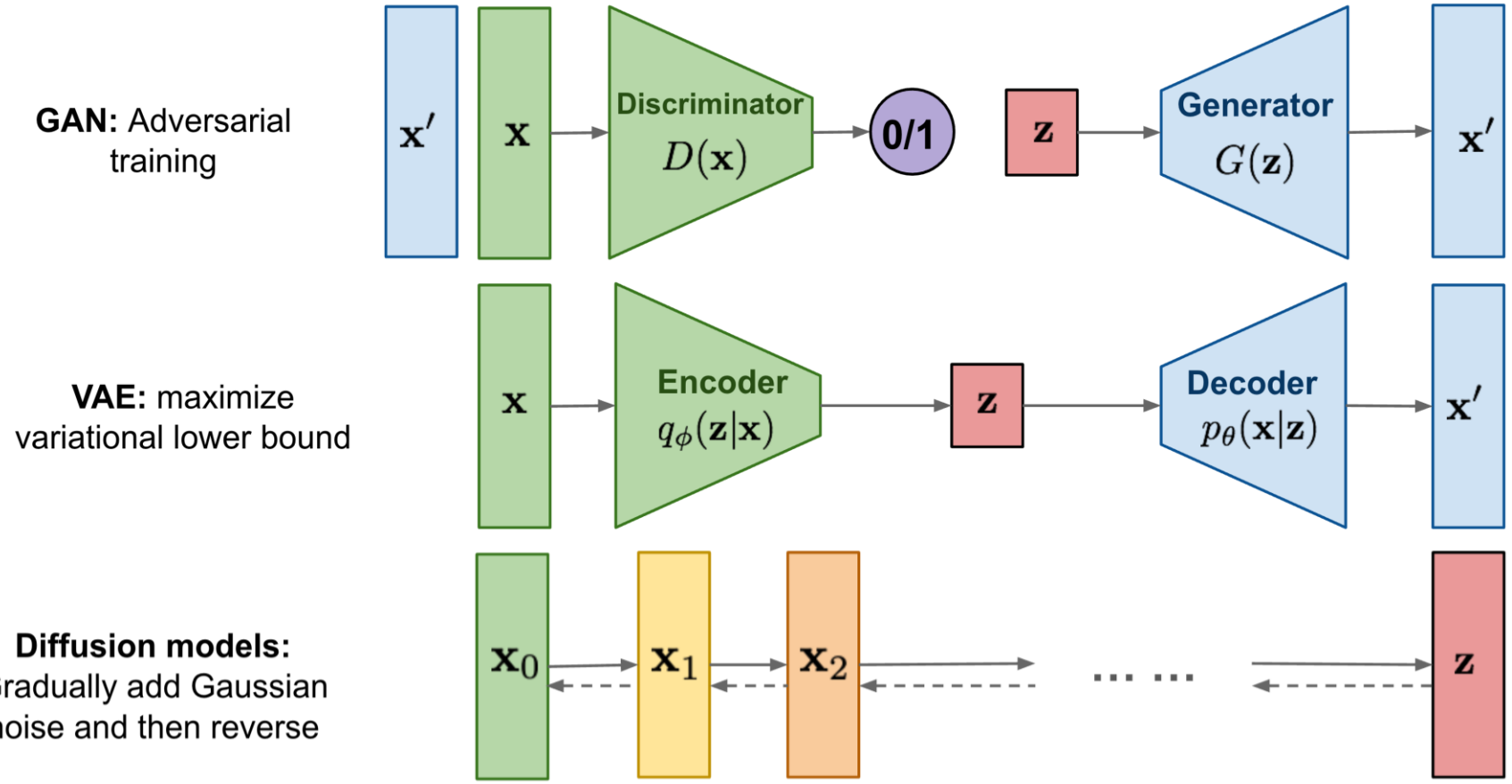
Ramesh et al., 2021

Teapot in the shape of a rubik's cube

Soap-dispenser in the shape of a doughnut

Store front with 'pytorch'

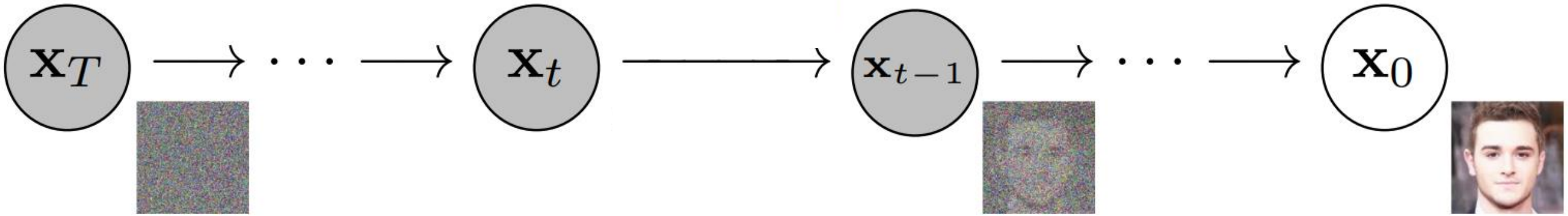
# Diffusion Models




https://lilianweng.github.io/lil-log/2021/07/11/diffusion-models.html

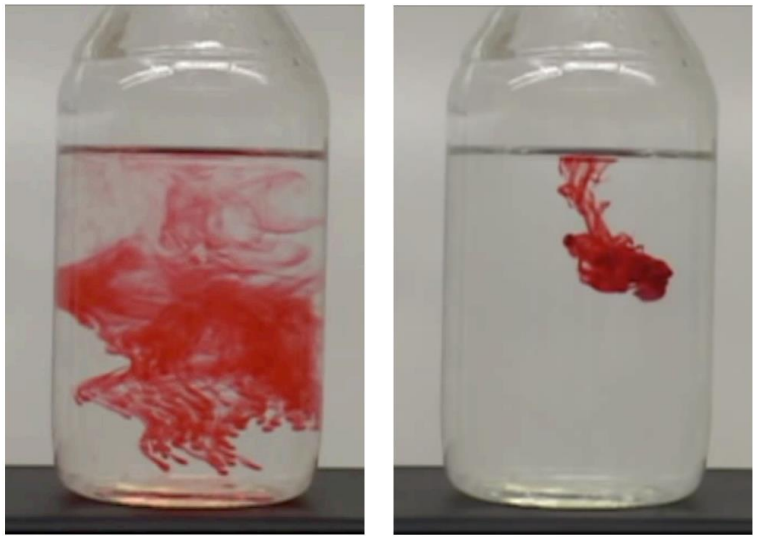
# Diffusion Models

1. How are we going to do that?!
2. Why should it even work?




Uniform/Normal Distribution

Data Distribution



Dye represents data density

<u>Observation:</u>

Diffusion destroys structure

<u>Core Idea:</u>

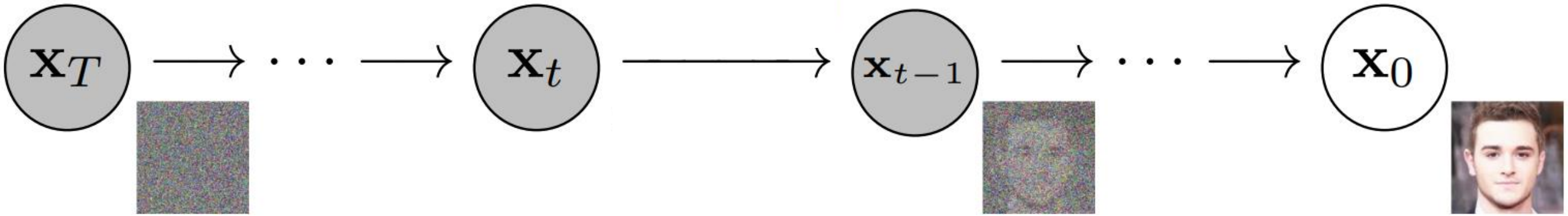
Recover structure by reversing time



Images from Jascha Sohl-Dickstein's slides and "Denoising Diffusion Probabilistic Models", [Ho et al. 2020]

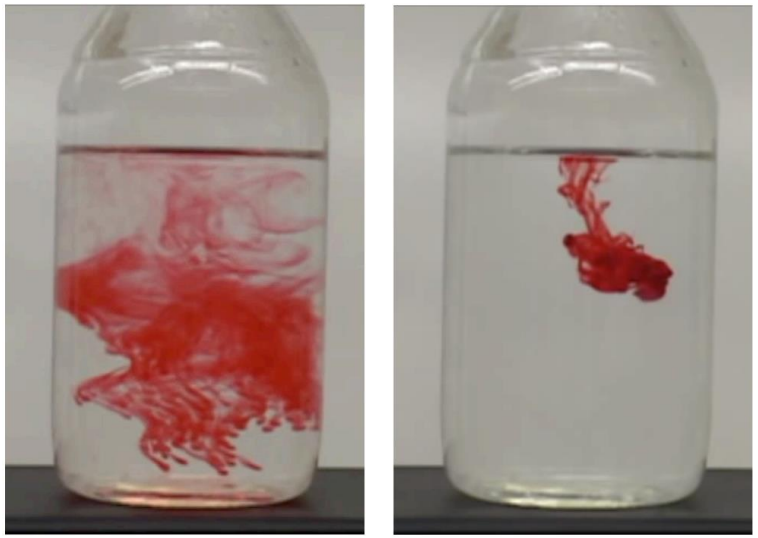
# Diffusion Models

1. How are we going to do that?!
2. Why should it even work?




Uniform/Normal Distribution

Data Distribution



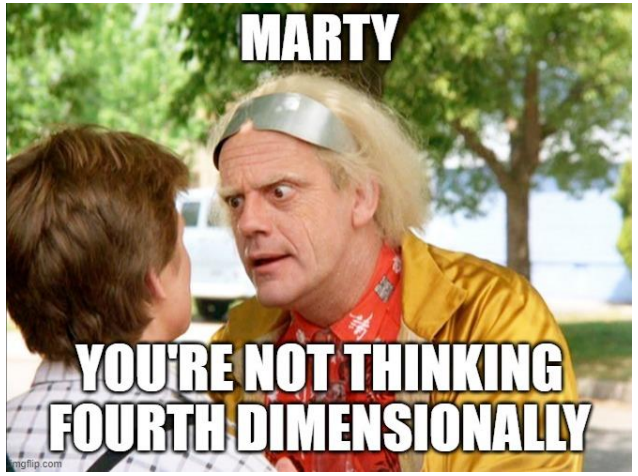
Dye represents data density

<u>Observation:</u>

Diffusion destroys structure

<u>Core Idea:</u>

Recover structure by reversing time

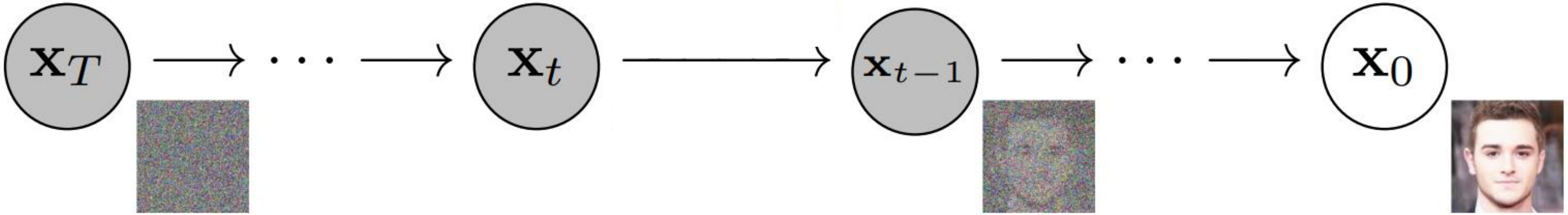


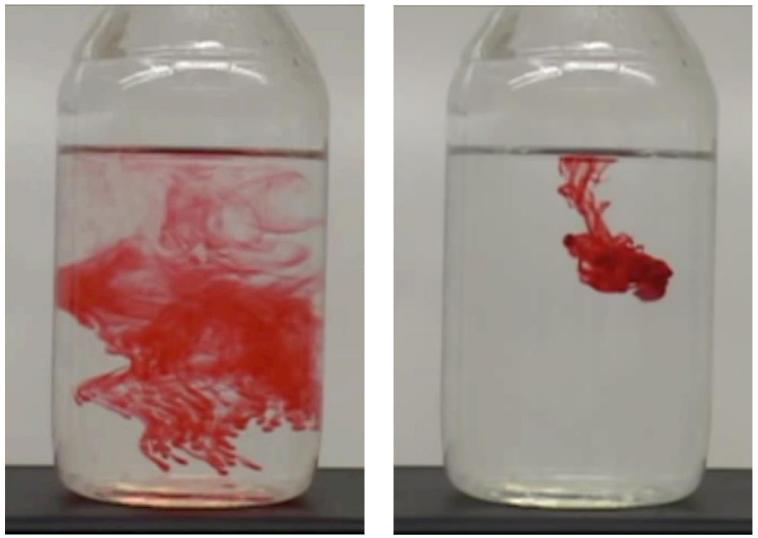


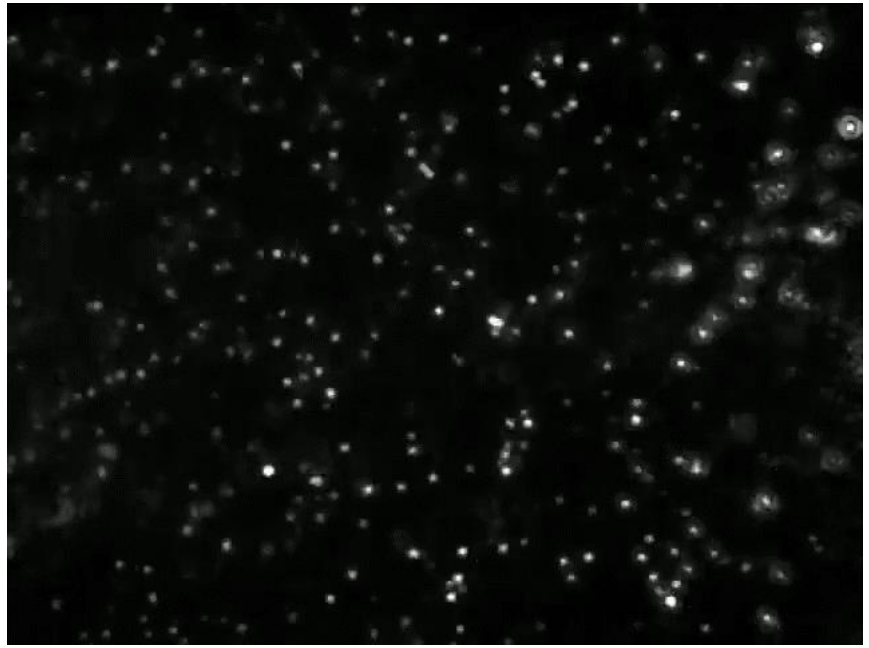

Images from Jascha Sohl-Dickstein's slides and "Denoising Diffusion Probabilistic Models", [Ho et al. 2020]

# Diffusion Models

1. How are we going to do that?!
2. Why should it even work?





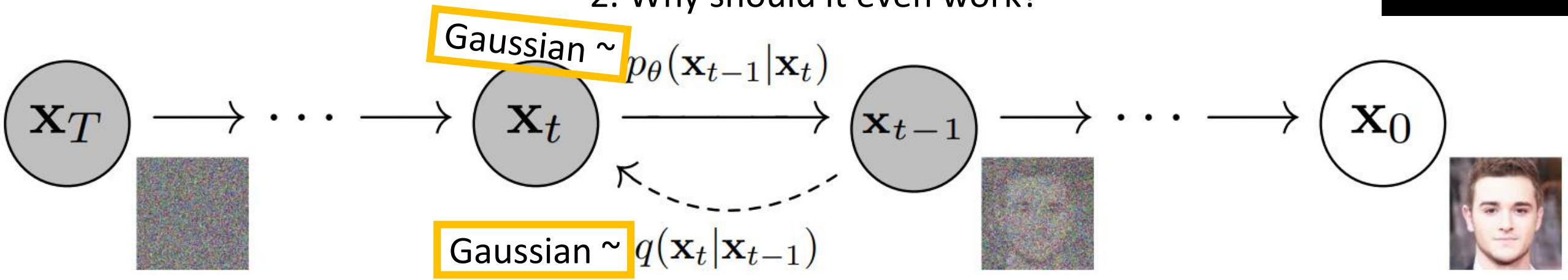
© Rutger Saly

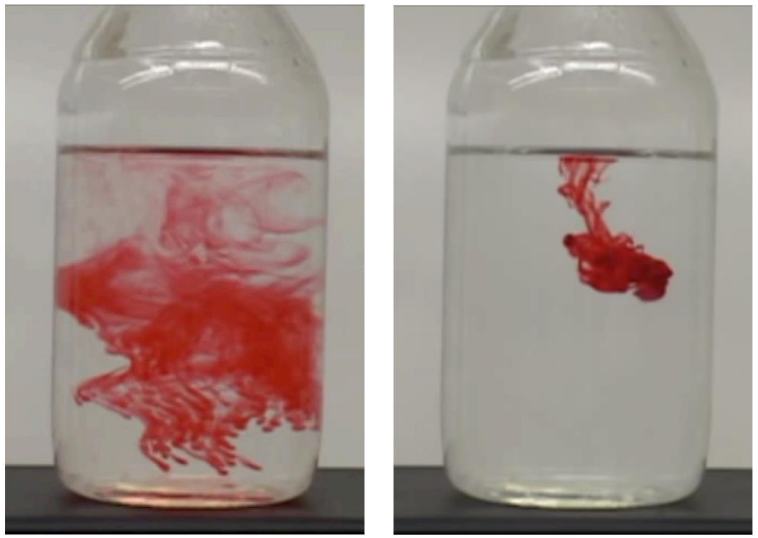


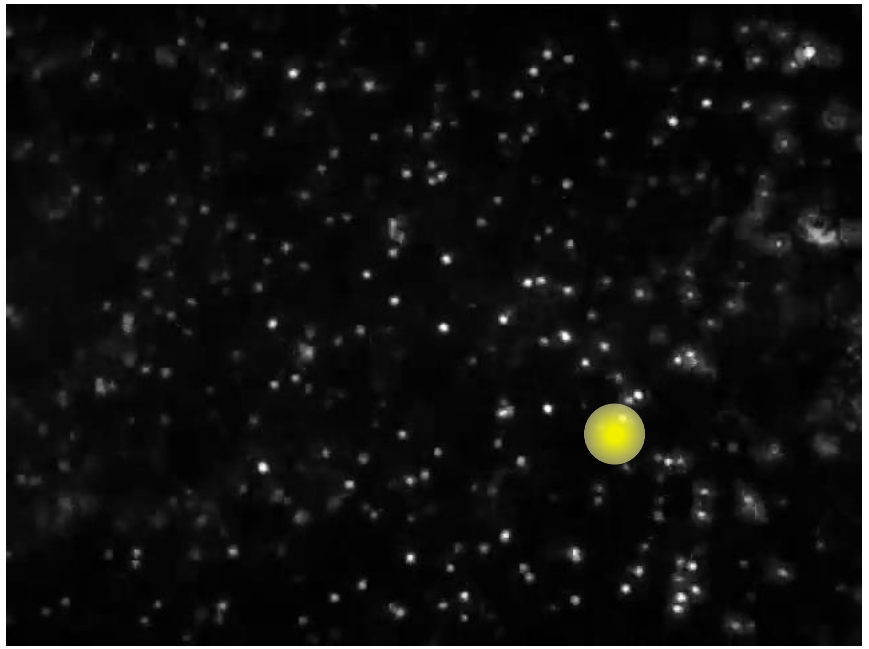
Images from Jascha Sohl-Dickstein's slides and "Denoising Diffusion Probabilistic Models", [Ho et al. 2020]

# Diffusion Models

1. How are we going to do that?!
2. Why should it even work?





© Rutger Saly

Position updates are small Gaussian

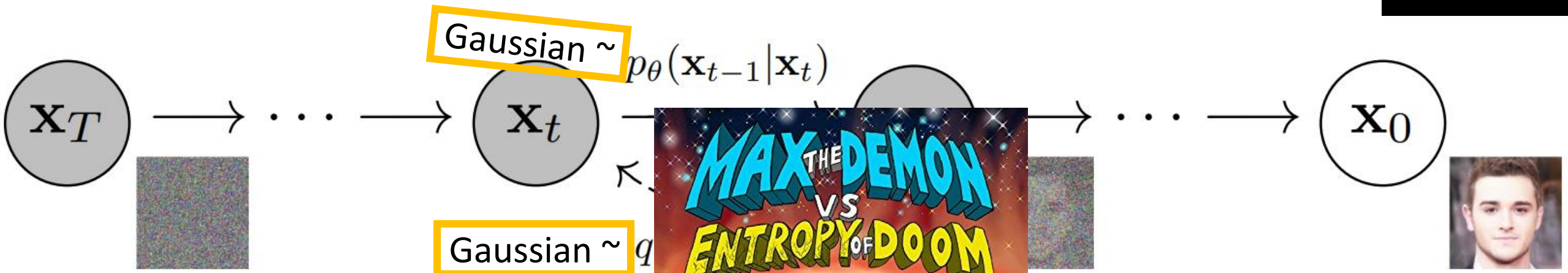
For Small step size:
Forward and Reverse
Have similar functional shape [Feller, 1949]

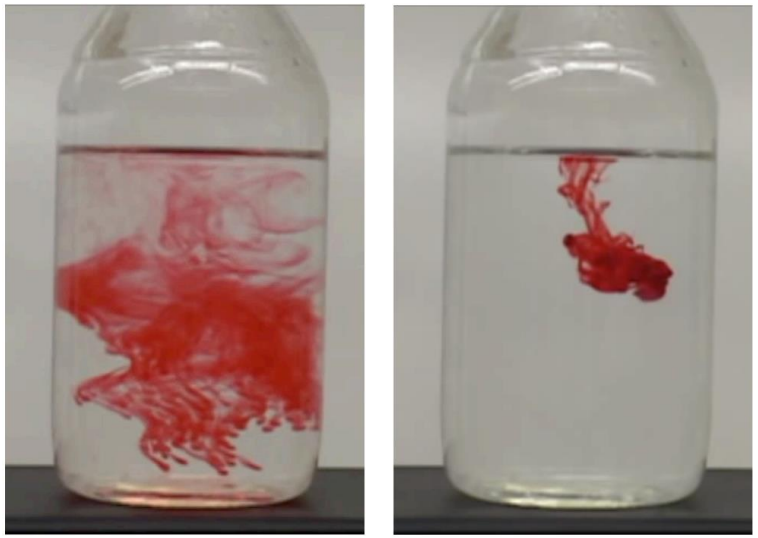


Images from Jascha Sohl-Dickstein's slides and "Denoising Diffusion Probabilistic Models", [Ho et al. 2020]

# Diffusion Models

1. How on earth are we going to do that?!
2. Why on earth should it even work?




© Rutger Saly

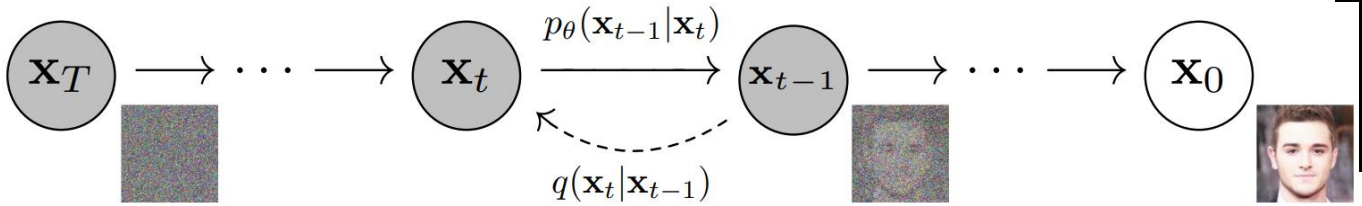
Position updates are small Gaussian

For Small step size:
Forward and Reverse
Have similar functional shape [Feller, 1949]



Images from Jascha Sohl-Dickstein's slides and "Denoising Diffusion Probabilistic Models", [Ho et al. 2020]

# Diffusion Models




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

add small noise

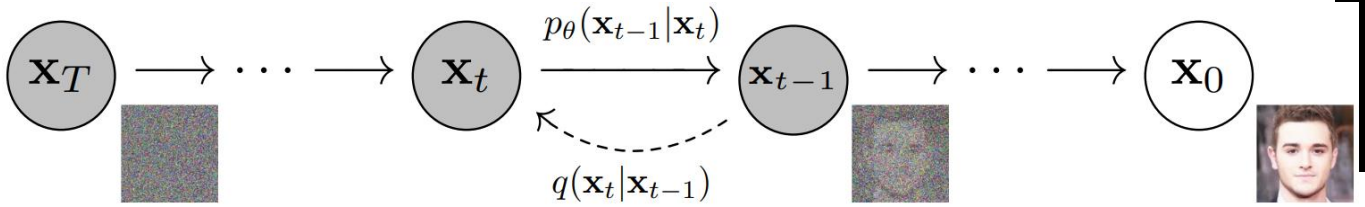
decay to origin

Forward process:

Gradually adding noise

According to beta schedule

# Diffusion Models

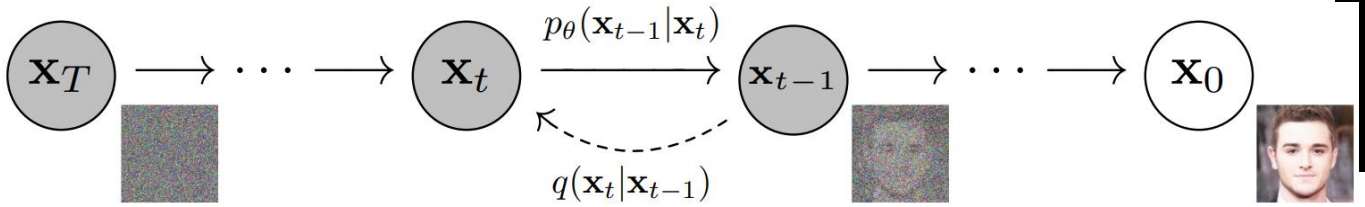



$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

$q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ Intractable ☹ (depends on entire dataset)

Yannic on DDPM: https://www.youtube.com/watch?v=W-O7AZNzbzQ&t=2015s

# Diffusion Models

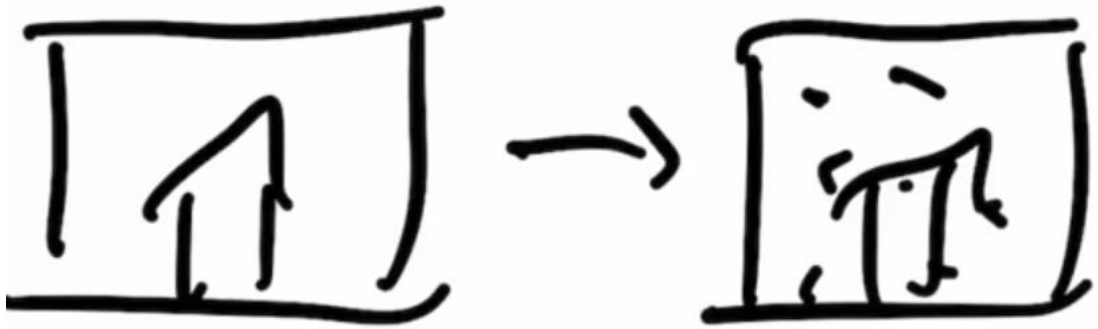



$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

$q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ Intractable ☹ (depends on entire dataset)

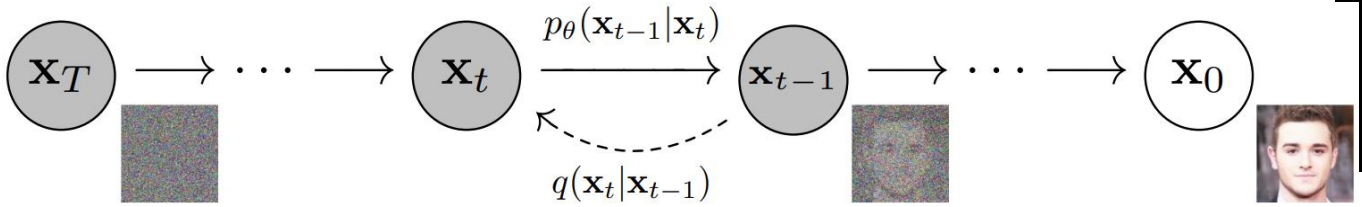
Reverse:
Needs a lot (!) of priors about the data/world

Forward:
Easy



Yannic on DDPM: https://www.youtube.com/watch?v=W-O7AZNzbzQ&t=2015s

# Diffusion Models




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

$q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ Intractable ☹ (depends on entire dataset)

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

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

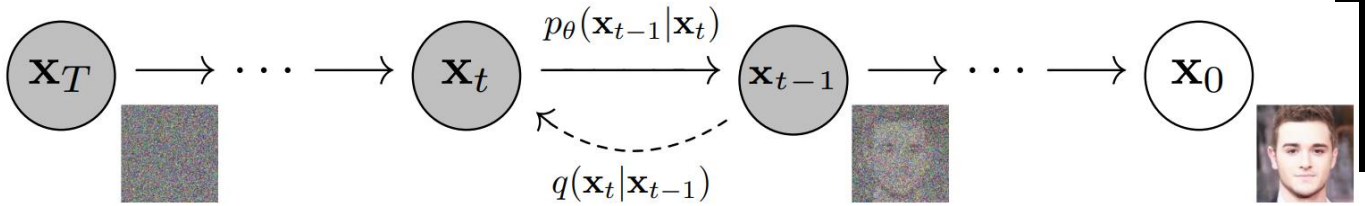
$$\Sigma(\mathbf{x}_t, t) = \sigma_t^2\mathbb{I}$$

$$\alpha_t = 1 - \beta_t$$

$$\bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i$$

$$\epsilon \sim \mathcal{N}(0, \mathbb{I})$$

# Diffusion Models




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

$q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ Intractable ☹ (depends on entire dataset)

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

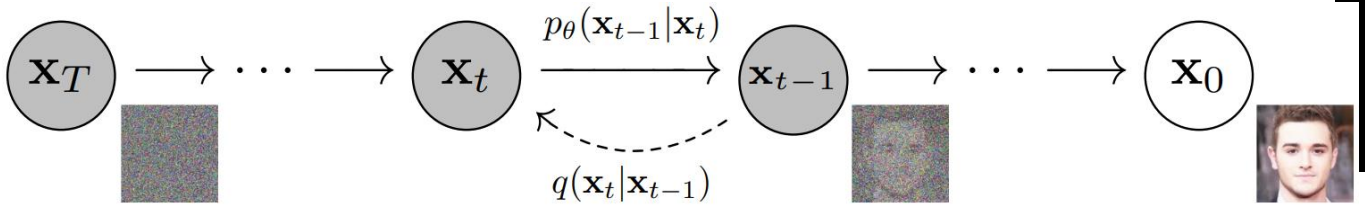
$$\mu(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right)$$

$$\alpha_t = 1 - \beta_t$$

$$\bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i$$

$$\epsilon \sim \mathcal{N}(0, \mathbb{I})$$

# Diffusion Models




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

$q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ Intractable ☹ (depends on entire dataset)

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

$$\mu(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t$$

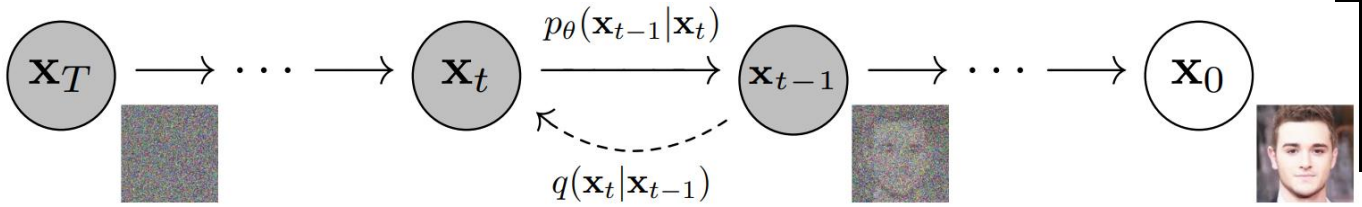
**Algorithm 1** Training

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

$\alpha_i$

$\mathbb{I})$

# Diffusion Models




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

$q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ Intractable ☹ (depends on entire dataset)

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

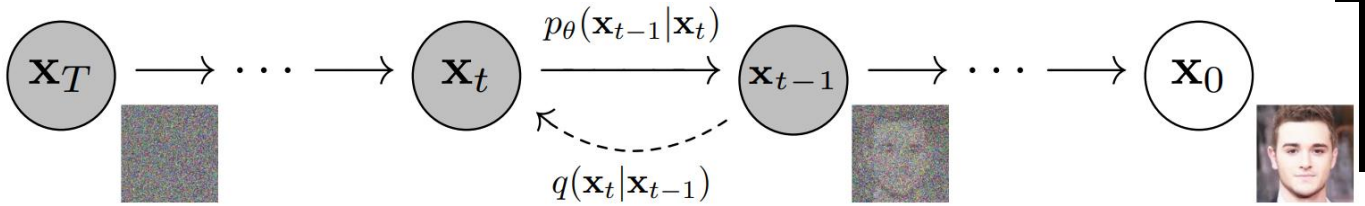
Train by maximizing log-likelihood:

$$L_{\text{CE}} = \mathbb{E}_{q(\mathbf{x}_0)}[-\log p_\theta(\mathbf{x}_0)]$$

$$= \ \ldots \text{ tons of algebra } \ldots$$

$$\leq [\text{terms of KL/entropy of Gaussians}](\theta)$$

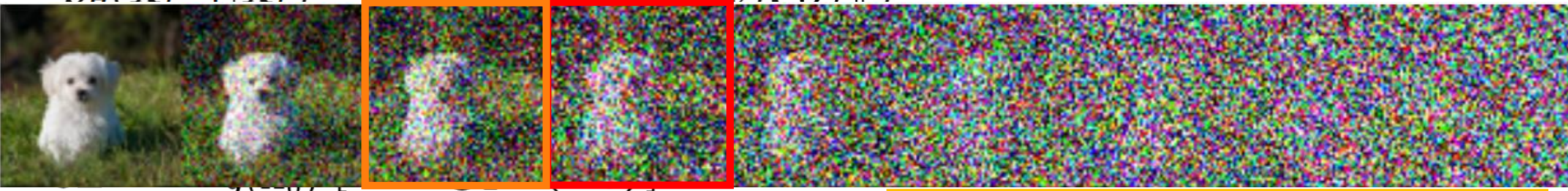
# Diffusion Models




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

$q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ Intractable ☹ (depends on entire dataset)

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



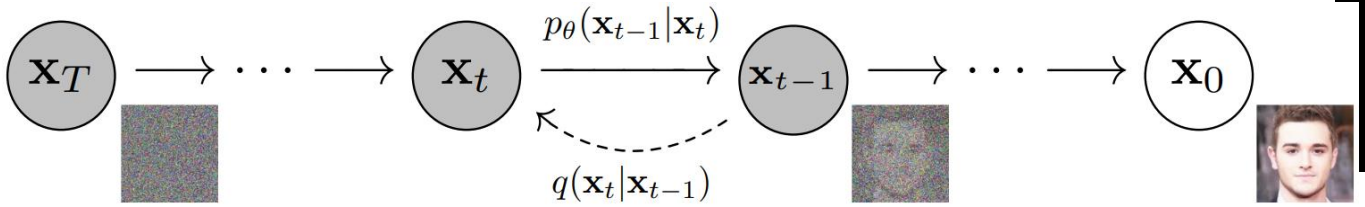
$$= \ldots \text{ tons of algebra } \ldots$$

$$\leq \ldots + D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))$$

$$= q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)}$$

Tractable!

# Diffusion Models




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

$q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ Intractable ☹ (depends on entire dataset)

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

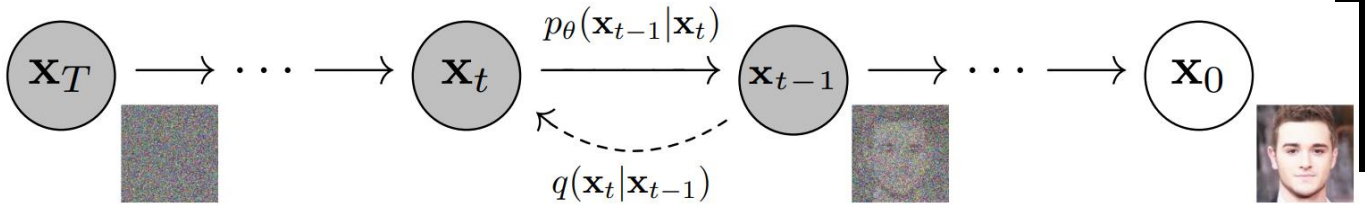
Train by maximizing log-likelihood:

$$L_{\text{CE}} = \mathbb{E}_{q(\mathbf{x}_0)}[-\log p_\theta(\mathbf{x}_0)]$$

$$= \ \ldots \text{ tons of algebra } \ldots$$

$$\leq \ \ldots + D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))$$

# Diffusion Models




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

$q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ Intractable ☹ (depends on entire dataset)

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

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

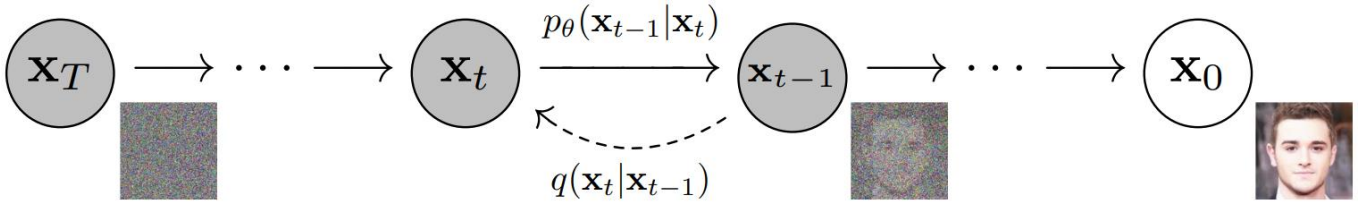
$$\alpha_t = 1 - \beta_t$$

$$\bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i$$

$$\epsilon \sim \mathcal{N}(0, \mathbb{I})$$

$$L_{\text{CE}} = \mathbb{E}_{q(\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_{t,x_0,\epsilon}\left[\|\epsilon - \epsilon_\theta(\mathbf{x}_t, t)\|^2\right]$$

# Diffusion Models



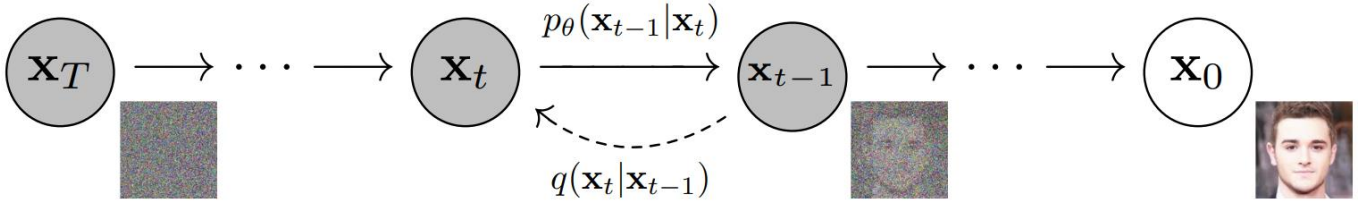

**Algorithm 1** Training

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



"Denoising Diffusion Probabilistic Models" [Ho et al., 2020]

# Diffusion Models




**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
$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta( \quad \mathbf{x}_t \quad , t) \right\|^2$
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** $\mathbf{x}_0$

Computed from x0

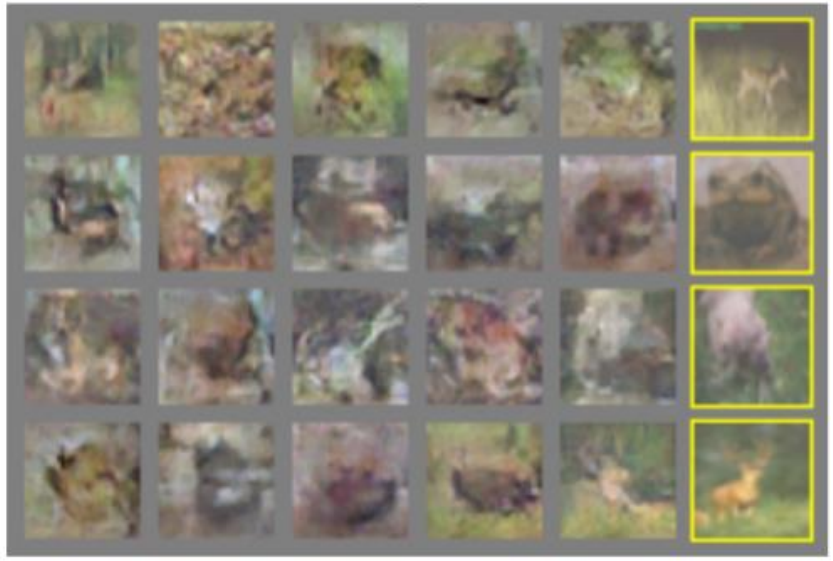
(by adding noise - analytically)



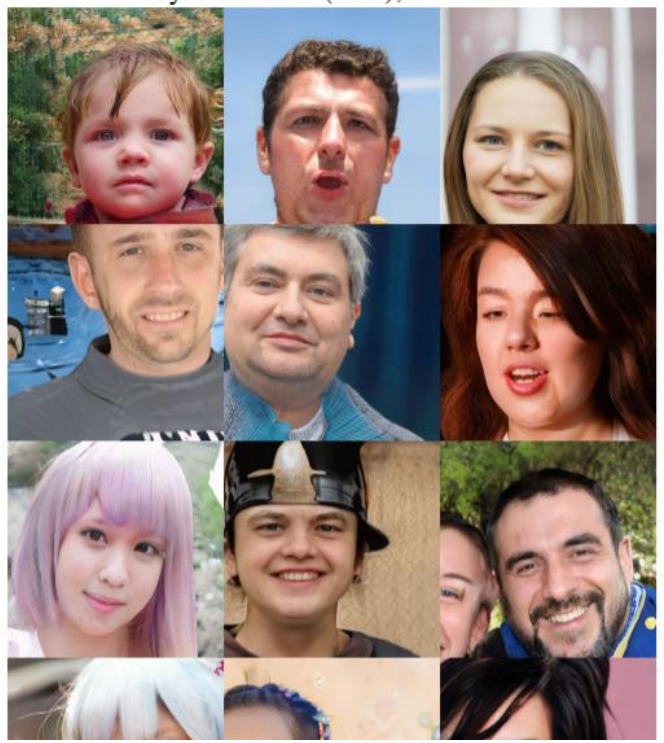
"Denoising Diffusion Probabilistic Models" [Ho et al., 2020]

# GANs – 2014 → 2021

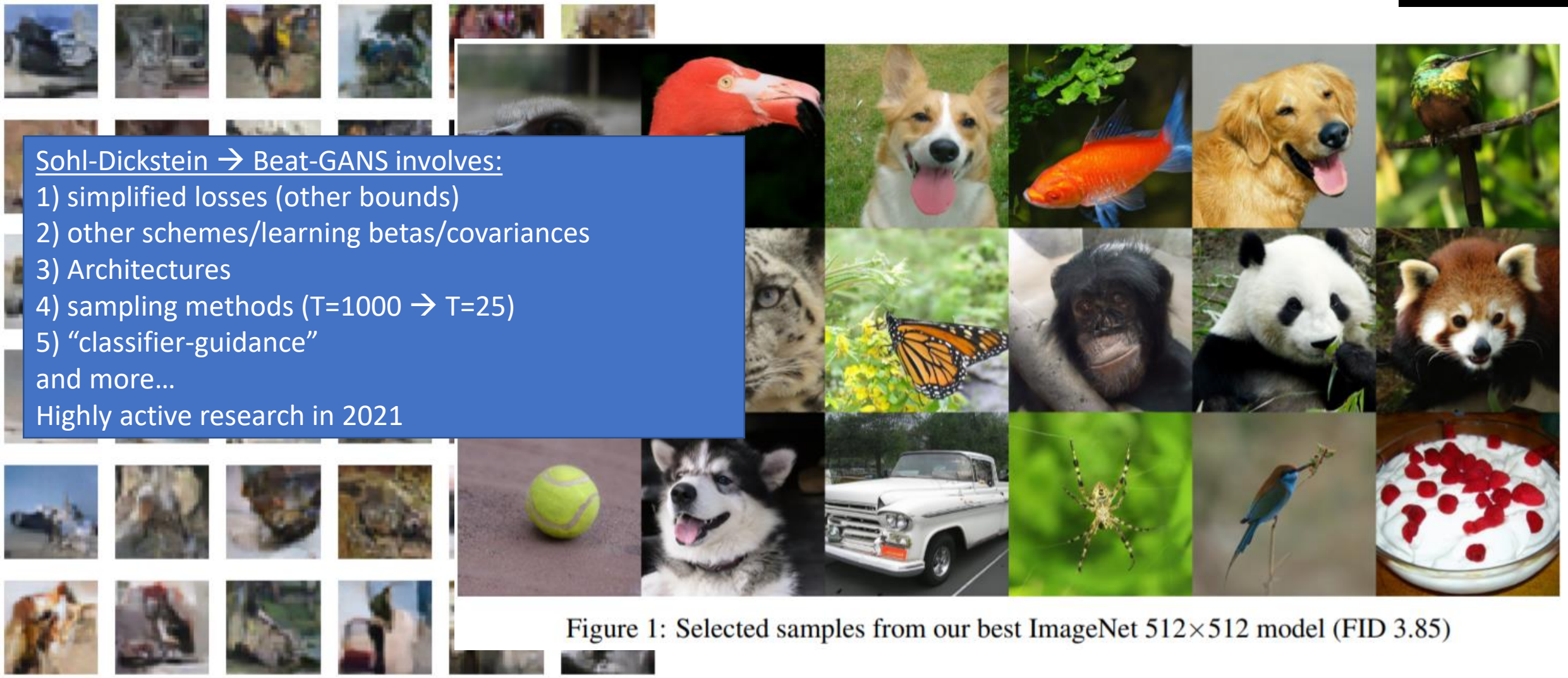
b)



StyleGAN3-T (ours), FID 3.67



# Diffusion Models



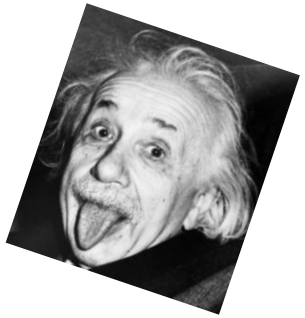
Sohl-Dickstein → Beat-GANS involves:
1) simplified losses (other bounds)
2) other schemes/learning betas/covariances
3) Architectures
4) sampling methods (T=1000 → T=25)
5) "classifier-guidance"
and more…
Highly active research in 2021

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

"Deep Unsupervised Learning using Nonequilibrium Thermodynamics" [Sohl-Dickstein et al. 2015]
"Diffusion Models Beat GANs on Image Synthesis " [Dhariwal & Nichol, 2021]

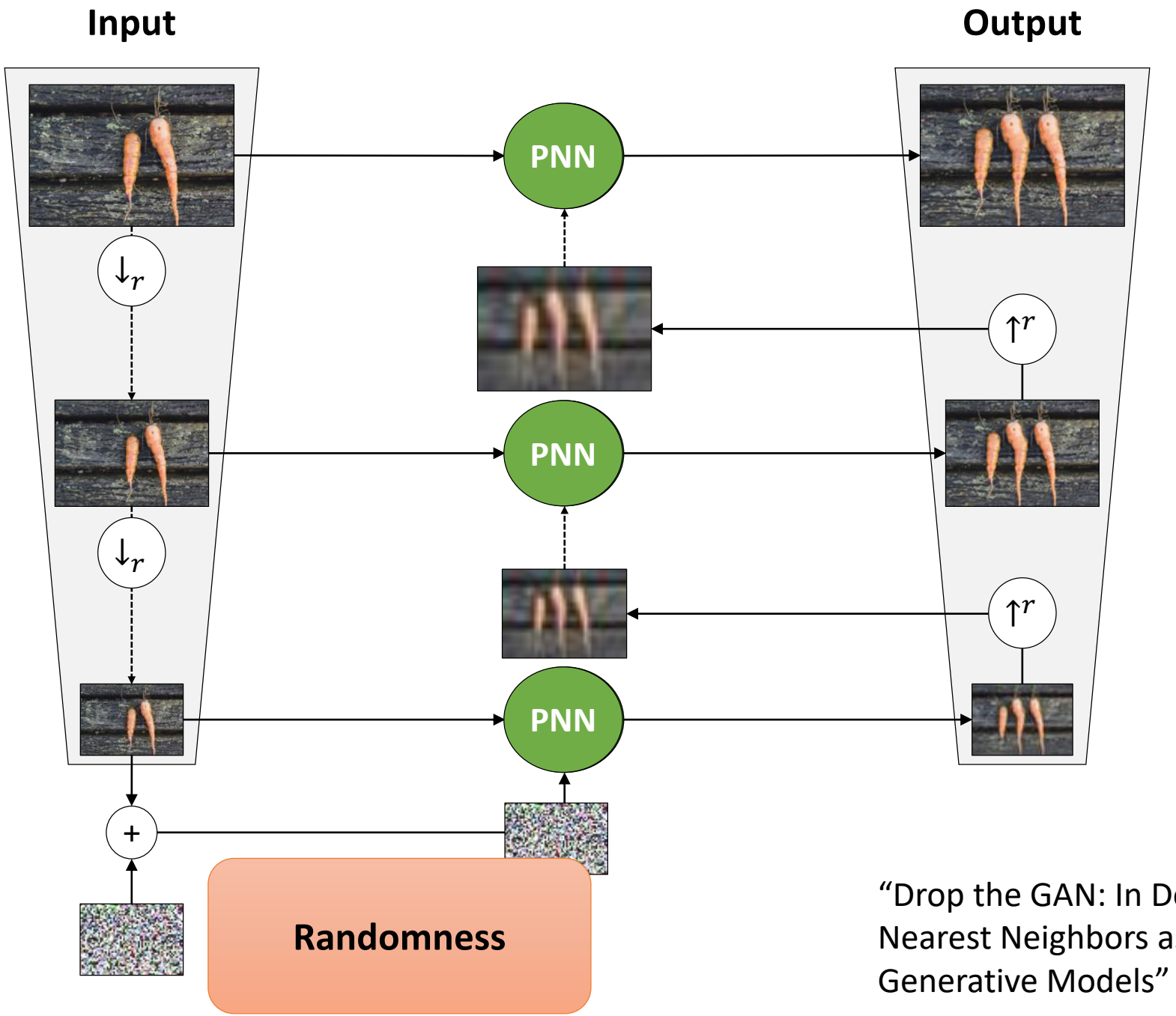
# Summary

- Autoencoders (AE)
- Variational Autoencoders (VAE)
- Vector Quantized VAE (VQ-VAE)
- VQGAN
- Diffusion Models



- Epilogue: PNNs
- Bonus: ..

# GPNN




"Drop the GAN: In Defense of Patches Nearest Neighbors as Single Image Generative Models" [Granot et al., 2021]

# Diverse images generated from a single image

Input

Generated

"Drop the GAN: In Defense of Patches Nearest Neighbors as Single Image Generative Models" [Granot et al., 2021]

Source Image

**GPNN**

SinGAN

# Diverse videos generated from a single video

Original Video (the rest are generated)

"Diverse Generation from a Single Video Made Possible" [Haim et al., 2021]

# Diffusion Models + CLIP

"a hedgehog using a calculator"

"a corgi wearing a red bowtie and a purple party hat"

"robots meditating in a vipassana retreat"

"a fall landscape with a small cottage next to a lake"

"a surrealist dream-like oil painting by salvador dalí of a cat playing checkers"

"a professional photo of a sunset behind the grand canyon"

"a high-quality oil painting of a psychedelic hamster dragon"

"an illustration of albert einstein wearing a superhero costume"

"GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models" [Nichol et al., 20/12/2021..

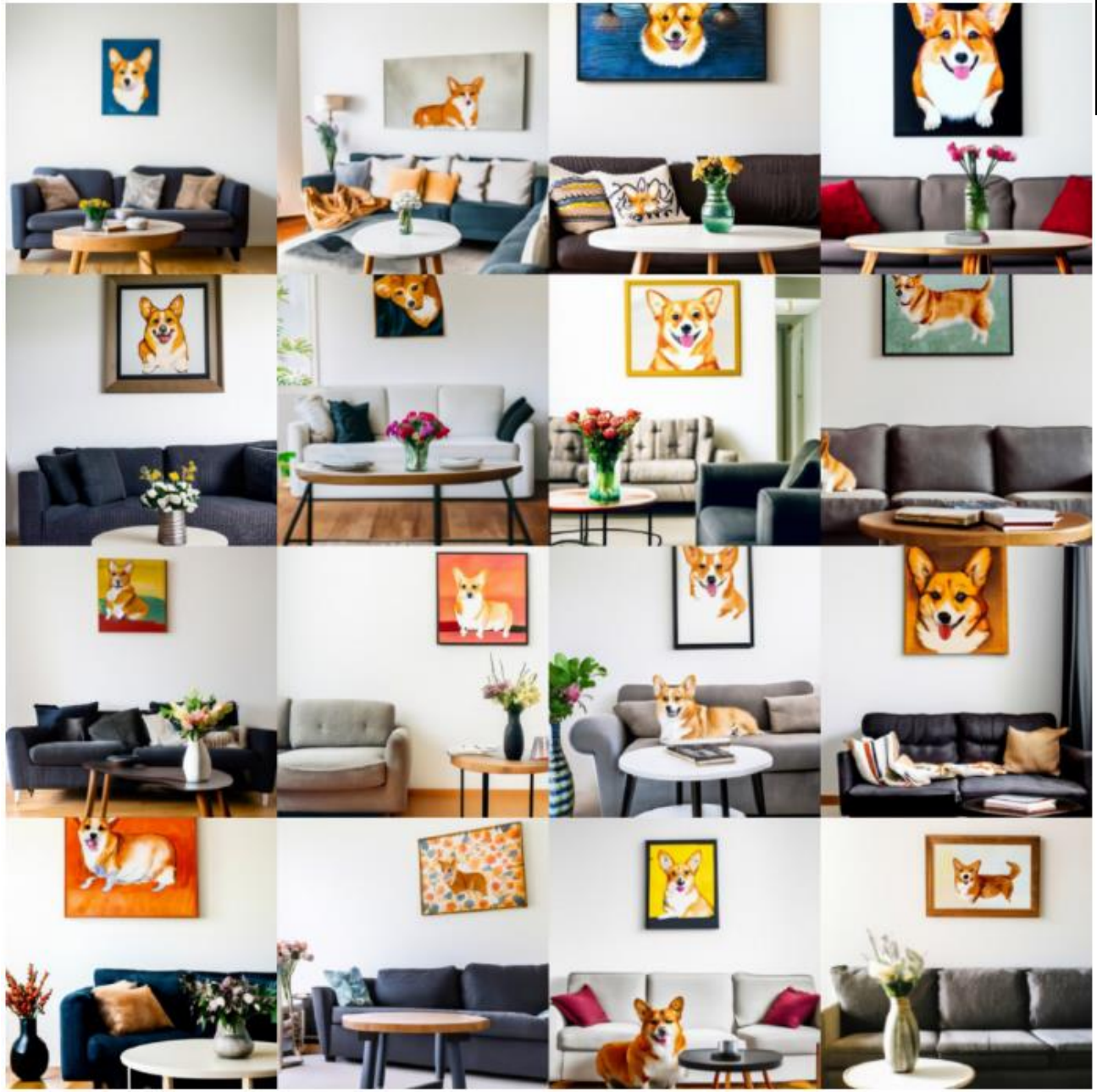
# Diffusion Models + CLIP

on the prompt “a stained glass window of a panda eating bamboo”.

“GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models“ [Nichol et al., 20/12/2021..]

# Diffusion Models + CLIP

Prompt:
“A cozy living room with a painting of a corgi on the wall above a couch and a round coffee table in front of a couch and a vase of flowers on a coffee table”



“GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models“ [Nichol et al., 20/12/2021..]

# Next Week:

Learning From Videos