University of Toronto
Department of Computer Science

CSC420

Introduction to Image Understanding

Tutorial 4: VAEs and diffusion models

Fall 2026

Presented by Javad Rajabi
Teaching Assistant

Instructor: David Lindell



t = 0

CSC420 Tutorial 4

From noise to images

Variational autoencoders and diffusion models

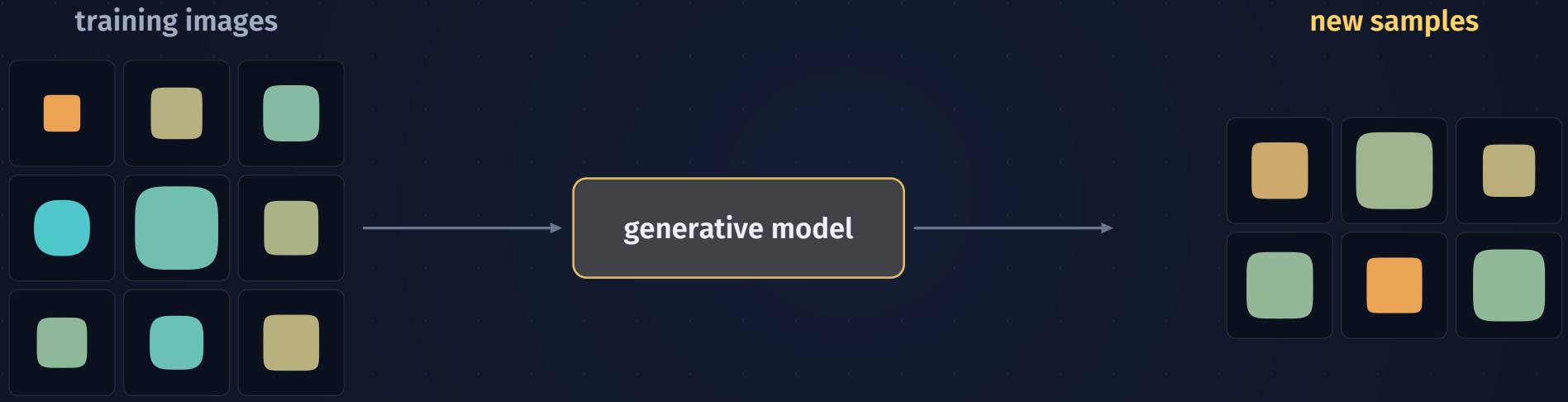
What we will cover

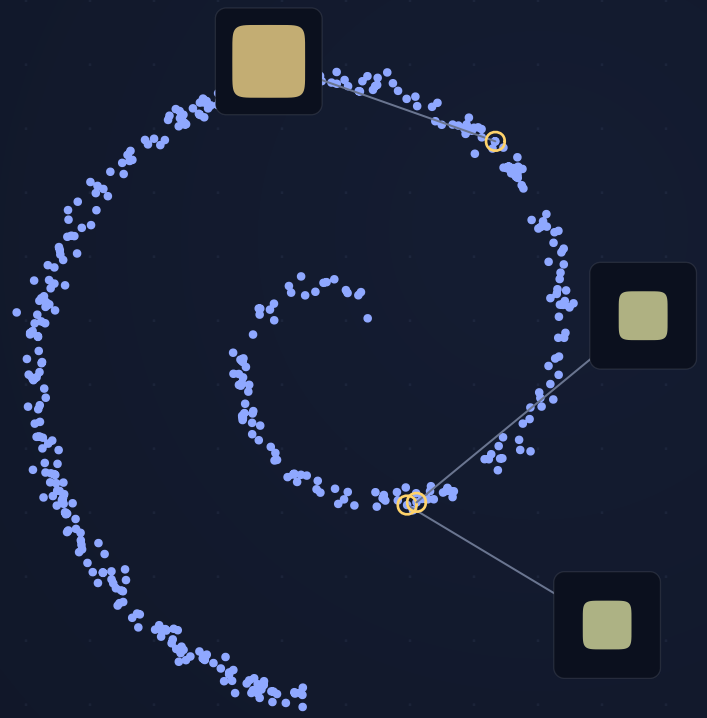
1. What is a generative model?
2. Maximum likelihood
3. KL divergence
4. Autoencoders and latent spaces
5. The variational autoencoder
6. The evidence lower bound
7. The reparameterization trick
8. VAEs in practice
9. Generative adversarial networks
10. Diffusion: the big idea
11. The forward process
12. Training a DDPM
13. Sampling with DDPM
14. DDIM: faster sampling
15. Score-based models
16. Flow matching
17. Conditioning and guidance
18. The U-Net
19. Latent diffusion and DiT
20. Recap

1 What is a generative model?

Learning to create new data

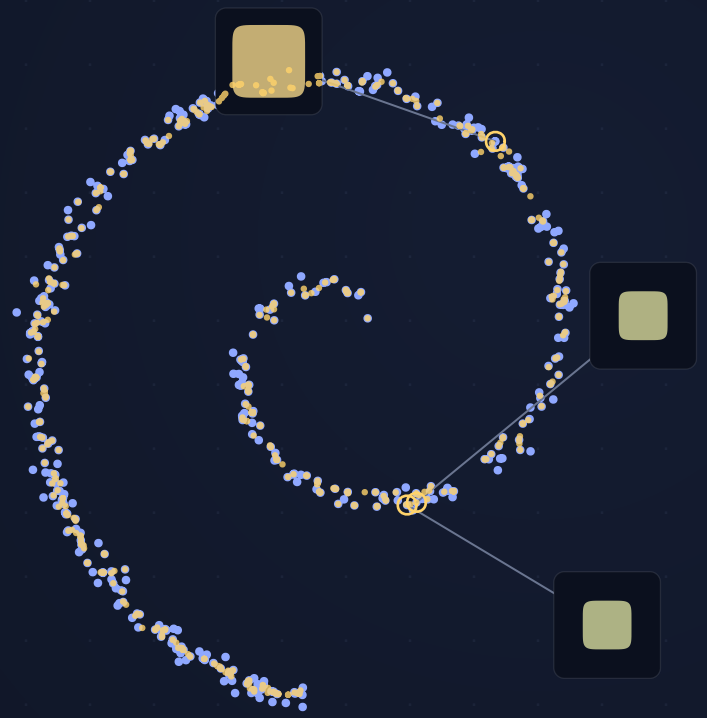
1 What is a generative model?





$$p_{\text{data}}(x)$$

each point is one image

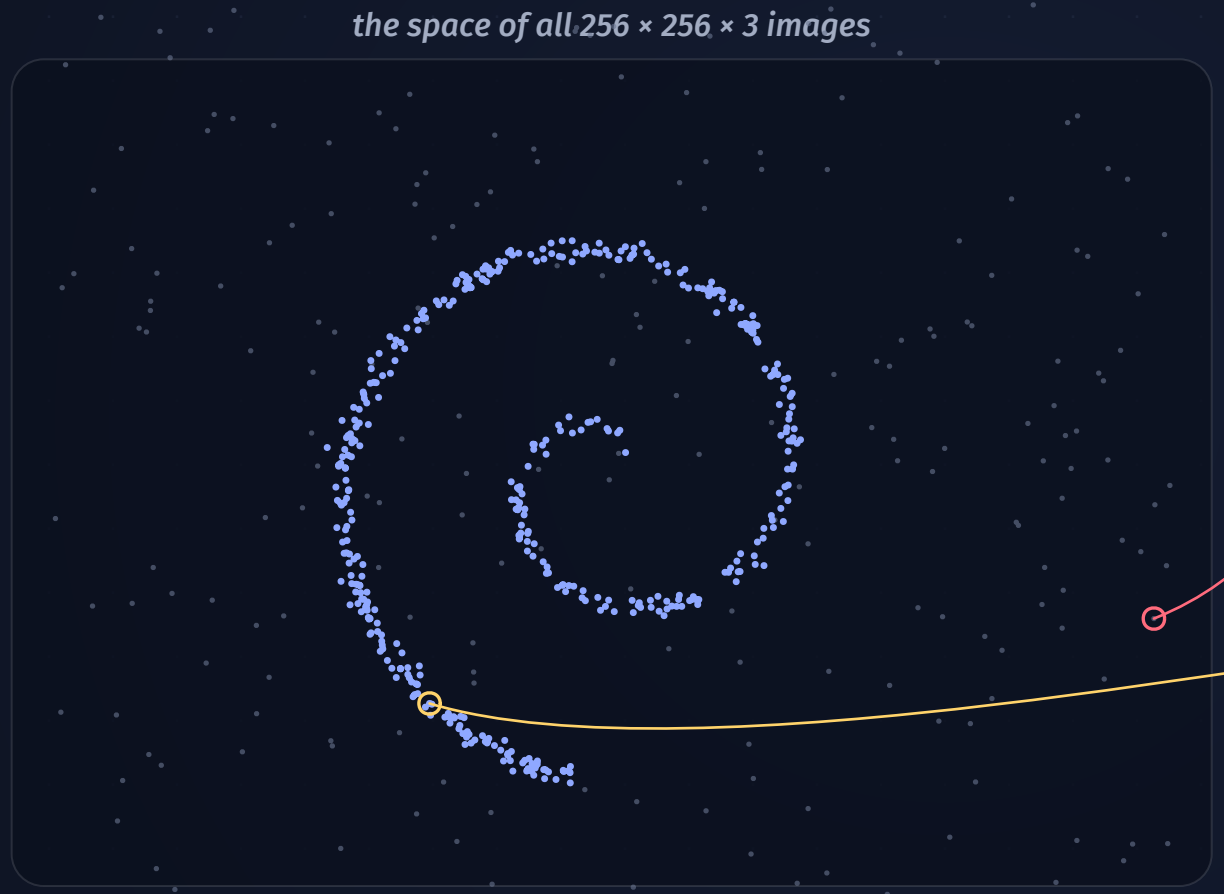


$$p_{\text{data}}(x)$$

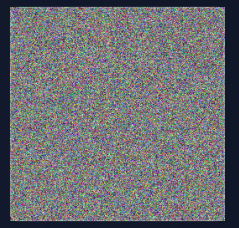
each point is one image

$$p_\theta(x) \approx p_{\text{data}}(x)$$

model samples (gold) cover the data



196,608 numbers per image



a random point: just noise



on the data: a real image

Variational autoencoders

latent variables

the ELBO and KL divergence

one-step generation

Diffusion models

DDPM and DDIM

score models

flow matching

2 Maximum likelihood

Making the data probable

$$p_\theta(x) = \mathcal{N}(x;\ \mu, \sigma^2)$$

μ = −1.40 σ = 0.55

average log-likelihood

-6.44



$$p_\theta(x) = \mathcal{N}(x;\ \mu, \sigma^2)$$

μ = −1.40 σ = 0.55

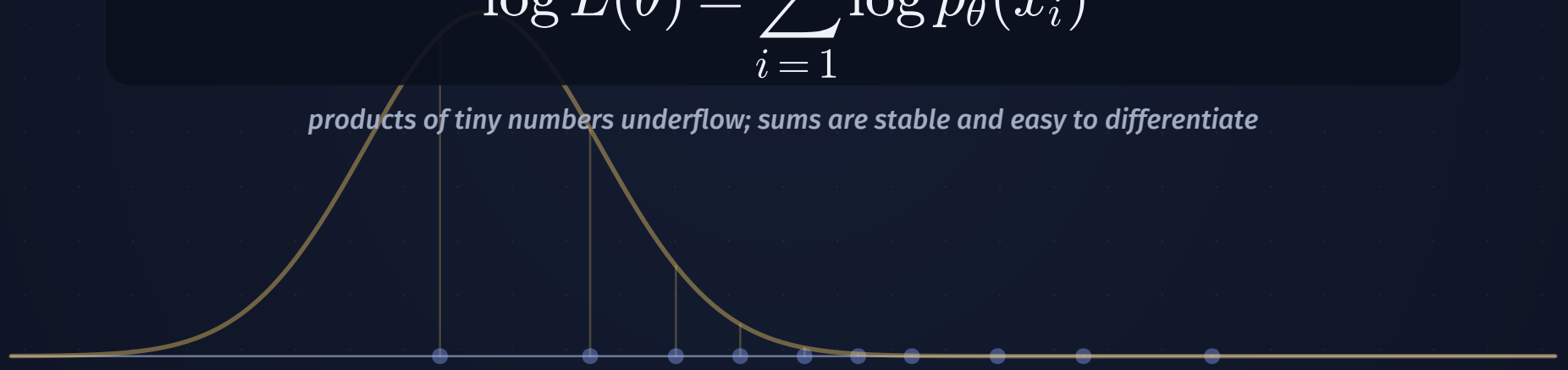
$$L(\theta) = \prod_{i=1}^{N} p_\theta(x_i)$$

$$\log L(\theta) = \sum_{i=1}^{N} \log p_\theta(x_i)$$

average log-likelihood

−6.44

products of tiny numbers underflow; sums are stable and easy to differentiate

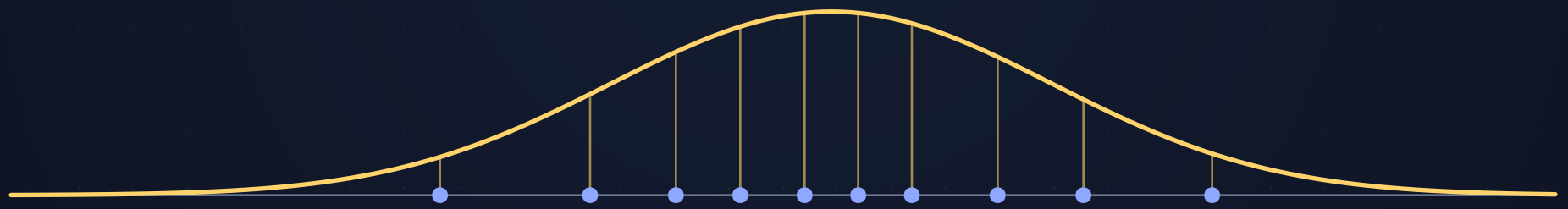


$$p_\theta(x) = \mathcal{N}(x;\ \mu, \sigma^2)$$

μ = 0.22 σ = 1.03

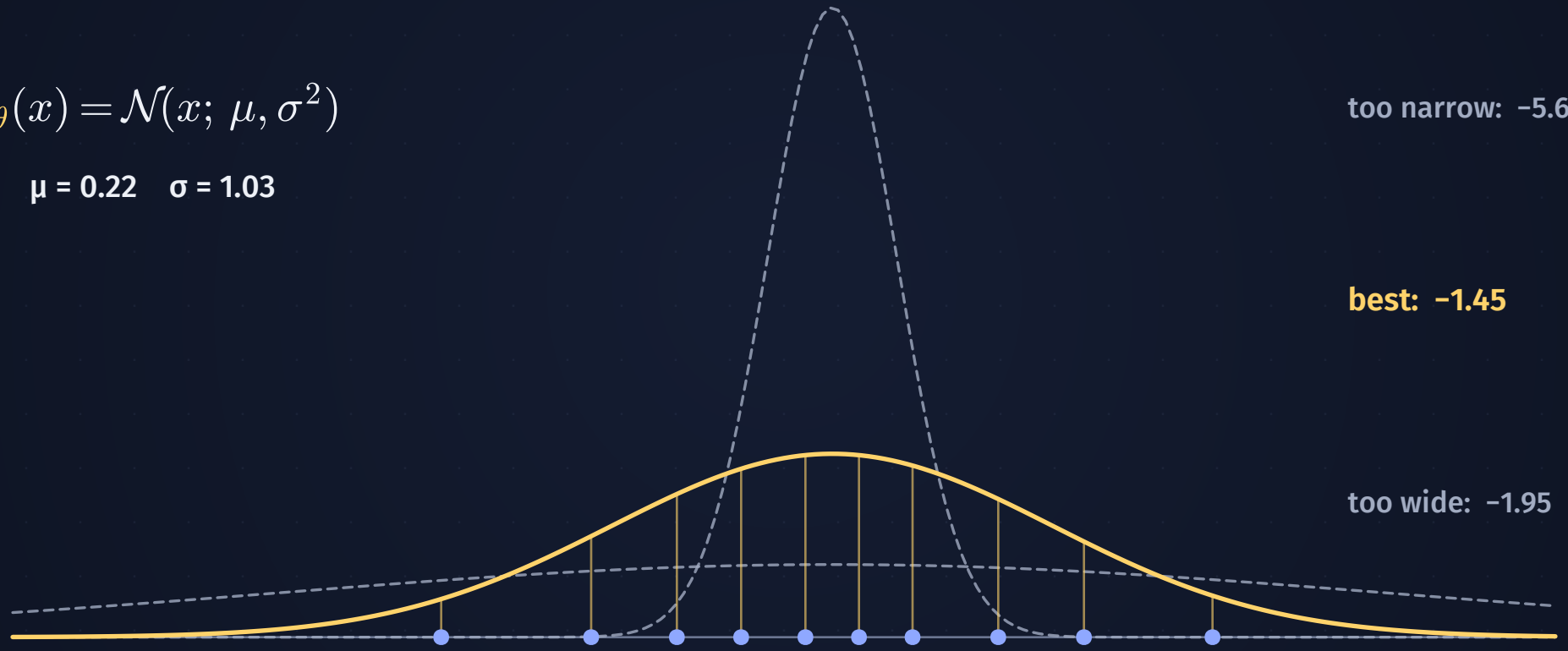
average log-likelihood

$$-1.45$$



$$p_\theta(x) = \mathcal{N}(x;\ \mu, \sigma^2)$$

μ = 0.22 σ = 1.03



too narrow: −5.61

best: −1.45

too wide: −1.95

$$\theta^* = \arg\max_\theta \sum_i \log p_\theta(x_i)$$



deep network

$$\log p_\theta(x) = ?$$

often intractable

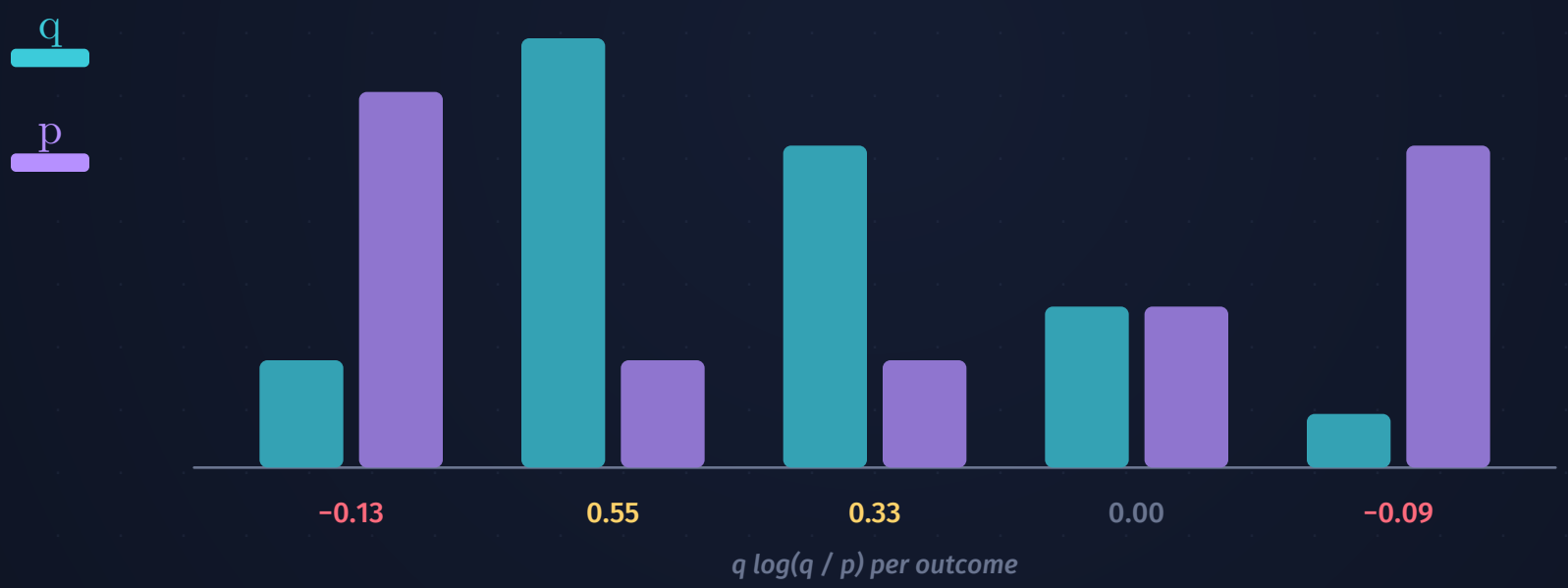
3 KL divergence

Measuring how distributions differ

$$D_{\mathrm{KL}}(q \,\|\, p) = \sum_{x} q(x) \log \frac{q(x)}{p(x)}$$

$$= \mathbb{E}_{x \sim q}\left[\log \frac{q(x)}{p(x)}\right]$$

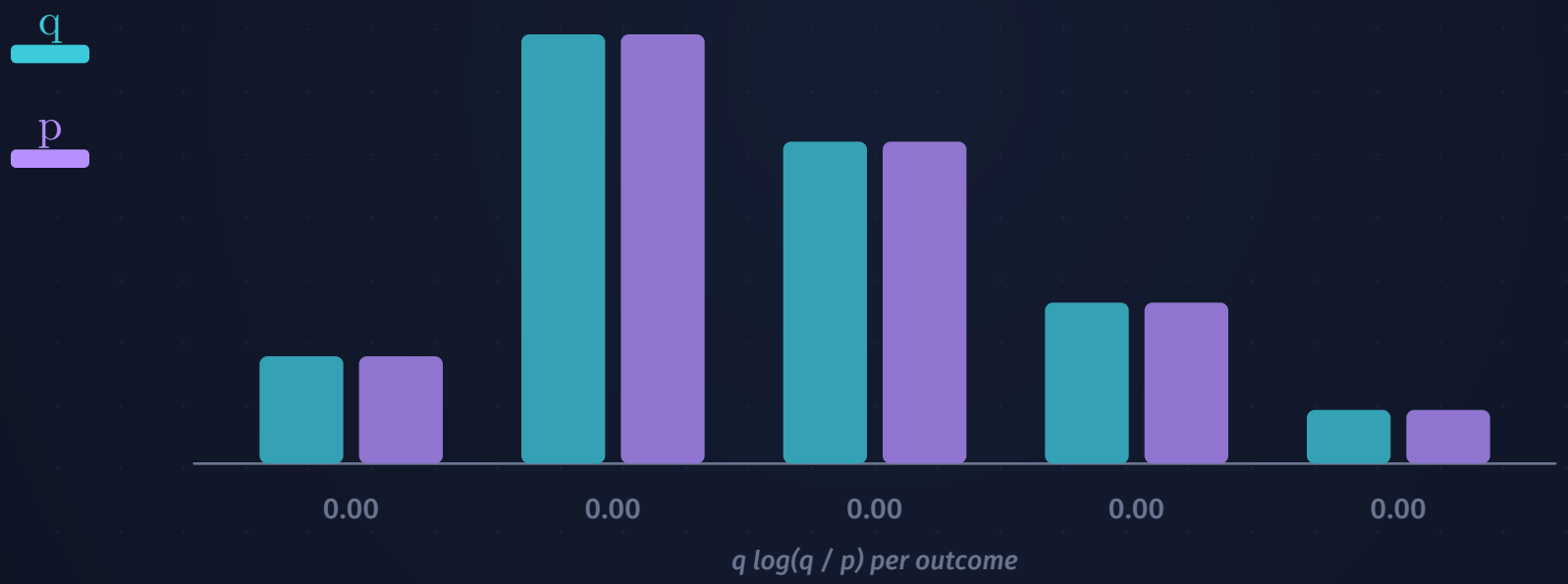
$$D_{\mathrm{KL}}(q \,\|\, p) = \sum_{x} q(x) \log \frac{q(x)}{p(x)}$$



KL(q || p)

0.67

$$D_{\mathrm{KL}}(q \,\|\, p) = \sum_{x} q(x) \log \frac{q(x)}{p(x)}$$



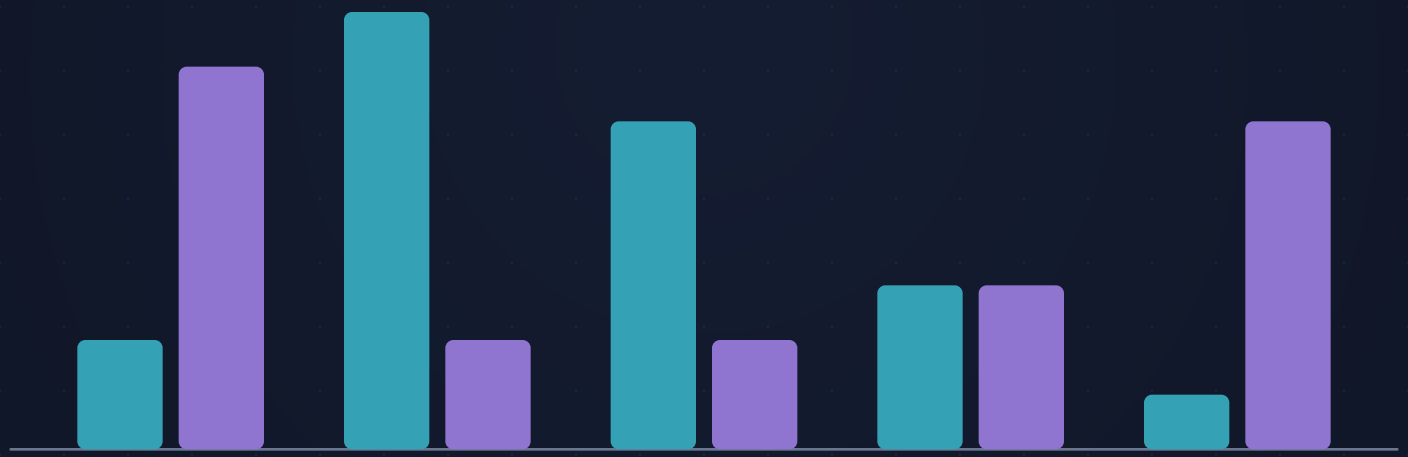
KL(q || p)

0.00

$$D_{\mathrm{KL}}(q \,\|\, p) = \sum_x q(x) \log \frac{q(x)}{p(x)}$$

q

p



q log(q / p) per outcome

KL(q || p)

0.67

KL(p || q) = 0.73

never negative

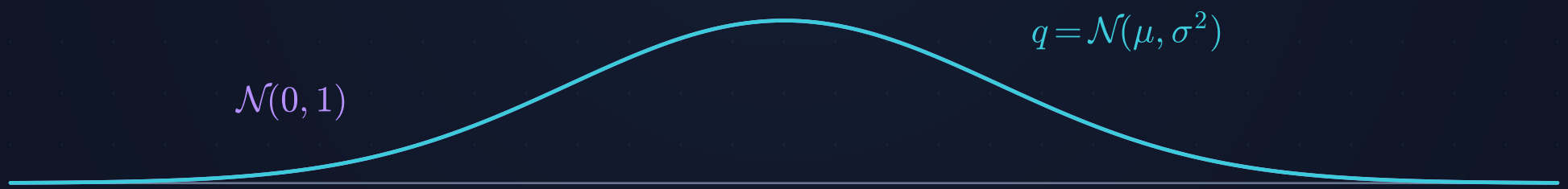
not symmetric

$$D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_\theta) = \underbrace{\mathbb{E}_{p_{\mathrm{data}}}[\log p_{\mathrm{data}}(x)]}_{\text{a constant: does not depend on } \theta} - \underbrace{\mathbb{E}_{p_{\mathrm{data}}}[\log p_\theta(x)]}_{\text{expected log-likelihood}}$$

$$\arg\min_\theta \; D_{\mathrm{KL}}(p_{\mathrm{data}} \,\|\, p_\theta) \;=\; \arg\max_\theta \; \mathbb{E}_{p_{\mathrm{data}}}[\log p_\theta(x)]$$

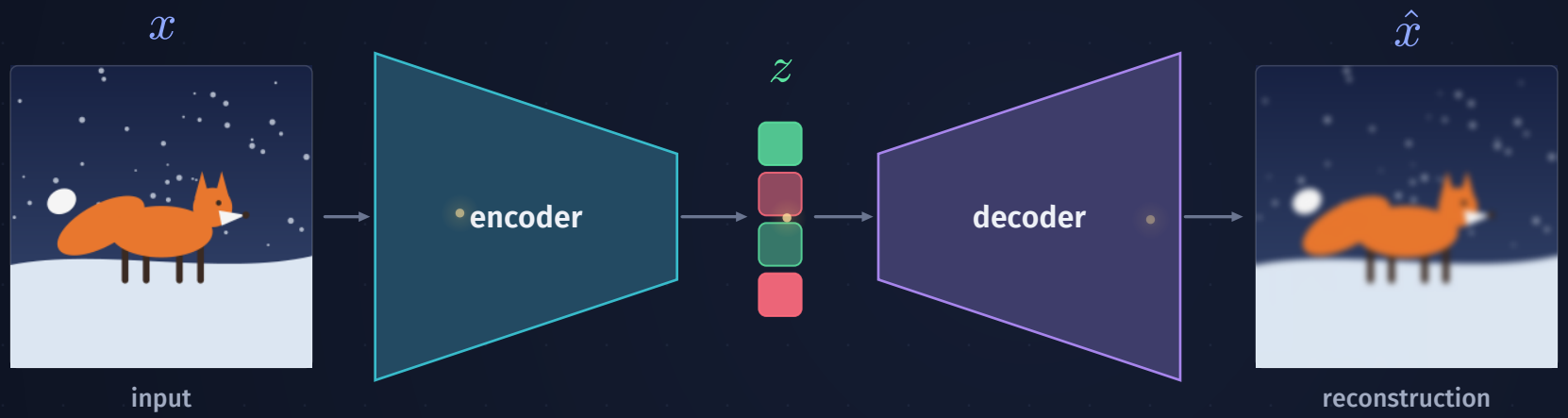
$$D_{\mathrm{KL}}\left(\mathcal{N}(\mu, \sigma^2) \,\|\, \mathcal{N}(0, 1)\right) = \tfrac{1}{2}\left(\mu^2 + \sigma^2 - \log \sigma^2 - 1\right)$$

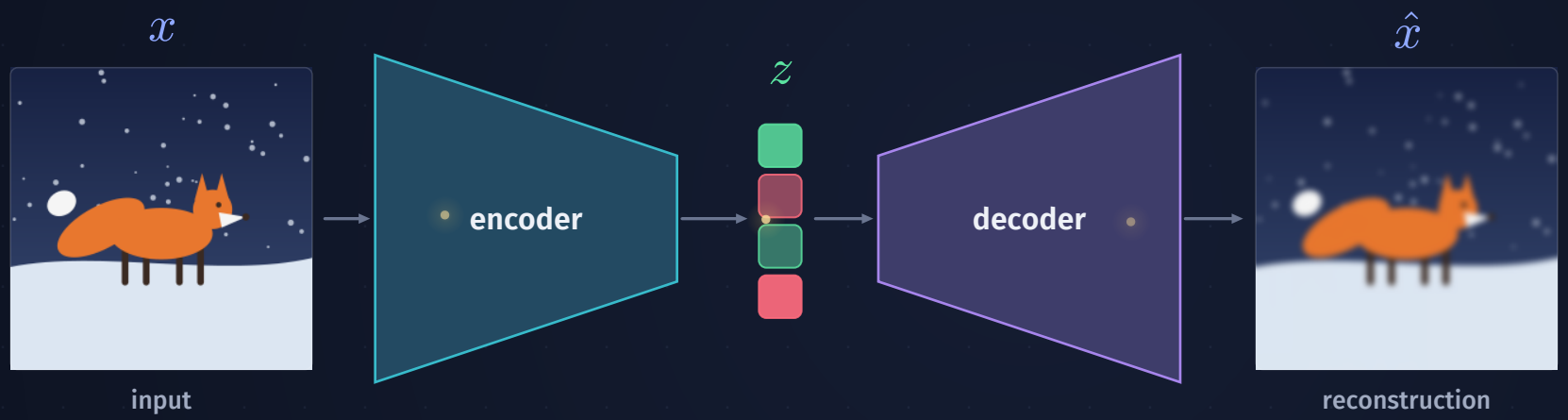
$\mathrm{KL} = 0.00$



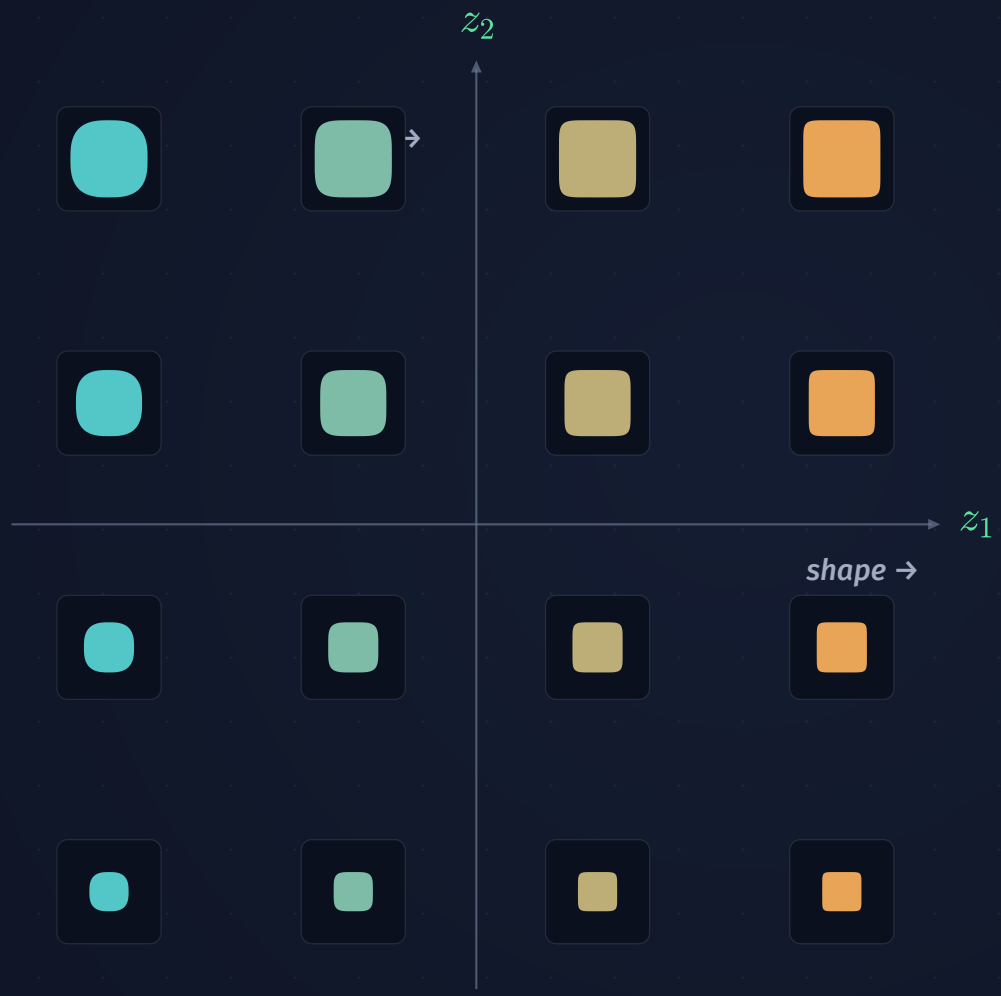
4 Autoencoders and latent spaces

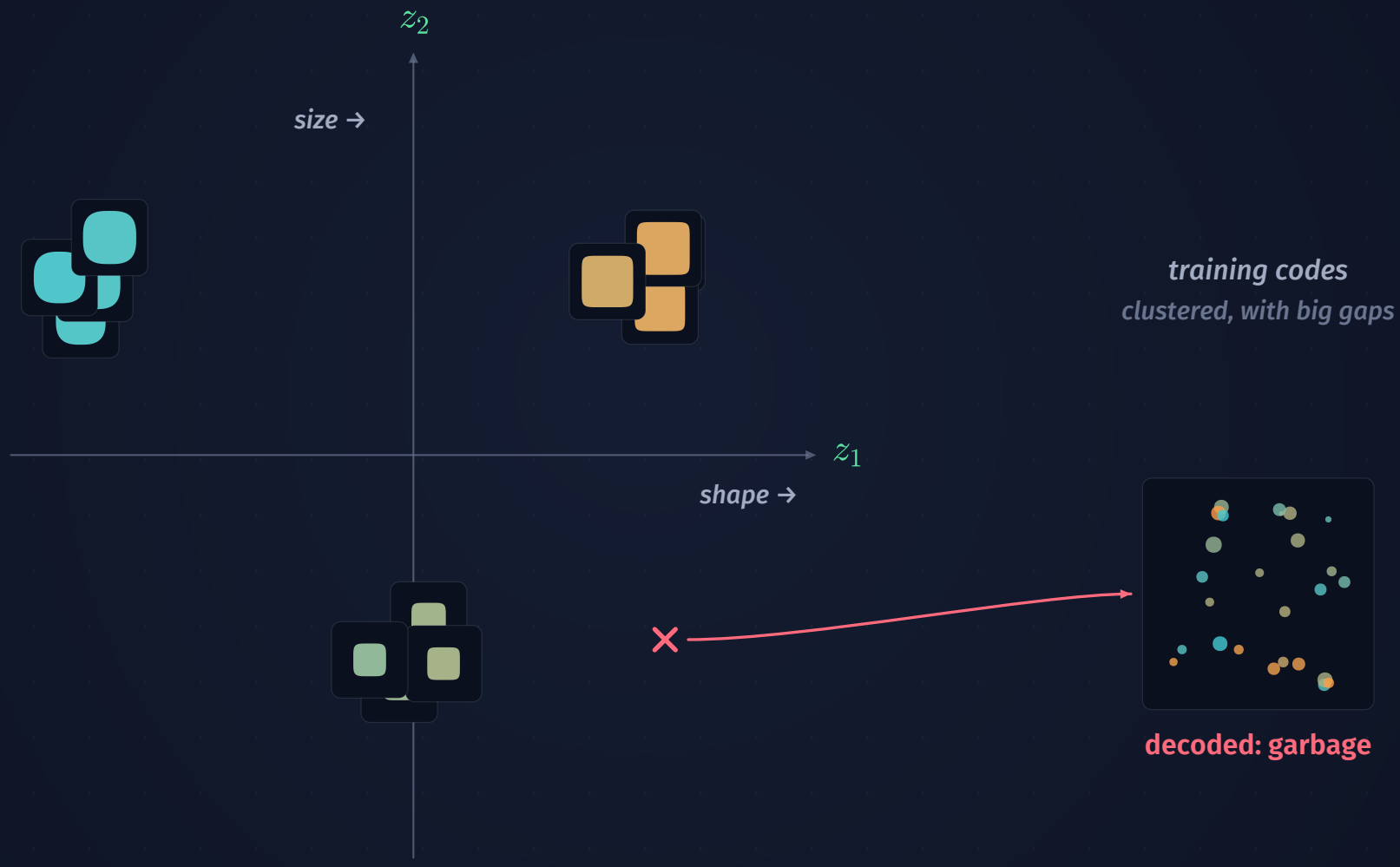
Compressing images into codes


x
encoder
z
decoder
$\hat{x}$
input
reconstruction



$$\mathcal{L} = \| x - \mathrm{Dec}(\mathrm{Enc}(x)) \|^2$$


z_2
z_1
shape →


z_2
size →
z_1
shape →
training codes
clustered, with big gaps
decoded: garbage

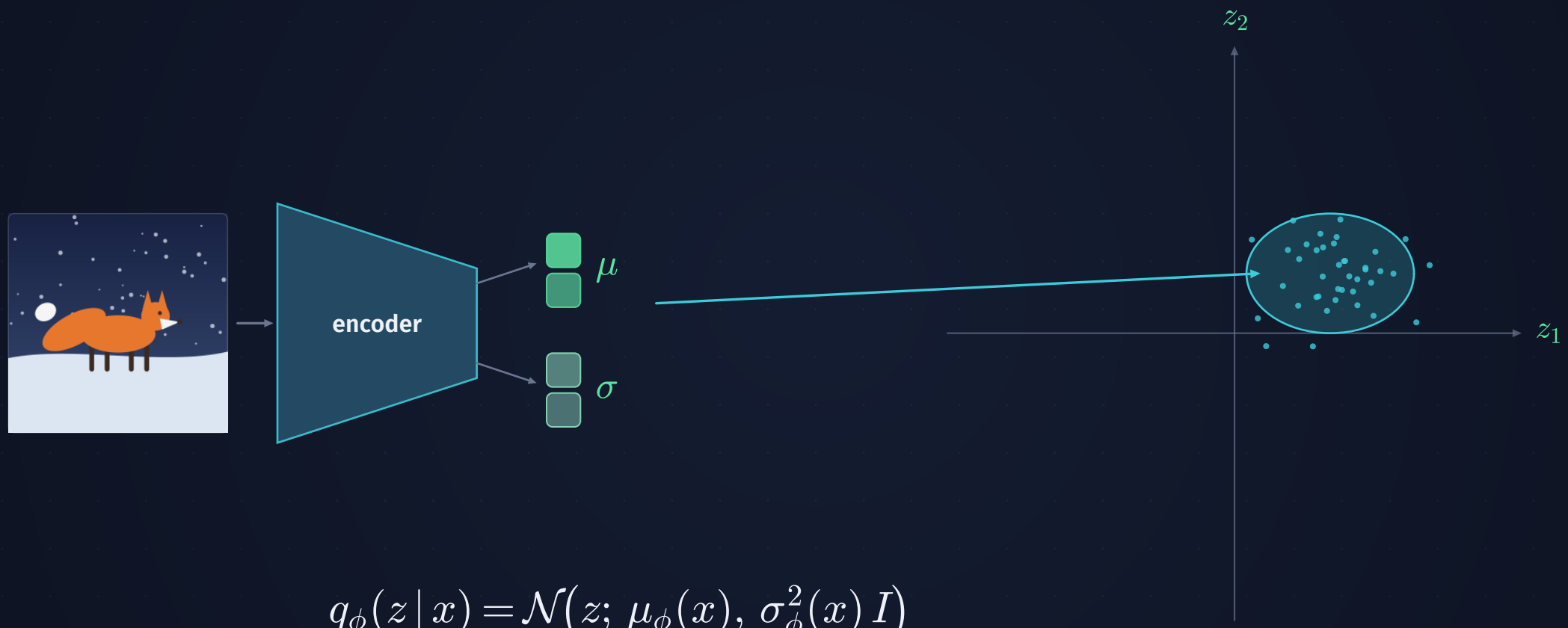


Wanted:

a latent space we can sample from

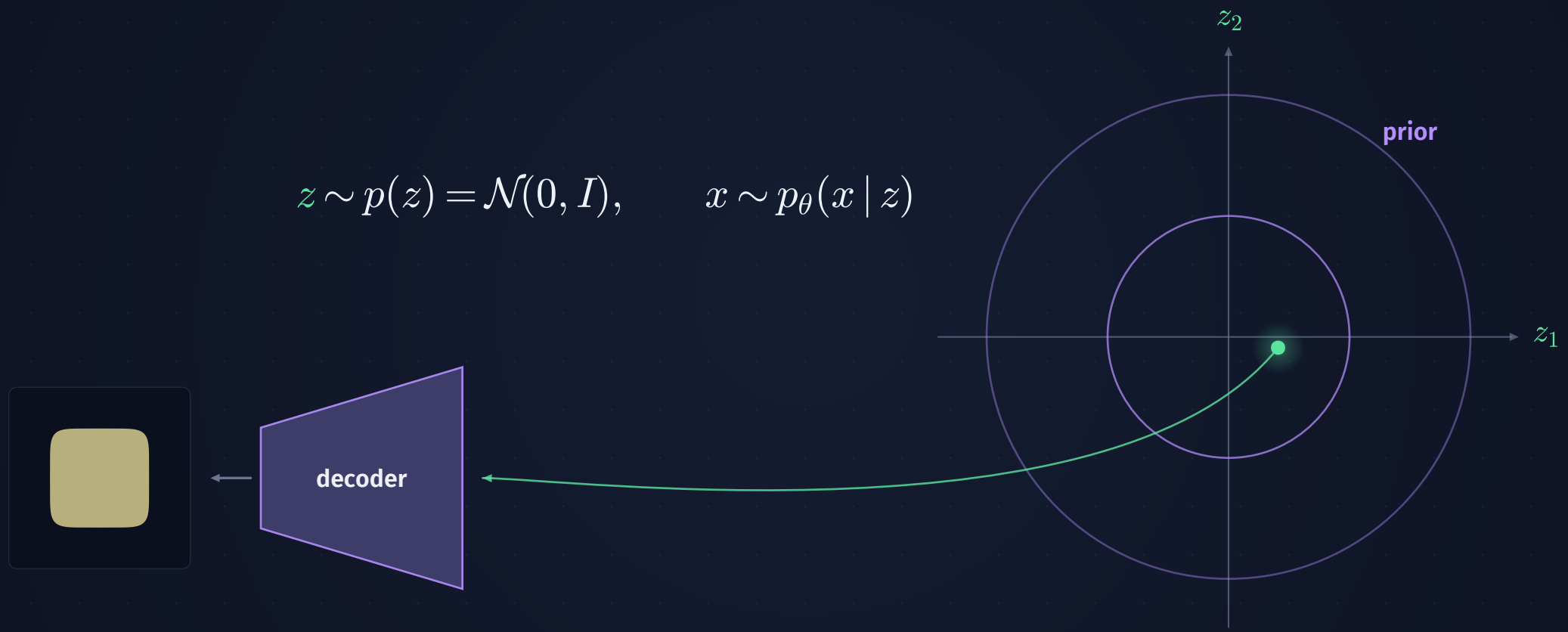
5 The variational autoencoder

A probabilistic encoder and decoder



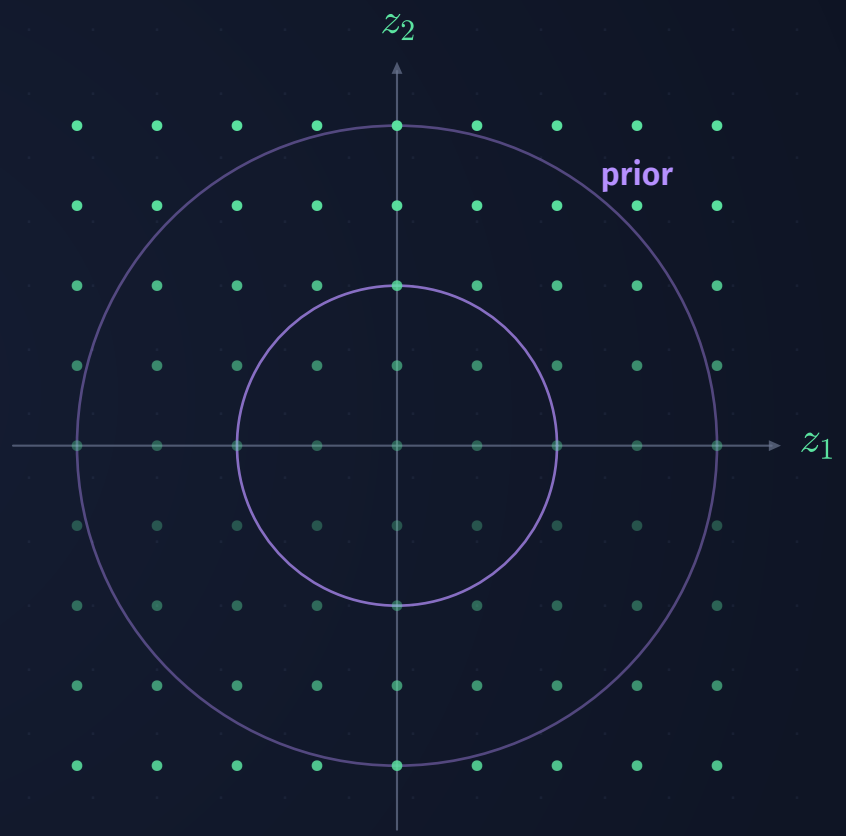
$$q_\phi(z \mid x) = \mathcal{N}\big(z;\, \mu_\phi(x),\, \sigma^2_\phi(x)\, I\big)$$

$$z \sim p(z) = \mathcal{N}(0, I), \qquad x \sim p_\theta(x \mid z)$$



$$p_\theta(x) = \int p_\theta(x \mid z)\, p(z)\, dz$$

intractable: a sum over every possible z

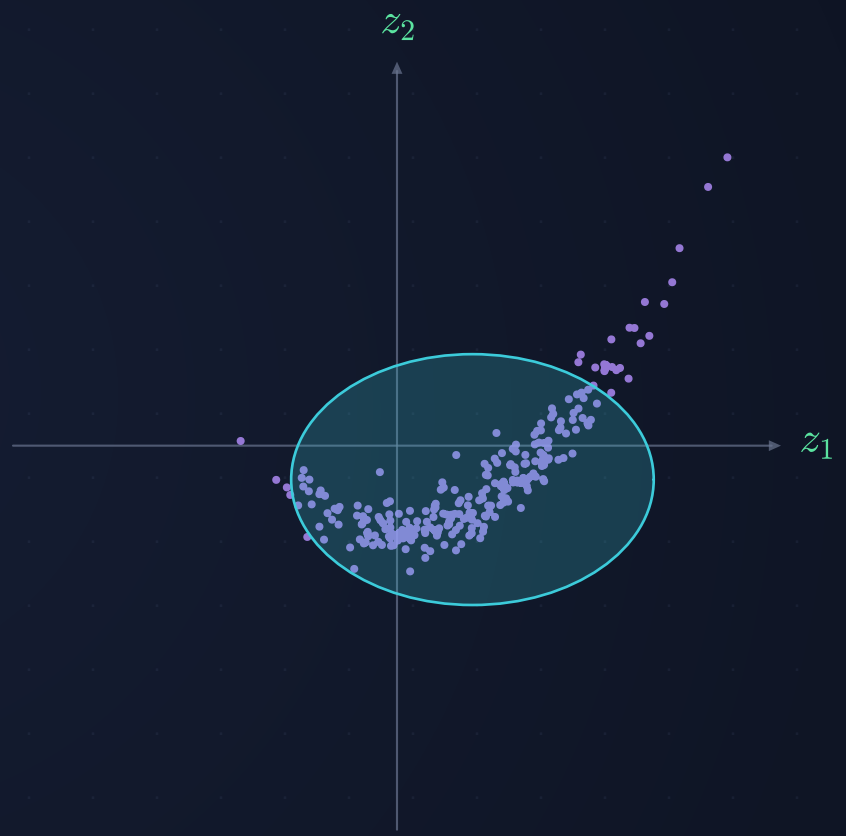


$$p_\theta(z \mid x)$$

true posterior: intractable

$$q_\phi(z \mid x)$$

Gaussian approximation



$$\max_{\theta} \log p_\theta(x) \longrightarrow \max_{\theta, \phi} \text{ELBO}$$

$$\text{ELBO} \leq \log p_\theta(x)$$

a lower bound we can compute and differentiate

6 The evidence lower bound

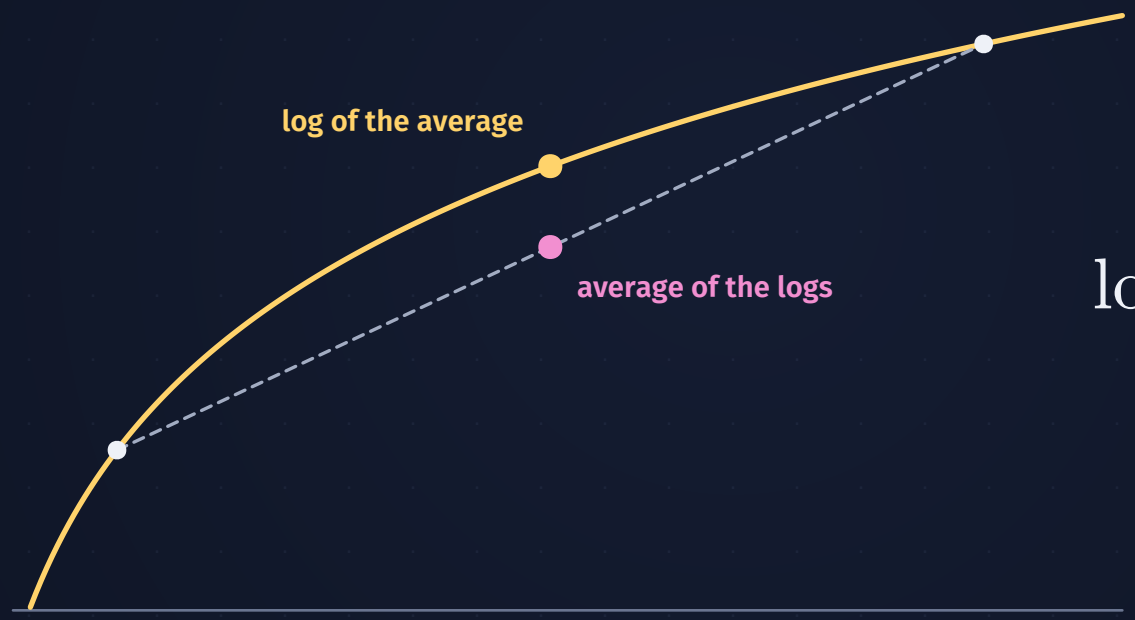
Deriving the ELBO

$$\log p_\theta(x) = \log \int p_\theta(x \mid z)\, p(z)\, dz$$

$$= \log \int q_\phi(z \mid x)\, \frac{p_\theta(x \mid z)\, p(z)}{q_\phi(z \mid x)}\, dz$$

$$= \log\ \mathbb{E}_{q_\phi(z|x)} \left[\frac{p_\theta(x \mid z)\, p(z)}{q_\phi(z \mid x)} \right]$$

$$= \log \; \mathbb{E}_{q_\phi(z|x)} \left[\frac{p_\theta(x \mid z)\, p(z)}{q_\phi(z \mid x)} \right]$$



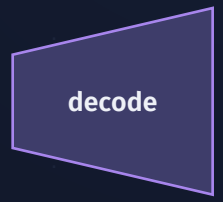
$$\log \mathbb{E}[Y] \geq \mathbb{E}[\log Y]$$

Jensen's inequality

$$\log p_\theta(x) \geq \underbrace{\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x \mid z)] - D_{\mathrm{KL}}(q_\phi(z \mid x) \,\|\, p(z))}_{\textbf{ELBO}}$$

$$\log p_\theta(x) \geq \underbrace{\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x \mid z)]}_{\text{reconstruction: does z explain x?}} - \underbrace{D_{\mathrm{KL}}(q_\phi(z \mid x) \,\|\, p(z))}_{\text{stay close to the prior}}$$





high when the decoded image matches x



q kept near N(0, I)

$$\log p_\theta(x) = \mathrm{ELBO} + D_{\mathrm{KL}}(q_\phi(z \mid x) \,\|\, p_\theta(z \mid x))$$



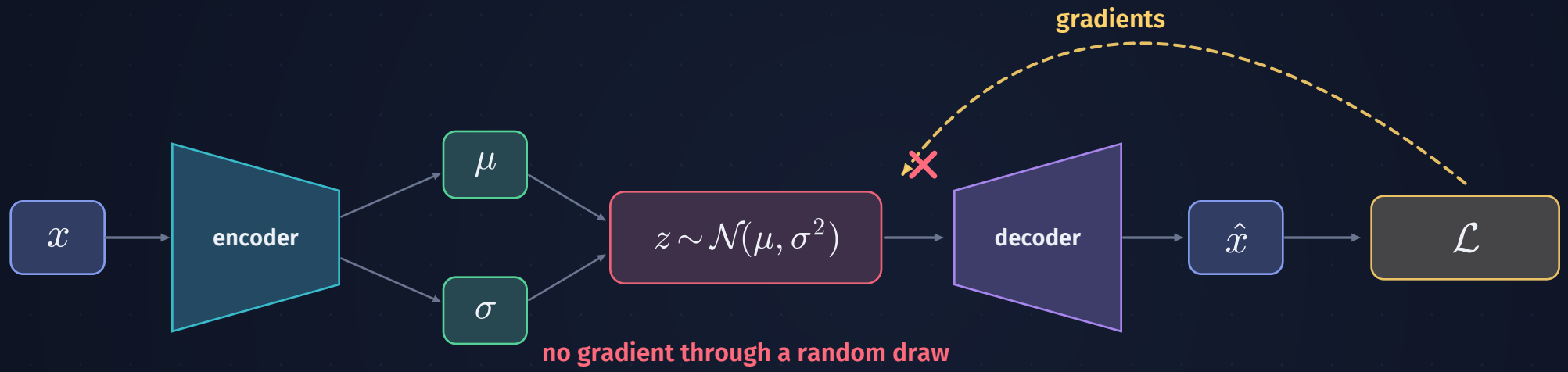
$$\mathcal{L}(\theta, \phi) = \underbrace{\| x - \hat{x} \|^2}_{\text{reconstruction error}} + \underbrace{\beta \cdot \frac{1}{2} \sum_j \left(\mu_j^2 + \sigma_j^2 - \log \sigma_j^2 - 1 \right)}_{\text{KL to N(0, I), closed form}}$$

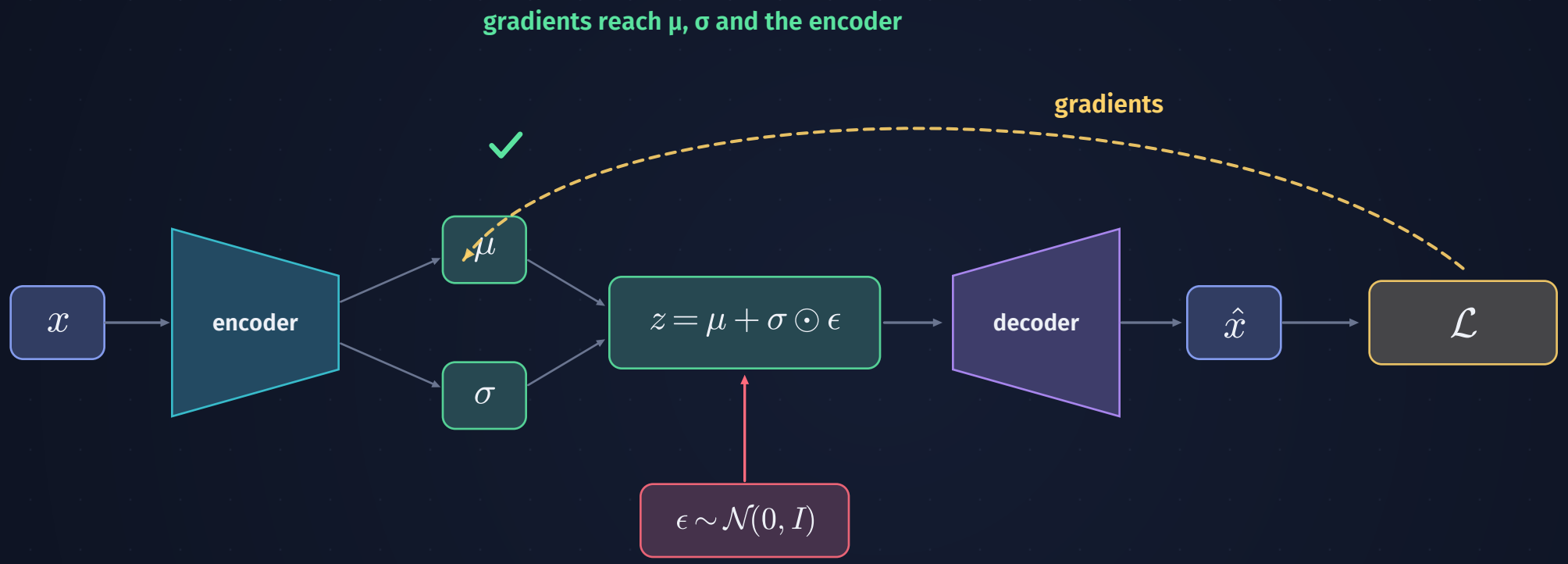
we minimize the negative ELBO (up to constants)

larger β: a smoother latent space, but blurrier reconstructions

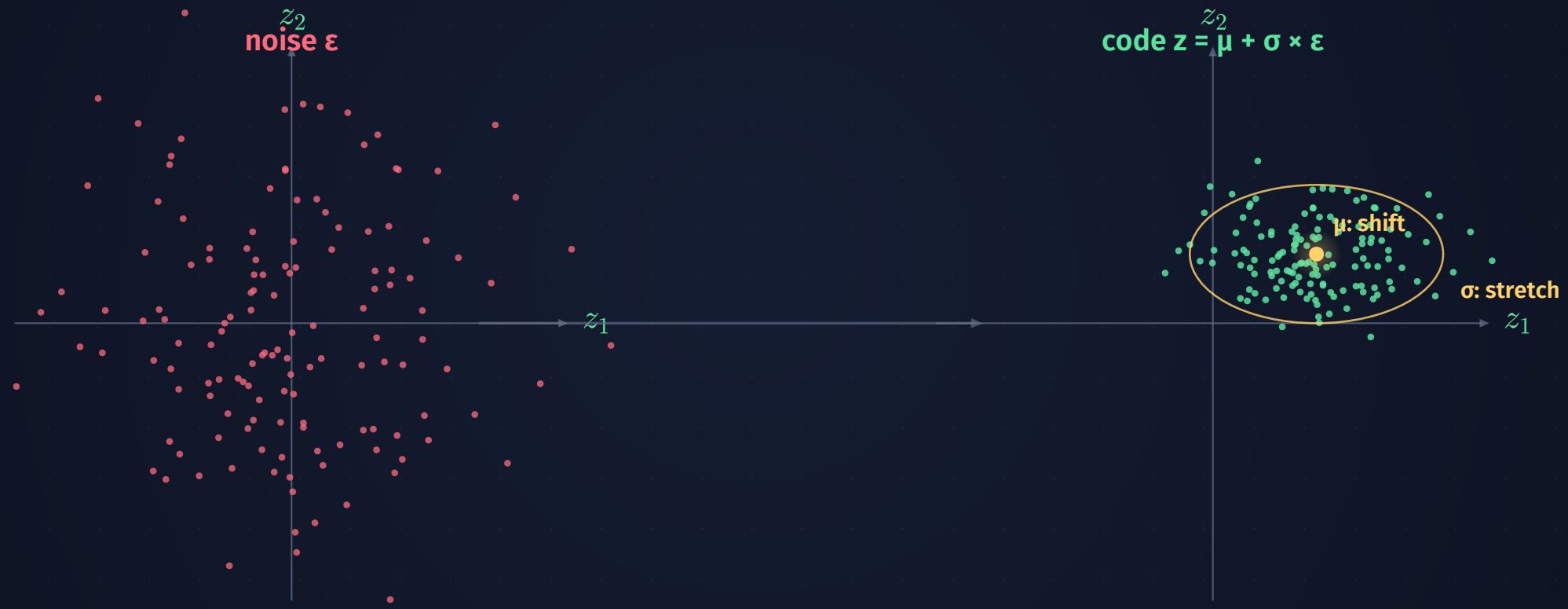
7 The reparameterization trick

Backpropagating through randomness

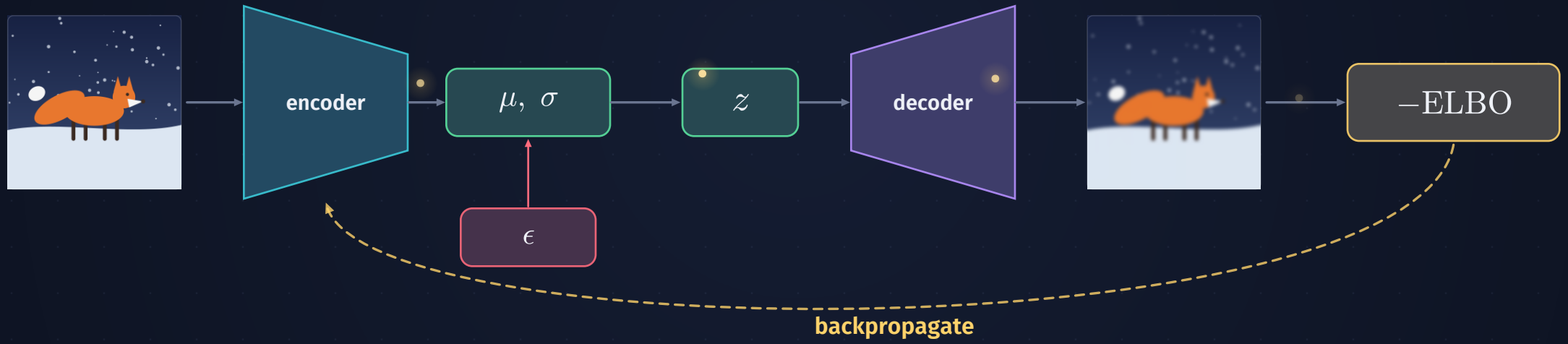

gradients
x
encoder
μ
σ
$z \sim \mathcal{N}(\mu, \sigma^2)$
decoder
$\hat{x}$
$\mathcal{L}$
no gradient through a random draw



$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I)$$


z_2
noise ε
z_1
z_2
code z = μ + σ × ε
μ: shift
σ: stretch
z_1

7 The reparameterization trick

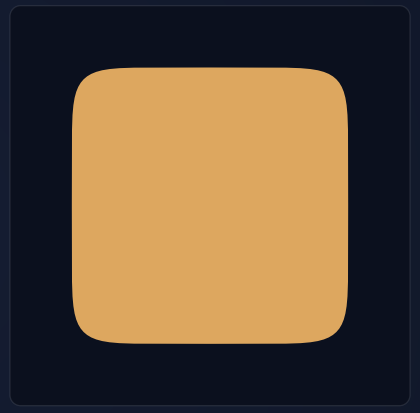


8 VAEs in practice

Exploring the latent space


z_2
z_1





decoded image



$$z \sim \mathcal{N}(0, I) \quad \rightarrow \quad \text{decoder} \quad \rightarrow \quad x$$



one network call per image



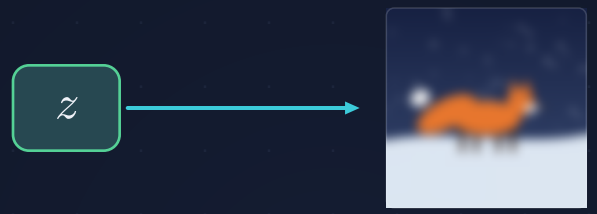
original



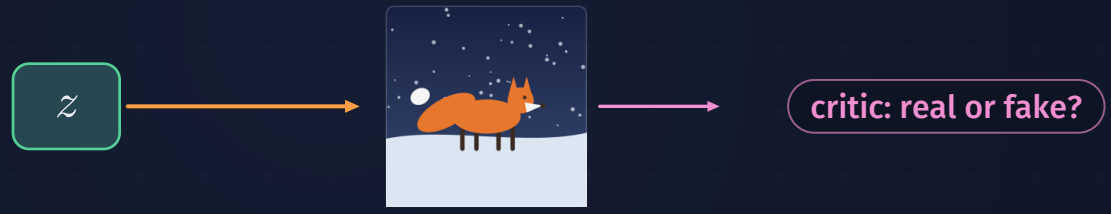
typical VAE output (illustrative)

fine detail is averaged away

VAE: one jump, blurry



GAN: one jump, judged by a critic



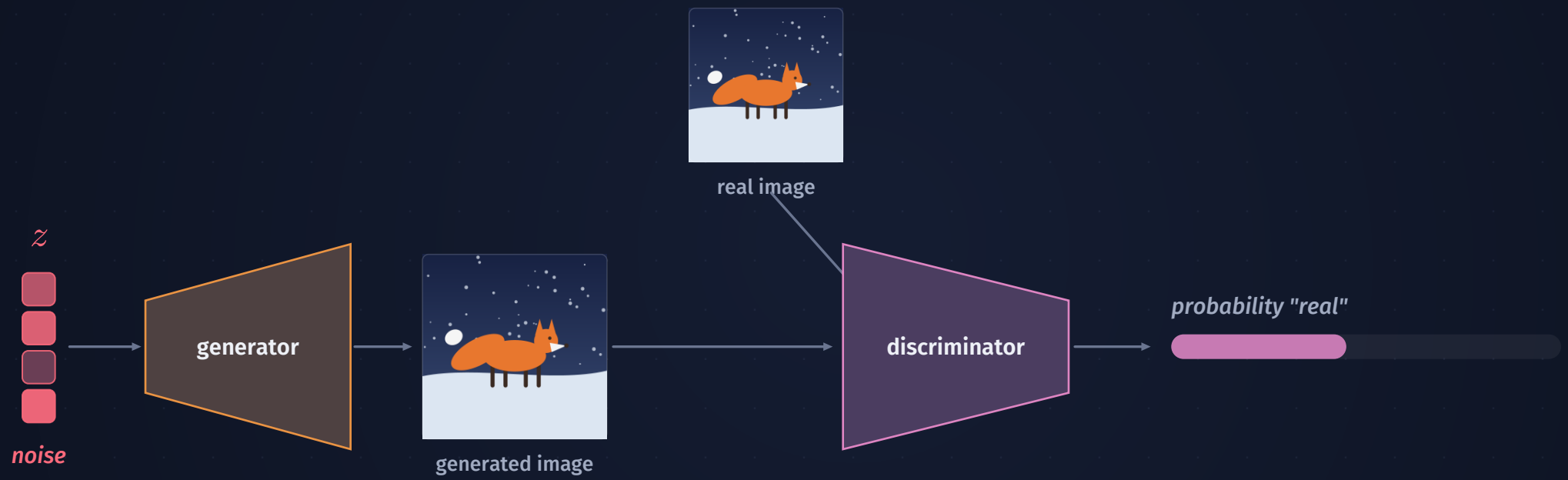
Diffusion: many small steps



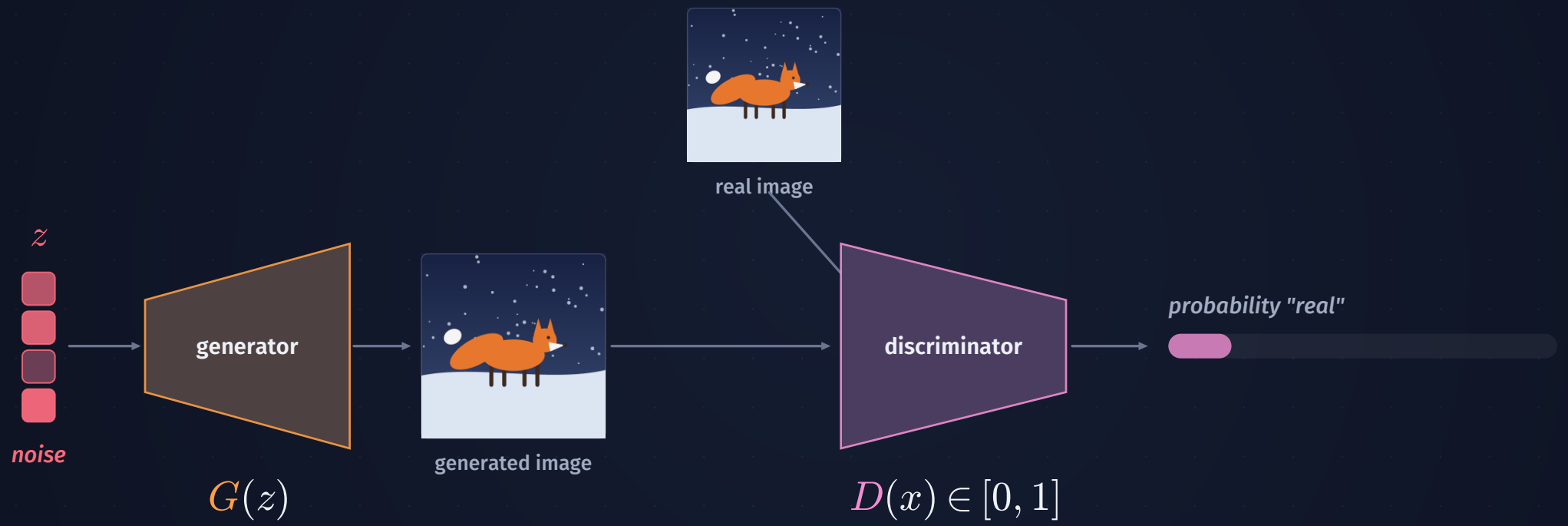
9 Generative adversarial networks

A generator versus a discriminator

Goodfellow et al., 2014



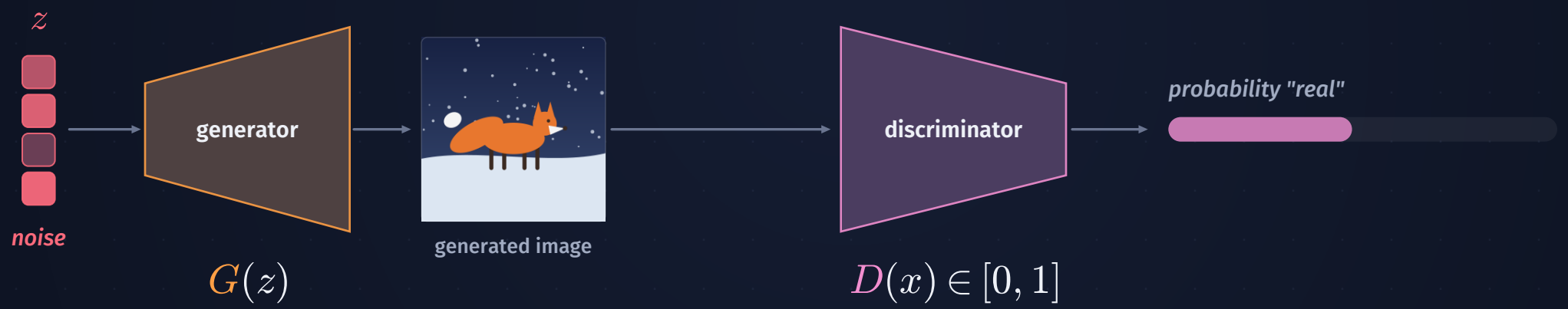
Goodfellow et al., 2014



$$\min_G \max_D \; \mathbb{E}_{x \sim p_{\text{data}}} [\log D(x)] + \mathbb{E}_{z \sim p(z)} [\log (1 - D(G(z)))]$$

D: score real images high

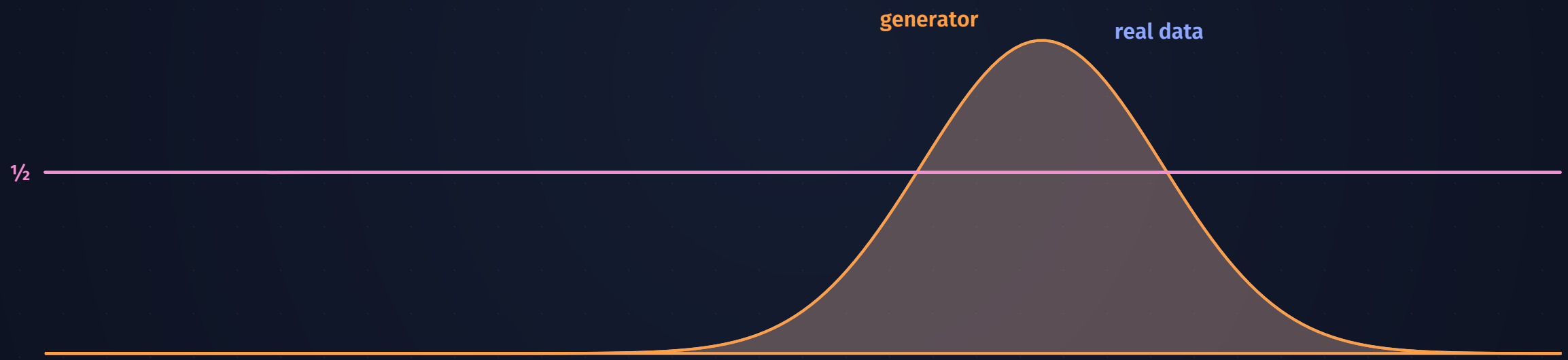
D: score fakes low; G: make them look real



$$D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}$$

training progress 100%

best discriminator D(x)





GAN-style sample: sharp

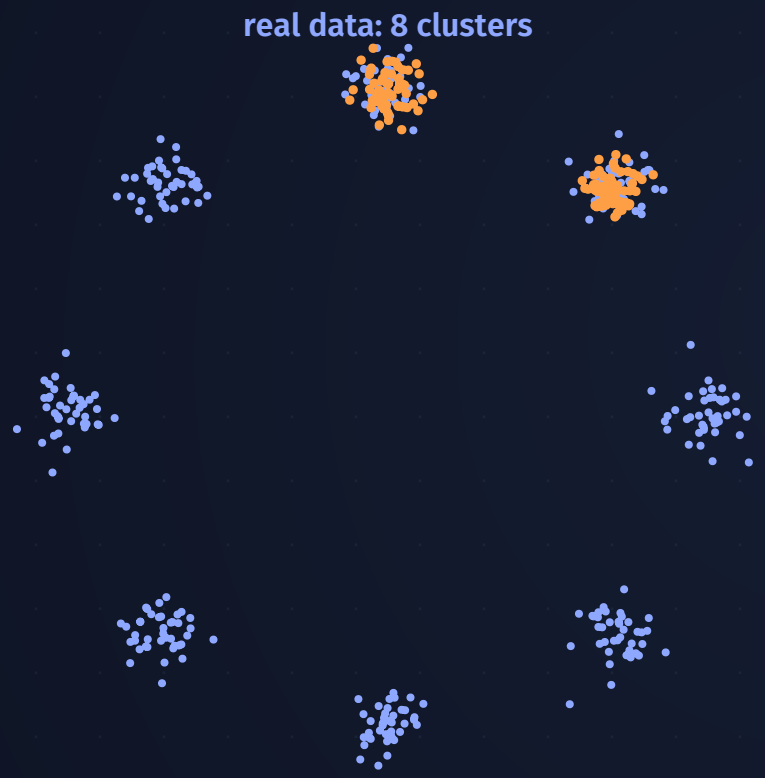
illustrative



VAE-style sample: blurry

one network call per image

StyleGAN: realistic faces (Karras et al., 2019)



Mode collapse: whole kinds of images go missing

Unstable training: the two networks can oscillate

→ **Diffusion:** stable training, covers every mode, but needs many steps

10 Diffusion: the big idea

Destroy slowly, then learn to undo

$$q(x_t \mid x_{t-1})$$

forward: add noise (fixed)



t = 0 t = 50 t = 150 t = 300 t = 500 t = 1000

$$q(x_t \mid x_{t-1})$$

forward: add noise (fixed)



reverse: remove noise (learned)

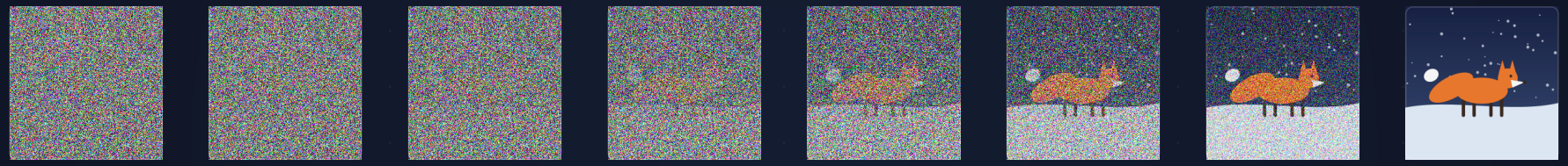
$$p_\theta(x_{t-1} \mid x_t)$$



$$t = 0$$

reverse: denoising

exact reverse process for this toy data



many small, easy steps

stable training, sharp and diverse samples

11 The forward process

Adding noise, with math

$$q(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t;\ \sqrt{1-\beta_t}\, x_{t-1},\ \beta_t I\right)$$

$$x_t = \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I)$$



step t − 1 noise ε step t

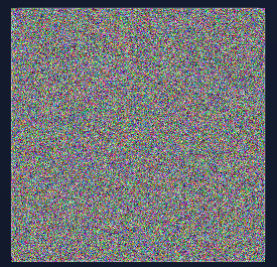
$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon$$

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



clean image

× 0.70 +



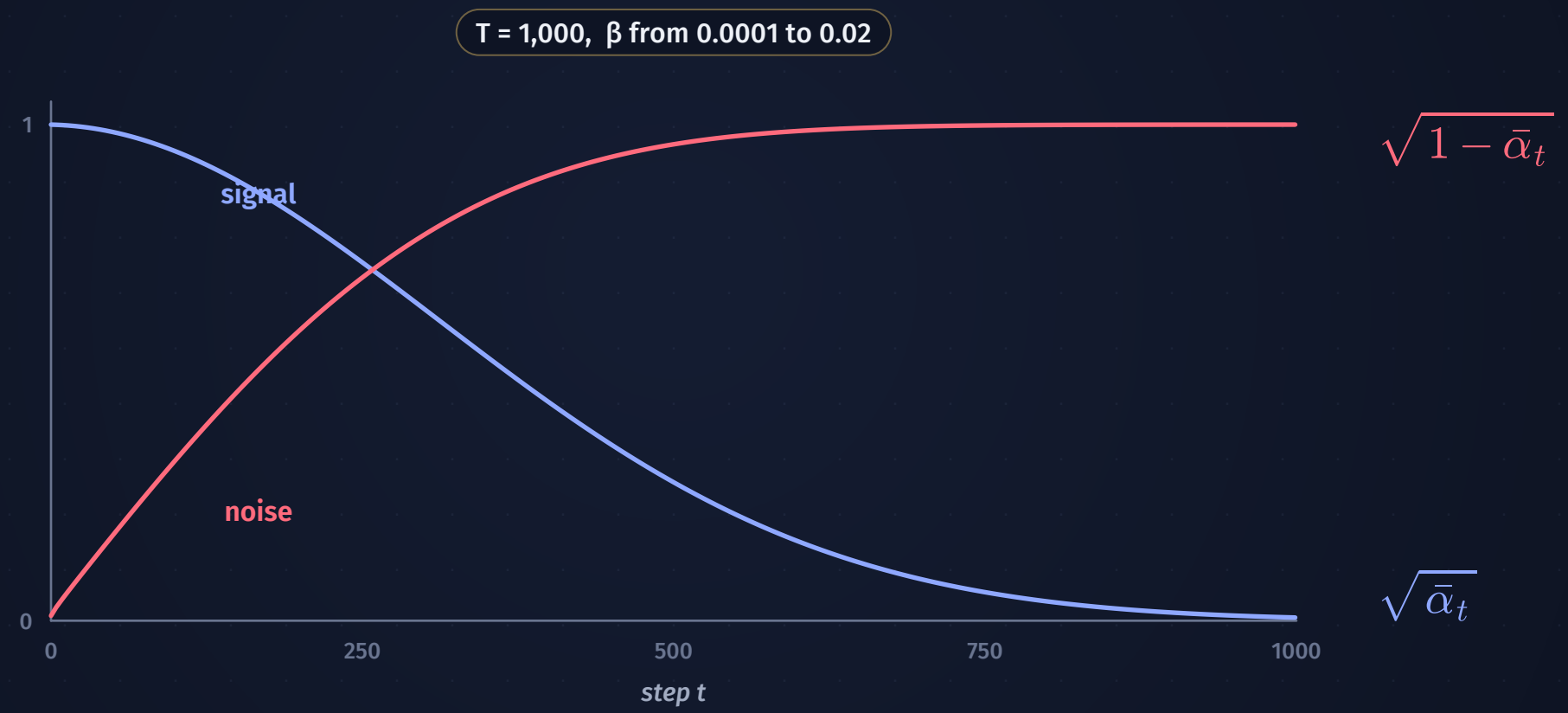
noise

× 0.71 →



noisy image, t ≈ 263

one shot, no loop over steps


T = 1,000, β from 0.0001 to 0.02
1
0
signal
noise
$\sqrt{1-\bar{\alpha}_t}$
$\sqrt{\bar{\alpha}_t}$
0
250
500
750
1000
step t



t = 95

signal weight 0.95

noise weight 0.31

variance: 0.91 + 0.09 = 1



t = 247

signal weight 0.73

noise weight 0.68

variance: 0.53 + 0.47 = 1

no learnable parameters in q

12 Training a DDPM

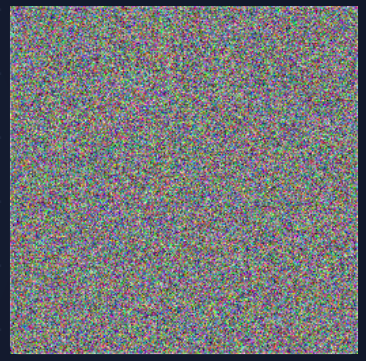
Learning to predict the noise



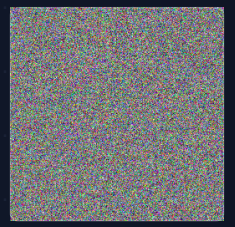
noisy image, t = 400

$$\epsilon_\theta(x_t,\ t)$$

neural network



predicted noise

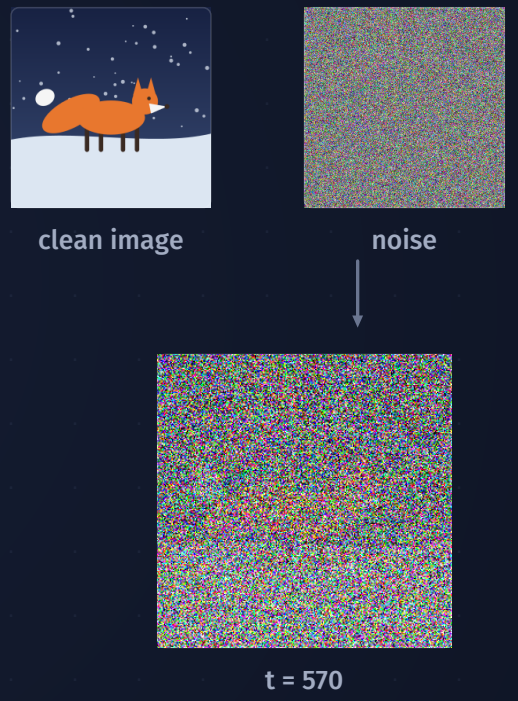


true noise ε

Training loop

1. $x_0 \sim \text{data}$
2. $t \sim \text{Uniform}\{1, \ldots, T\}$
3. $\epsilon \sim \mathcal{N}(0, I)$
4. $x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon$
5. gradient step on $\| \epsilon - \epsilon_\theta(x_t, t) \|^2$

repeat



Training loop

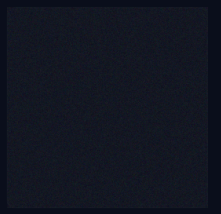
1. $x_0 \sim \text{data}$
2. $t \sim \text{Uniform}\{1, \dots, T\}$
3. $\epsilon \sim \mathcal{N}(0, I)$
4. $x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon$
5. gradient step on $\| \epsilon - \epsilon_\theta(x_t, t) \|^2$

repeat

$$\mathcal{L}_{\text{simple}}(\theta) = \mathbb{E}_{x_0,\, t,\, \epsilon}\left[\| \epsilon - \epsilon_\theta\left(\sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon,\ t\right) \|^2 \right]$$

mean squared error between true and predicted noise

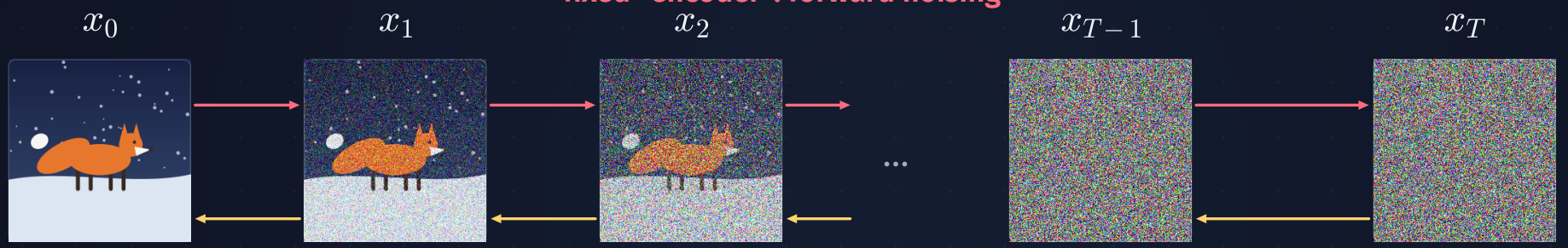






t = 360

fixed "encoder": forward noising

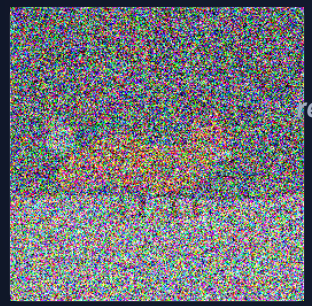


learned "decoder": reverse denoising

$$-\text{ELBO} = \sum_t w_t \, \mathbb{E}[\| \epsilon - \epsilon_\theta(x_t, t) \|^2] + \text{const}$$

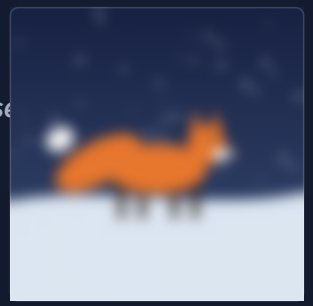
DDPM simply sets every weight to 1

$$\hat{x}_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\ \epsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}}$$



noisy image

remove predicted noise



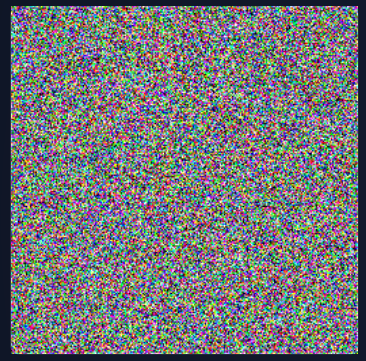
estimated clean image (illustrative)

blurry at high noise, sharp at low noise

13 Sampling with DDPM

From noise to data, step by step

13 Sampling with DDPM



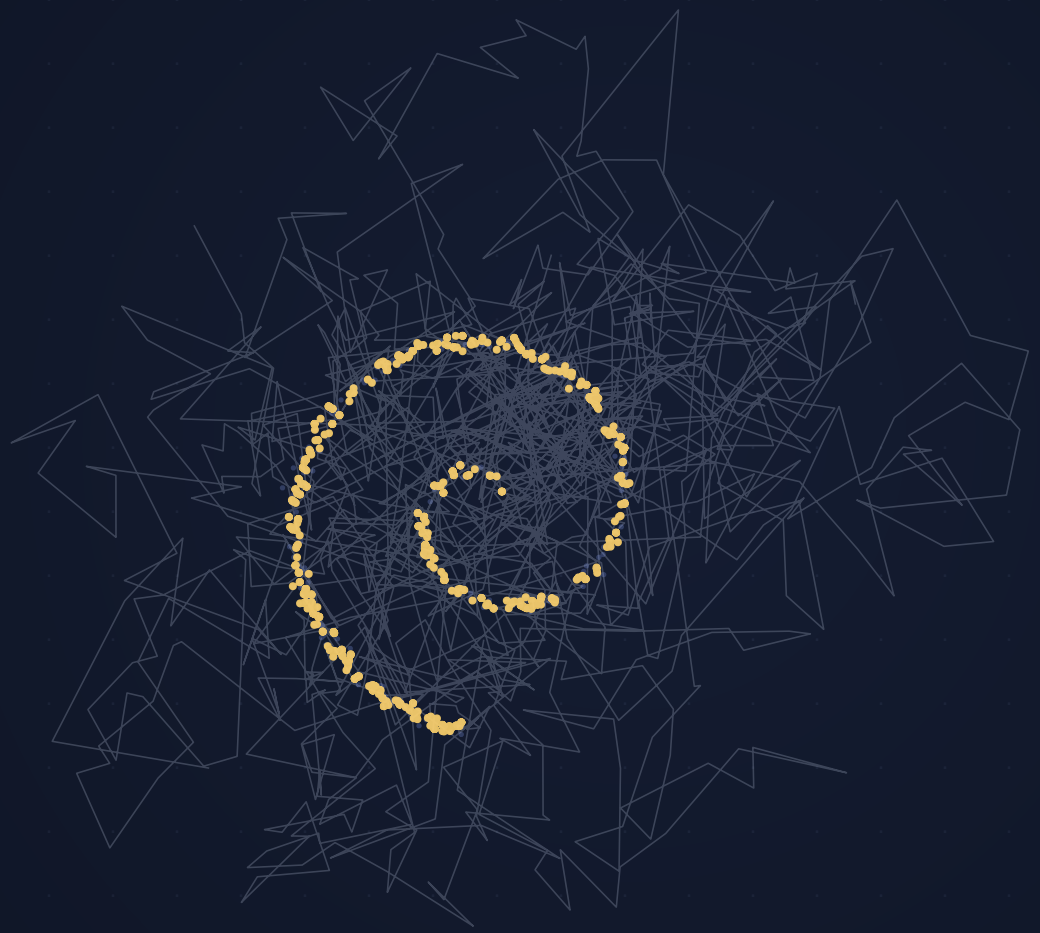
x_T: pure noise



x_0: a sample

$$x_{t-1} = \underbrace{\frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\epsilon_\theta(x_t, t)\right)}_{\text{remove the predicted noise, rescale}} + \underbrace{\sigma_t\, z}_{\text{fresh noise}}$$

$$z \sim \mathcal{N}(0, I), \qquad \sigma_t^2 = \beta_t$$



$$t = 0$$

exact noise predictor for this toy data



coarse first, details last (illustrative)

1,000

network calls for one image

14 DDIM: faster sampling

Deterministic jumps

Song, Meng and Ermon, 2021

same trained network

$$\epsilon_\theta(x_t, t)$$

new sampler

- skips steps
- no fresh noise
- deterministic

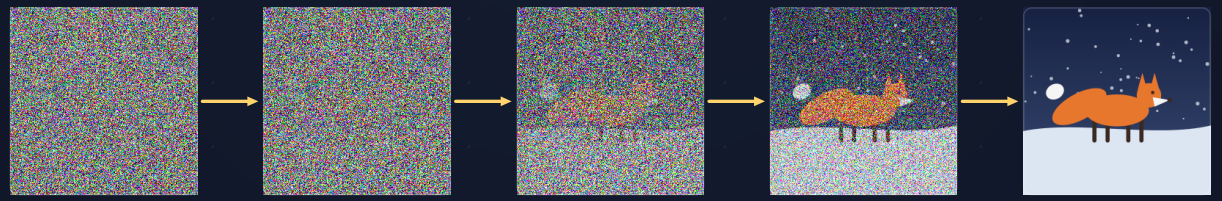
Song, Meng and Ermon, 2021

$$\hat{x}_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\ \epsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}}$$

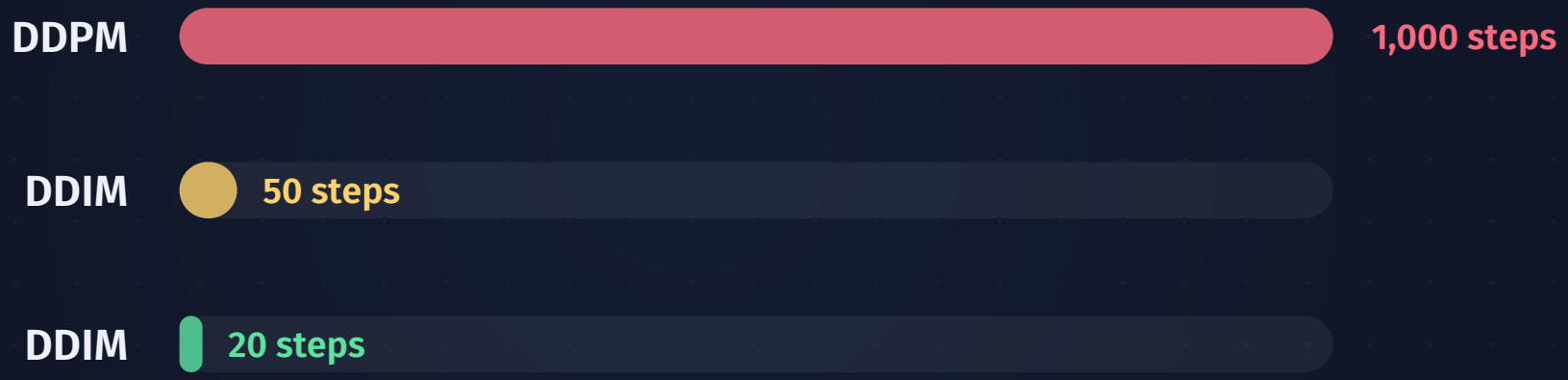
1. predict the clean image

$$x_{t'} = \sqrt{\bar{\alpha}_{t'}}\ \hat{x}_0 + \sqrt{1-\bar{\alpha}_{t'}}\ \epsilon_\theta(x_t, t), \qquad t' < t$$

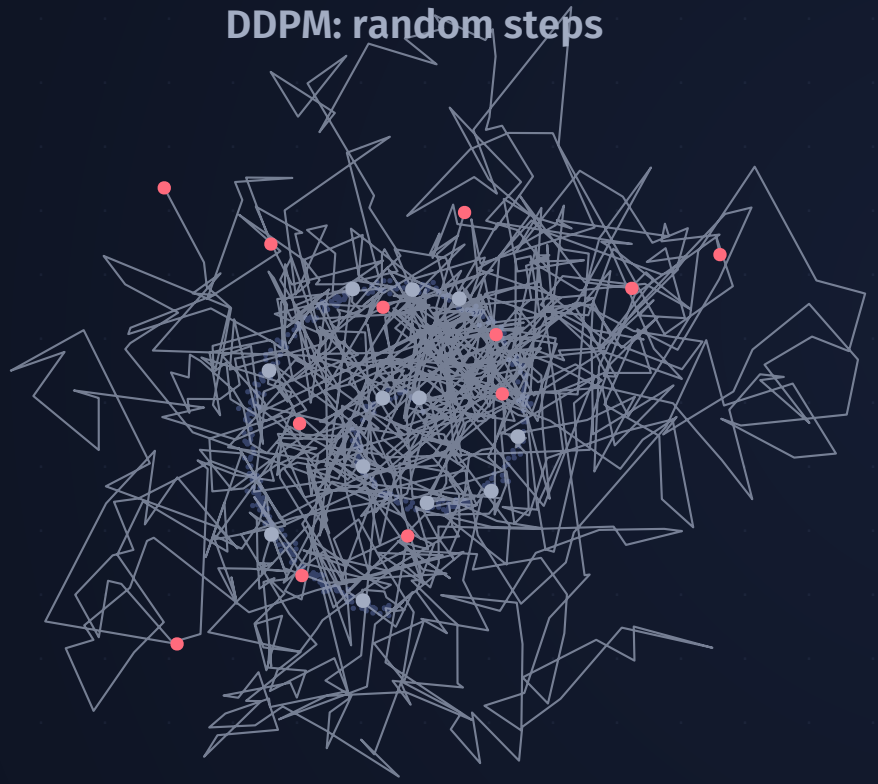
2. jump to a lower noise level, reusing the predicted noise



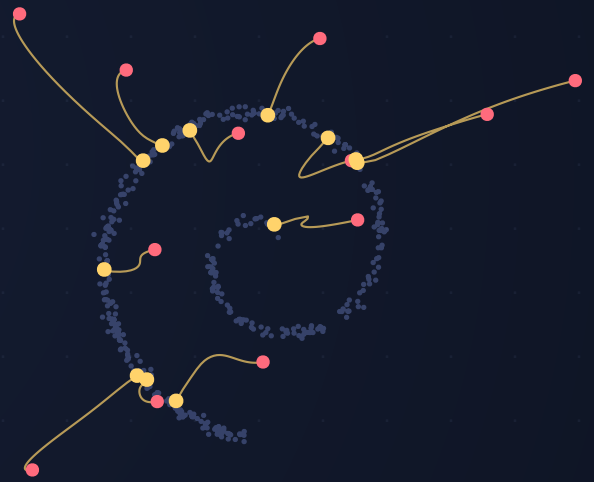
DDIM: faster sampling



DDPM: random steps

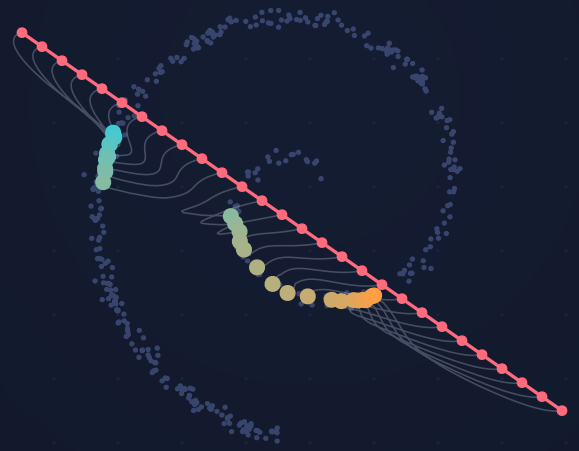


DDIM: deterministic



same starting points for both

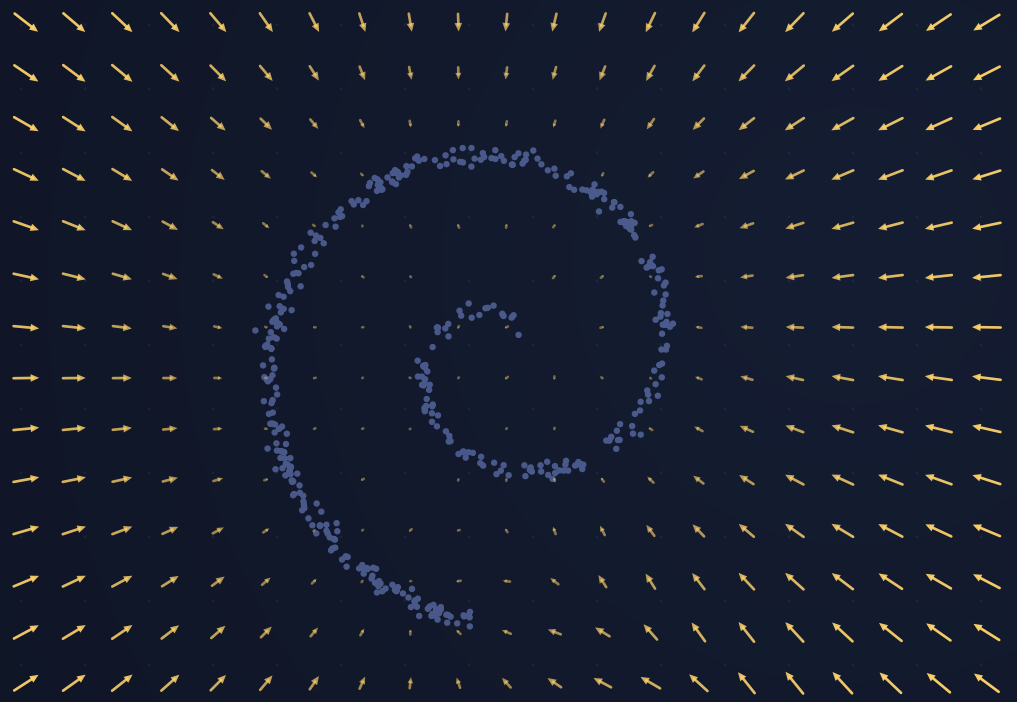
line of starting noises



samples move smoothly along the data

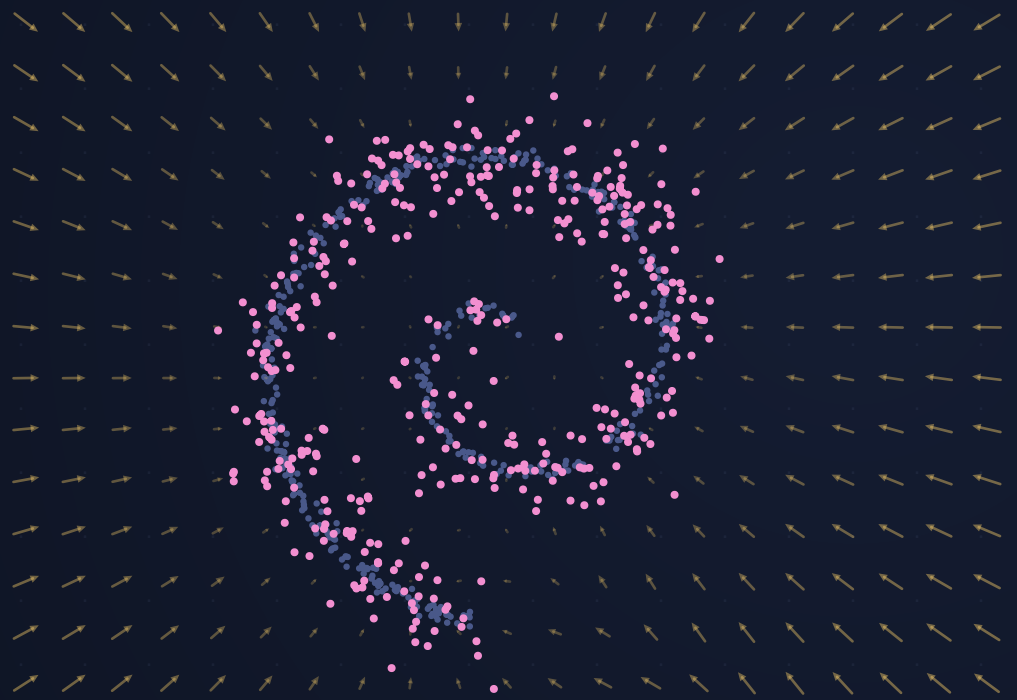
15 Score-based models

Following the gradient of log-density



$$s(x) = \nabla_x \log p(x)$$

points toward higher probability



$$x \leftarrow x + \eta \nabla_x \log p(x) + \sqrt{2\eta}\, z$$

step along the score, plus a little noise

Song and Ermon, 2019



noise level σ = 0.03

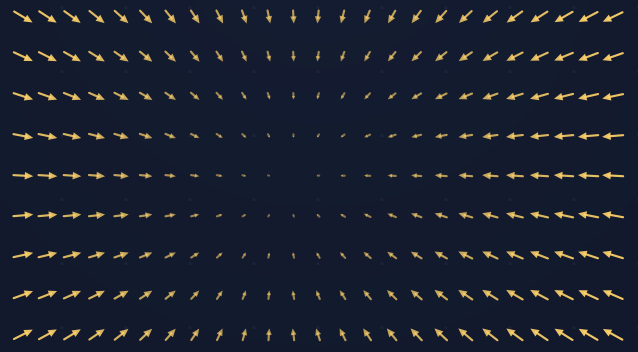
small σ: sharp field, fine detail

annealed Langevin dynamics

$$\nabla_{x_t} \log p_t(x_t) \approx -\frac{\epsilon_\theta(x_t, t)}{\sqrt{1-\bar{\alpha}_t}}$$

score model

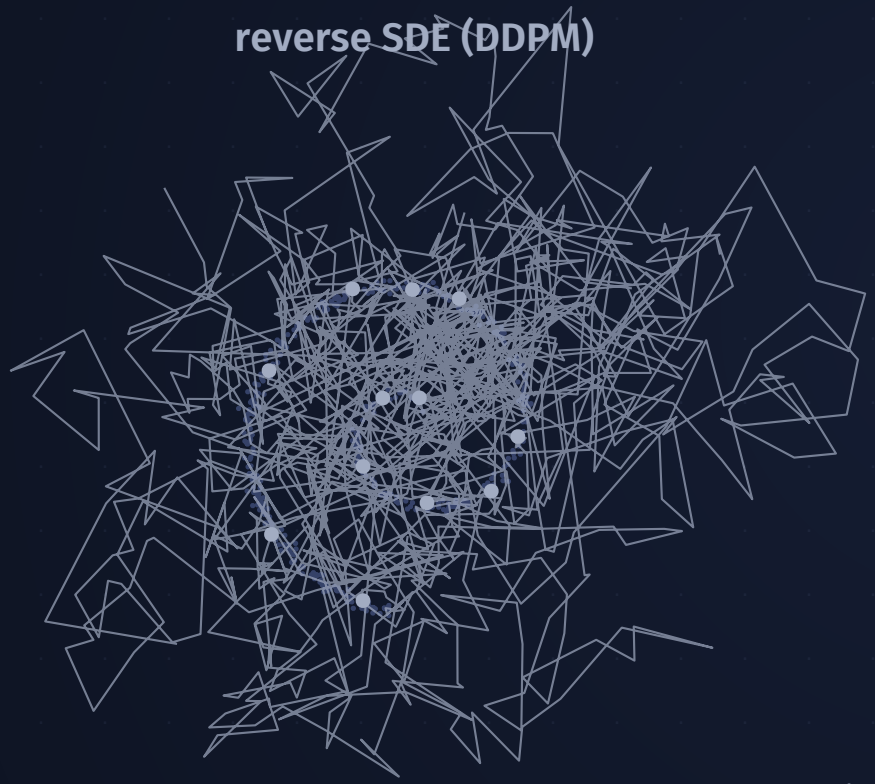
noise predictor, rescaled



the same arrows either way

Song et al., 2021

reverse SDE (DDPM)



probability flow ODE (DDIM)



same marginal distributions, different paths

16 Flow matching

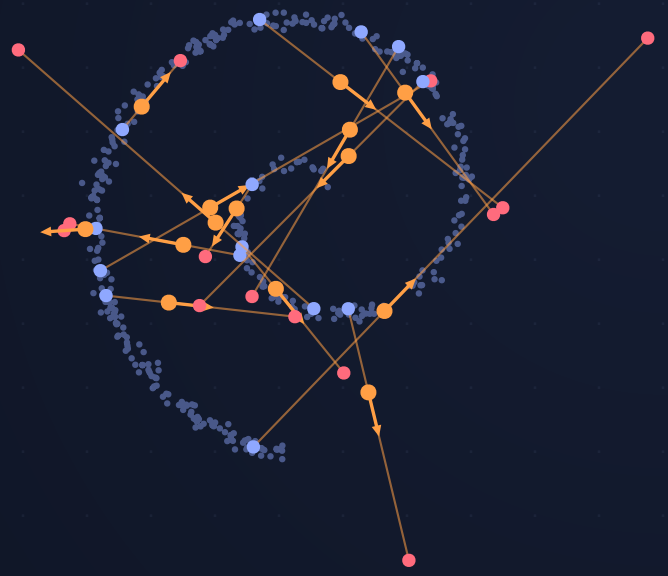
Straight paths from noise to data



$$x_t = (1 - t)\, x_0 + t\, \epsilon$$

t = 0: data t = 1: noise

t = 0.49



$$x_t = (1 - t)\, x_0 + t\, \epsilon$$

t = 0: data t = 1: noise

t = 0.33

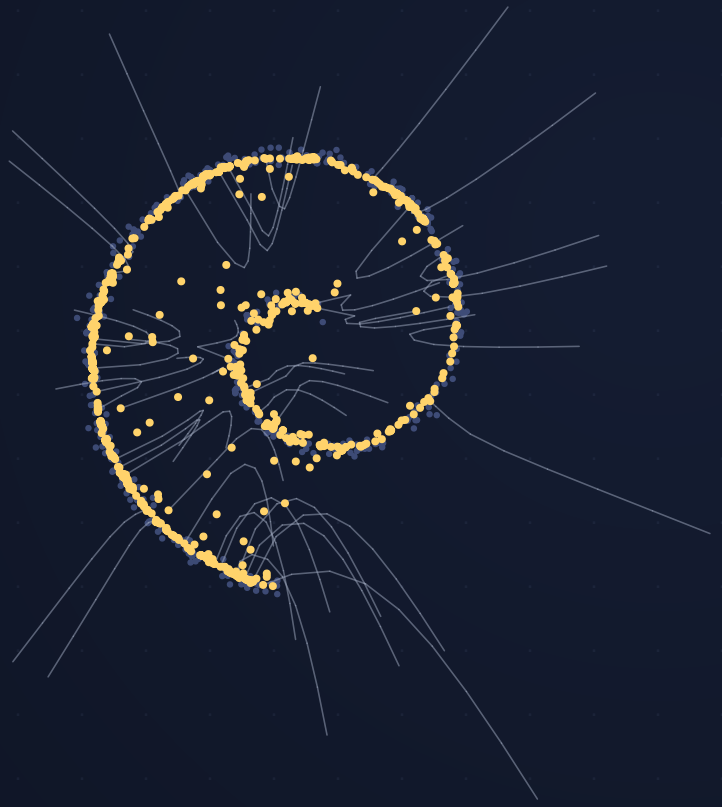
$$\frac{dx_t}{dt} = \epsilon - x_0$$

$$\mathcal{L}(\theta) = \mathbb{E}_{t,\, x_0,\, \epsilon}[\| v_\theta(x_t, t) - (\epsilon - x_0) \|^2]$$



$$t = 0.03$$

arrows: the direction of generation

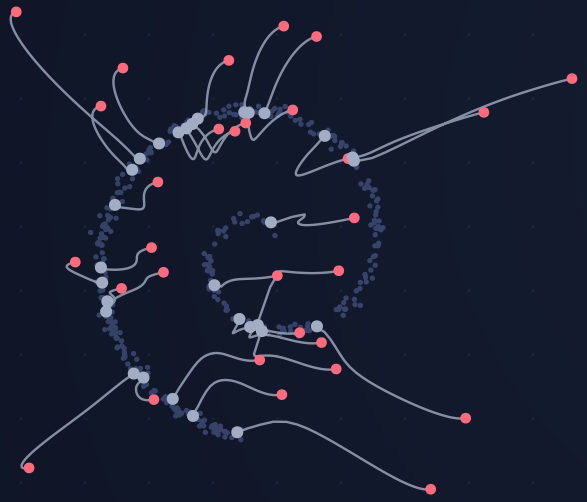


$$x_{t-\Delta t} = x_t - \Delta t \; v_\theta(x_t, t)$$

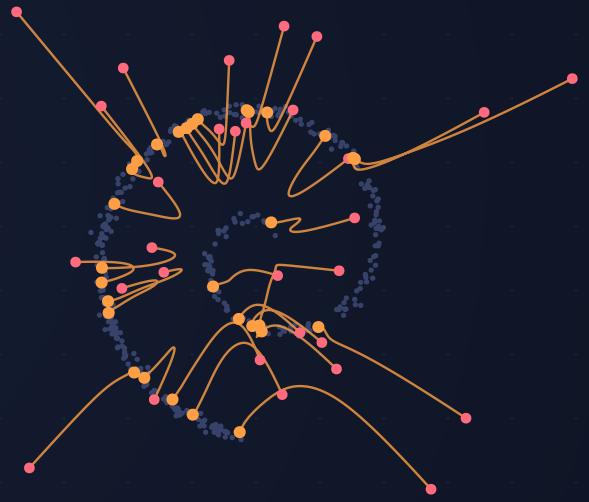
Euler step 8 of 8

Lipman et al., 2023; Liu et al., 2023

diffusion ODE: curved paths



flow matching: nearly straight



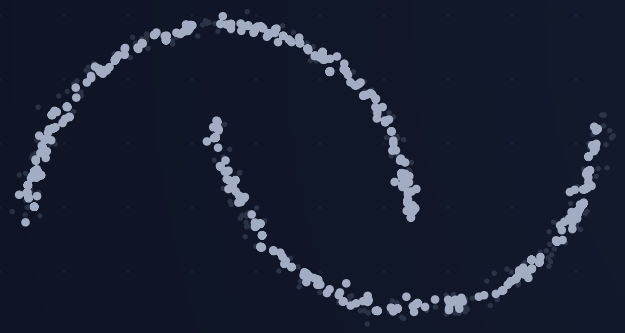
rectified flow: Stable Diffusion 3, Flux

17 Conditioning and guidance

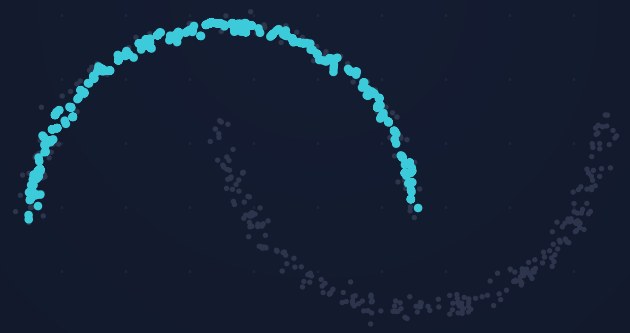
Steering what gets generated

$$\epsilon_\theta(x_t,\ t,\ c)$$

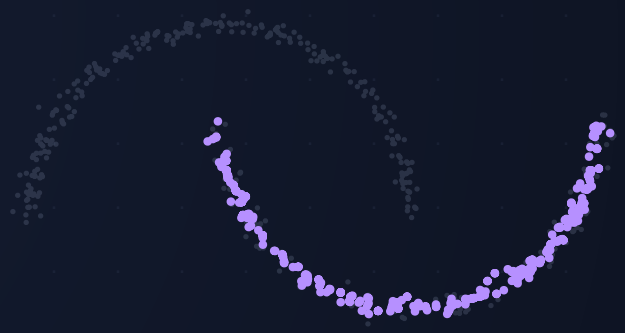
no condition

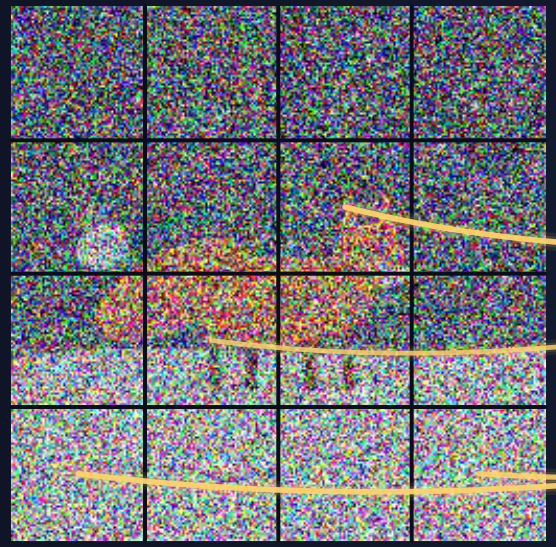


c = class 0



c = class 1





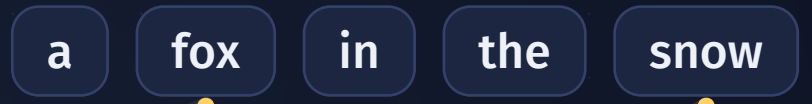


image features: queries

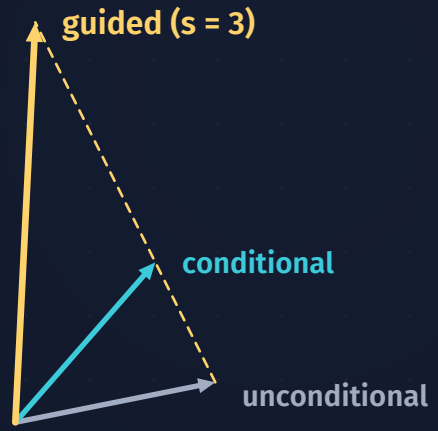
prompt words: keys and values

Ho and Salimans, 2022

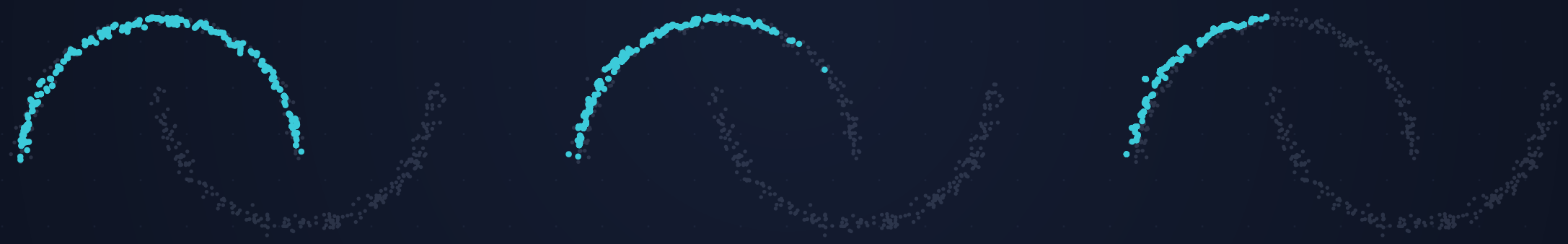
$$\tilde{\epsilon} = \epsilon_\theta(x_t, t, \emptyset) + s \cdot (\epsilon_\theta(x_t, t, c) - \epsilon_\theta(x_t, t, \emptyset))$$

without the prompt

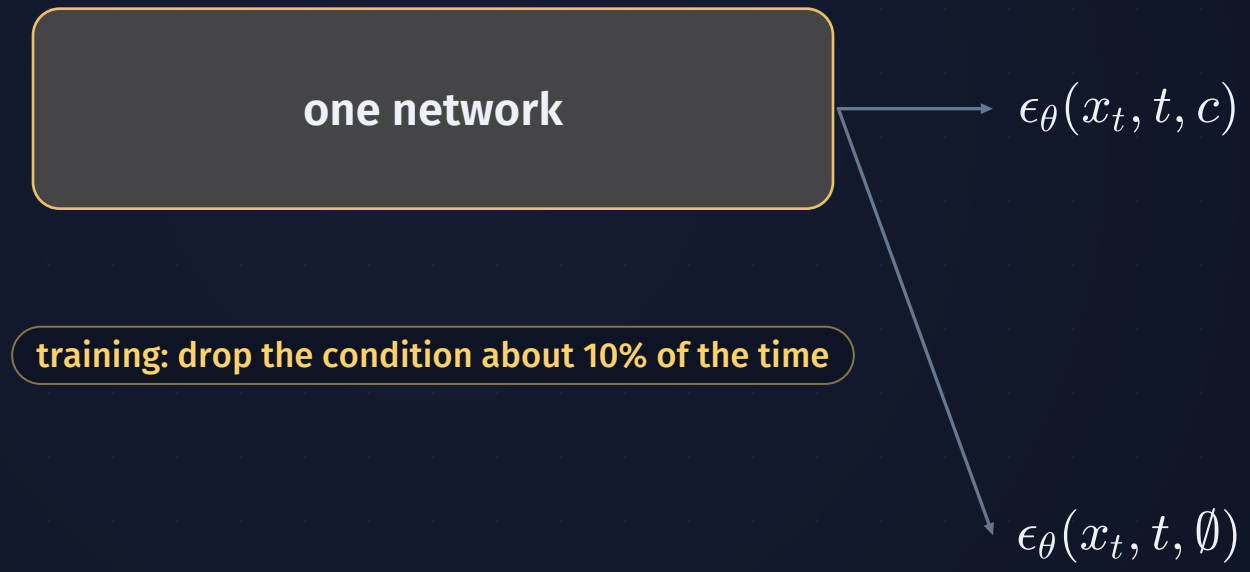
the direction the prompt adds



$s = 1$ $s = 3$ $s = 7.5$



samples for class 0 at three guidance scales

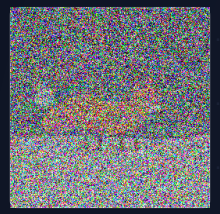

one network
$\epsilon_\theta(x_t, t, c)$
training: drop the condition about 10% of the time
$\epsilon_\theta(x_t, t, \emptyset)$

18 The U-Net

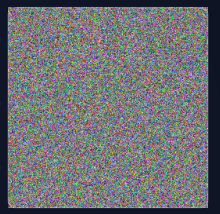
The classic denoising network

The U-Net

Ronneberger, Fischer and Brox, 2015



noisy image

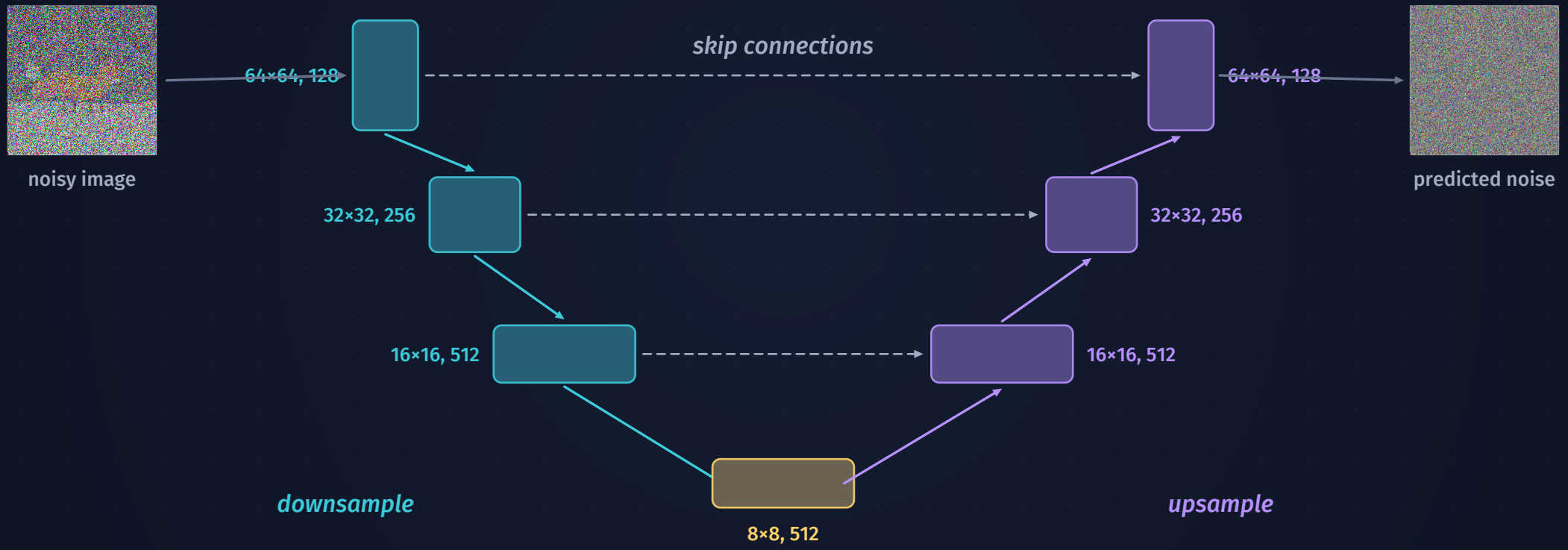


predicted noise

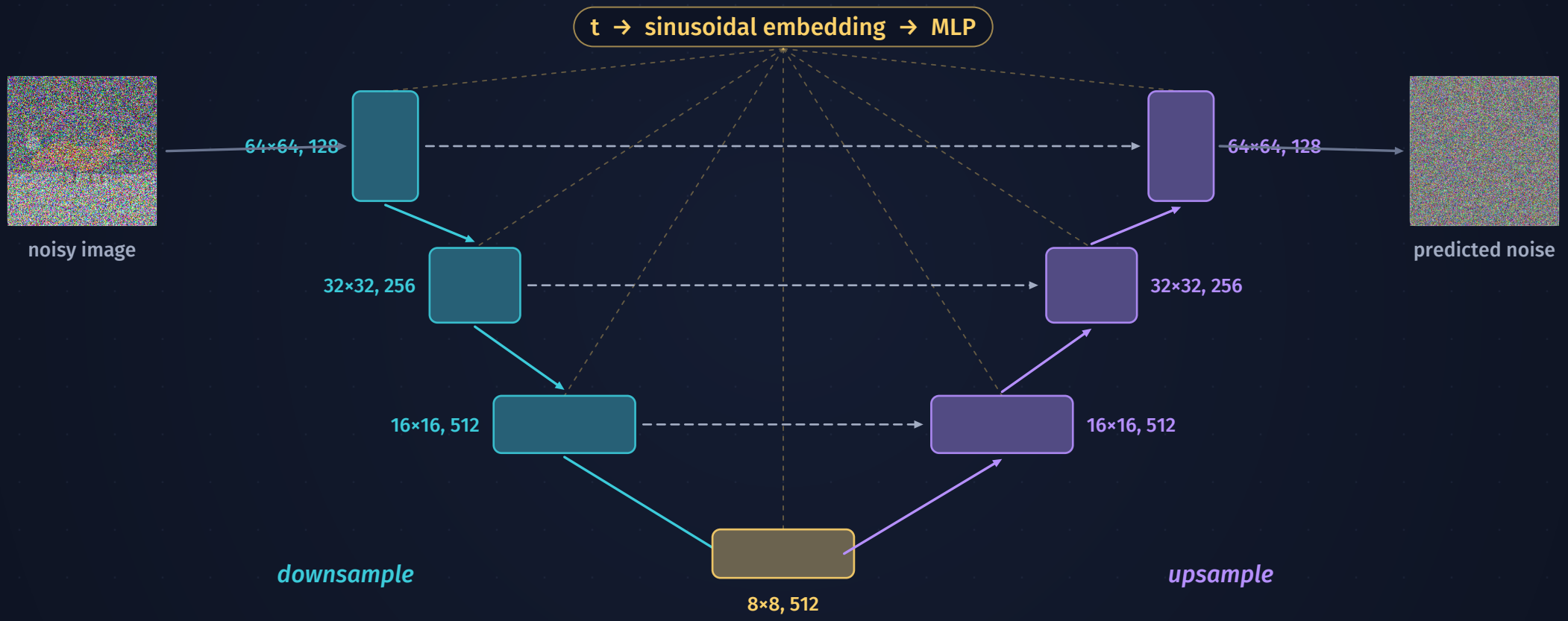
Ronneberger, Fischer and Brox, 2015



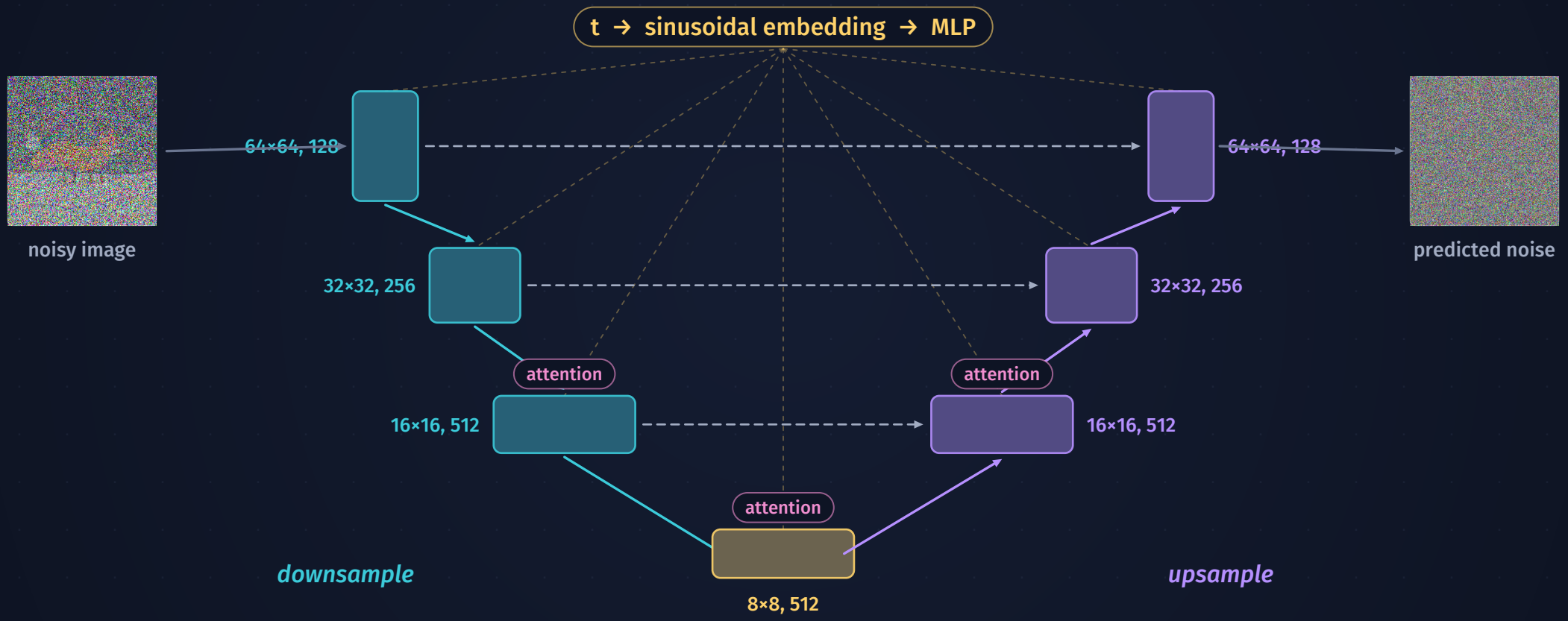
Ronneberger, Fischer and Brox, 2015



Ronneberger, Fischer and Brox, 2015



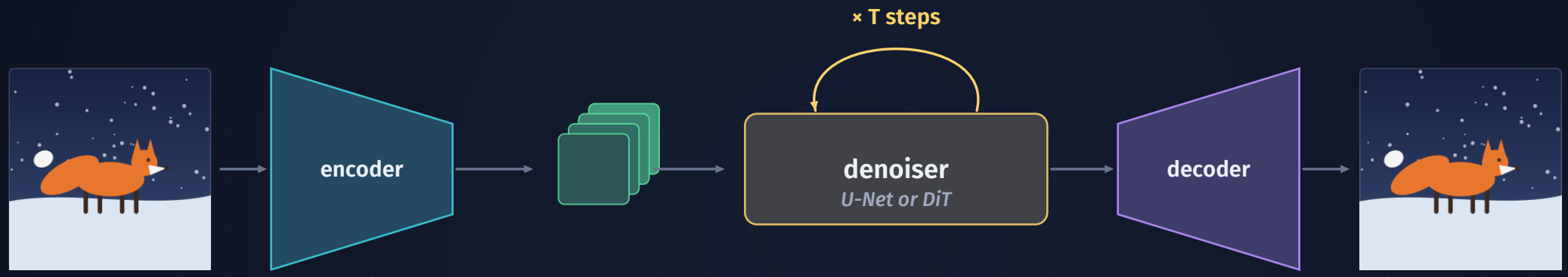
Ronneberger, Fischer and Brox, 2015



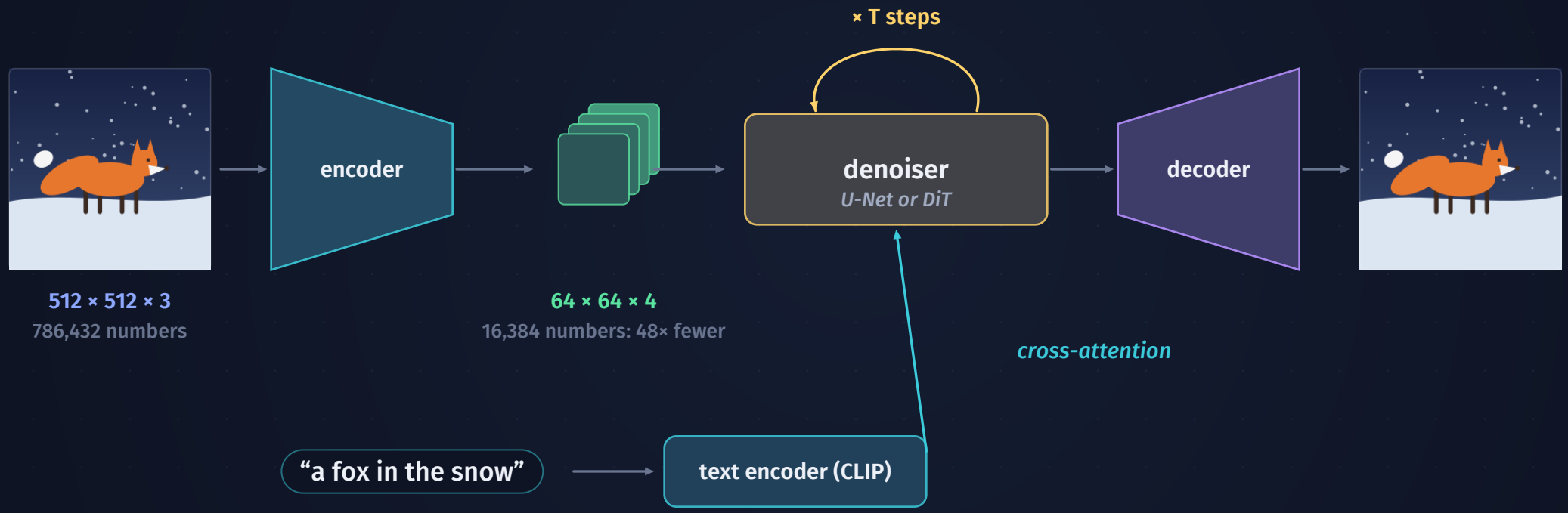
self-attention and cross-attention to the prompt

19 Latent diffusion and DiT

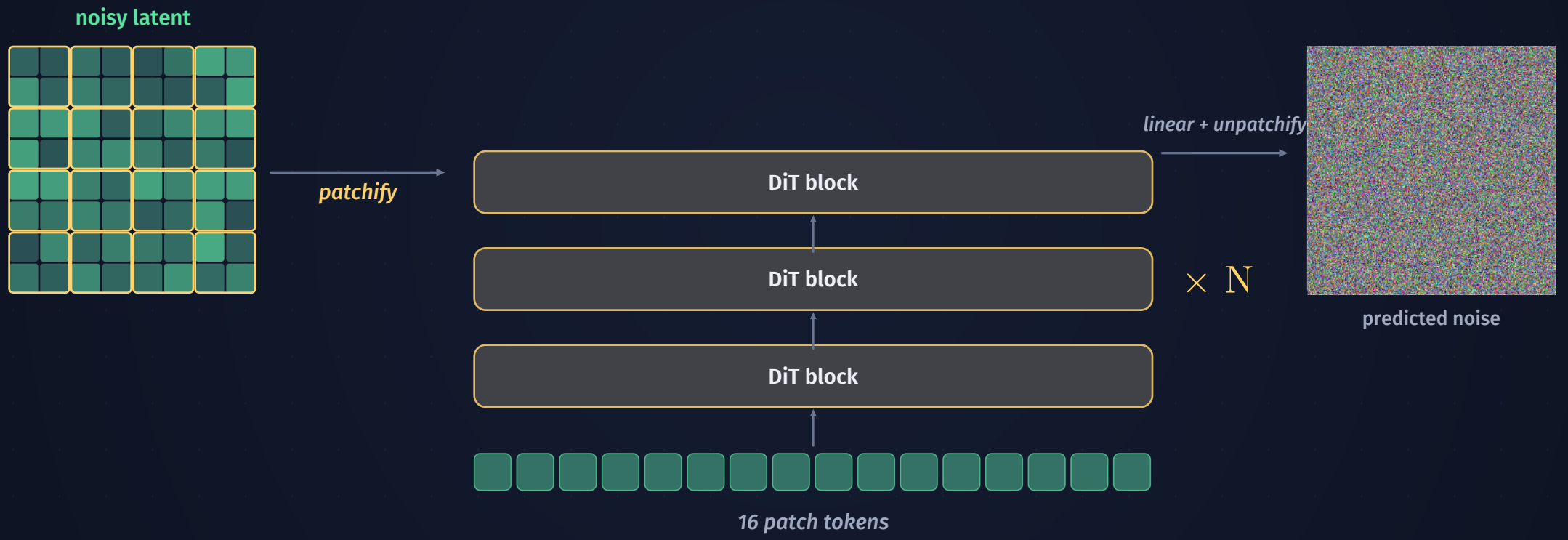
Smaller spaces, bigger networks


× T steps
encoder
denoiser
U-Net or DiT
decoder
512 × 512 × 3
786,432 numbers
64 × 64 × 4
16,384 numbers: 48× fewer

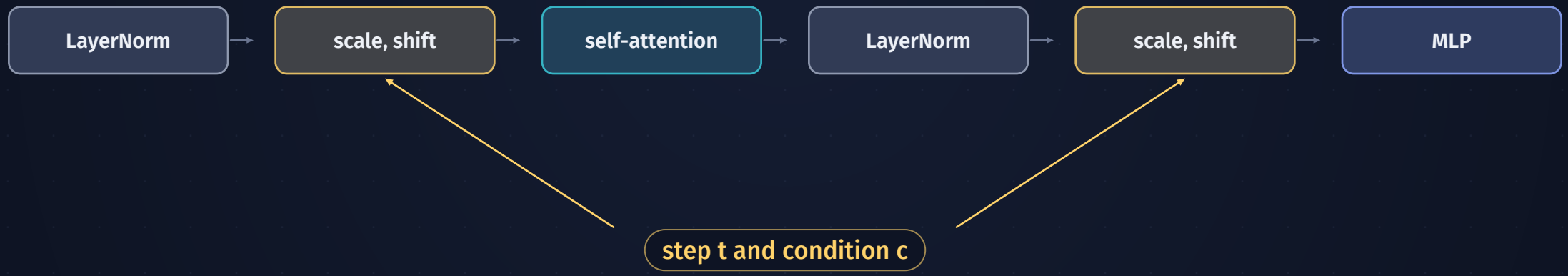
Rombach et al., 2022

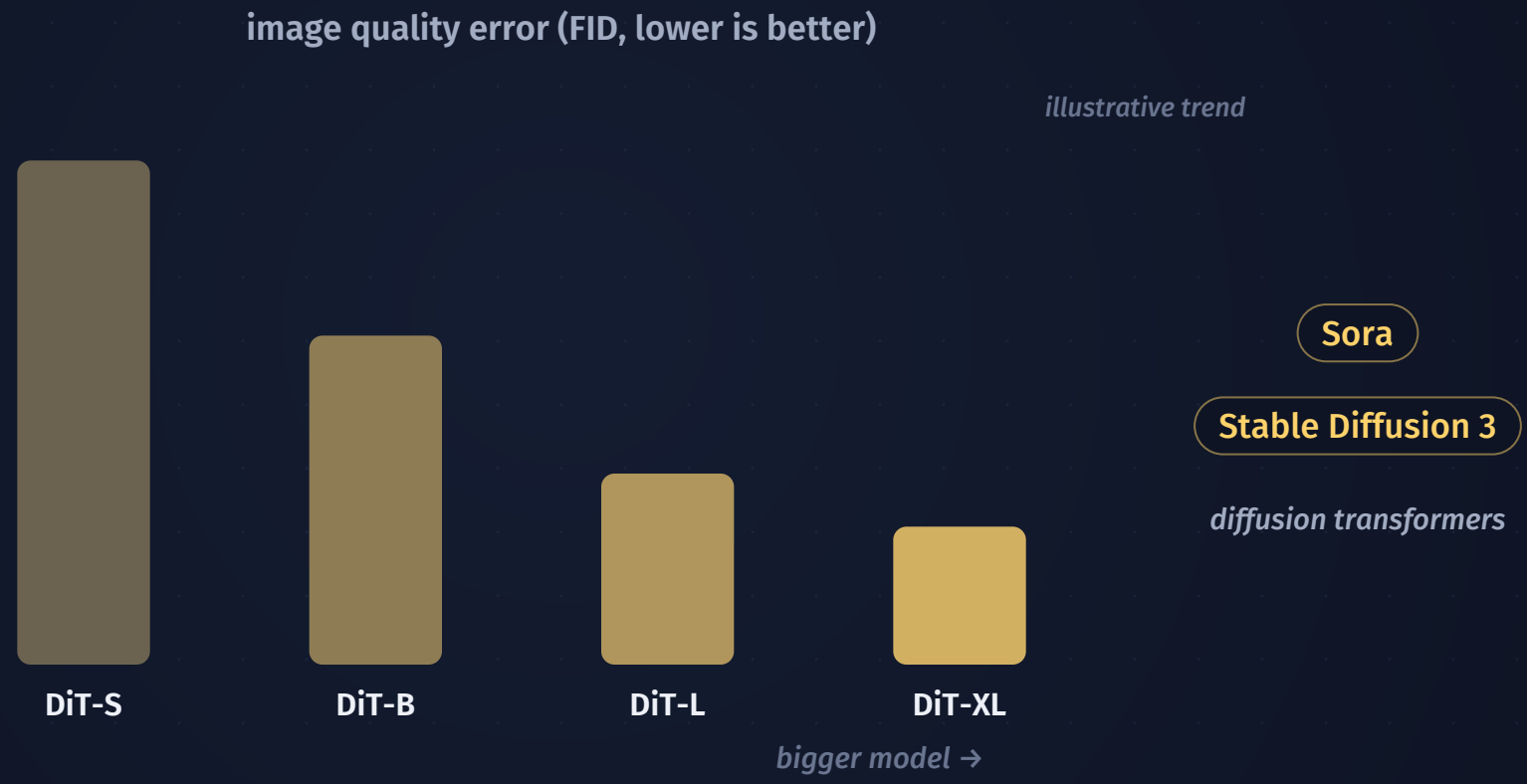


Peebles and Xie, 2023



$$h \leftarrow a(t, c) \odot \mathrm{LayerNorm}(h) + b(t, c)$$




image quality error (FID, lower is better)
illustrative trend
DiT-S
DiT-B
DiT-L
DiT-XL
bigger model →
Sora
Stable Diffusion 3
diffusion transformers

20 Recap

Everything in one place

Likelihood and KL — Fit the model by maximizing log-likelihood: the same as minimizing KL from data to model.

Likelihood and KL	Fit the model by maximizing log-likelihood: the same as minimizing KL from data to model.
VAEs and GANs	VAEs maximize the ELBO = reconstruction − KL, with $z = \mu + \sigma\varepsilon$; GANs train a generator against a discriminator.

Likelihood and KL	Fit the model by maximizing log-likelihood: the same as minimizing KL from data to model.
VAEs and GANs	VAEs maximize the ELBO = reconstruction − KL, with $z = \mu + \sigma\varepsilon$; GANs train a generator against a discriminator.
Diffusion	Fixed noising, learned denoising by predicting the noise; DDPM is stochastic, DDIM deterministic and faster.

Likelihood and KL	Fit the model by maximizing log-likelihood: the same as minimizing KL from data to model.
VAEs and GANs	VAEs maximize the ELBO = reconstruction − KL, with $z = \mu + \sigma\varepsilon$; GANs train a generator against a discriminator.
Diffusion	Fixed noising, learned denoising by predicting the noise; DDPM is stochastic, DDIM deterministic and faster.
Scores and flows	The noise predictor is a rescaled score; flow matching regresses straight-line velocities.

Likelihood and KL	Fit the model by maximizing log-likelihood: the same as minimizing KL from data to model.
VAEs and GANs	VAEs maximize the ELBO = reconstruction − KL, with $z = \mu + \sigma\varepsilon$; GANs train a generator against a discriminator.
Diffusion	Fixed noising, learned denoising by predicting the noise; DDPM is stochastic, DDIM deterministic and faster.
Scores and flows	The noise predictor is a rescaled score; flow matching regresses straight-line velocities.
In practice	Classifier-free guidance, U-Net or DiT denoisers, usually in an autoencoder latent space.

Maximum likelihood

$$\max_{\theta} \sum_{i} \log p_\theta(x_i)$$

ELBO

$$\mathbb{E}_q[\log p_\theta(x \mid z)] - D_{\mathrm{KL}}(q_\phi(z \mid x) \,\|\, p(z))$$

Forward process

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon$$

DDPM loss

$$\| \epsilon - \epsilon_\theta(x_t, t) \|^2$$

Score

$$\nabla_x \log p_t(x) \approx -\epsilon_\theta(x, t) \,/\, \sqrt{1 - \bar{\alpha}_t}$$

Flow matching

$$\| v_\theta(x_t, t) - (\epsilon - x_0) \|^2$$

Thanks for watching

Further reading

Kingma and Welling (2014)
Auto-Encoding Variational Bayes

Goodfellow et al. (2014)
Generative Adversarial Nets

Sohl-Dickstein et al. (2015)
Deep Unsupervised Learning using Nonequilibrium Thermodynamics

Ho, Jain and Abbeel (2020)
Denoising Diffusion Probabilistic Models

Song, Meng and Ermon (2021)
Denoising Diffusion Implicit Models

Song and Ermon (2019)
Generative Modeling by Estimating Gradients of the Data Distribution

Song et al. (2021)
Score-Based Generative Modeling through Stochastic Differential Equations

Lipman et al. (2023)
Flow Matching for Generative Modeling

Liu, Gong and Liu (2023)
Flow Straight and Fast (rectified flow)

Ho and Salimans (2022)
Classifier-Free Diffusion Guidance

Ronneberger, Fischer and Brox (2015)
U-Net

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

Peebles and Xie (2023)
Scalable Diffusion Models with Transformers (DiT)

CSC420 Tutorial 4: VAEs and diffusion models, University of Toronto

Presented by Javad Rajabi, Teaching Assistant

cs.toronto.edu/~lindell/teaching/420