Vision: Generative Modeling - 2

EECE454 Intro. to Machine Learning Systems

Fall 2024

Today

- Generative Adversarial Nets
- Diffusion Models

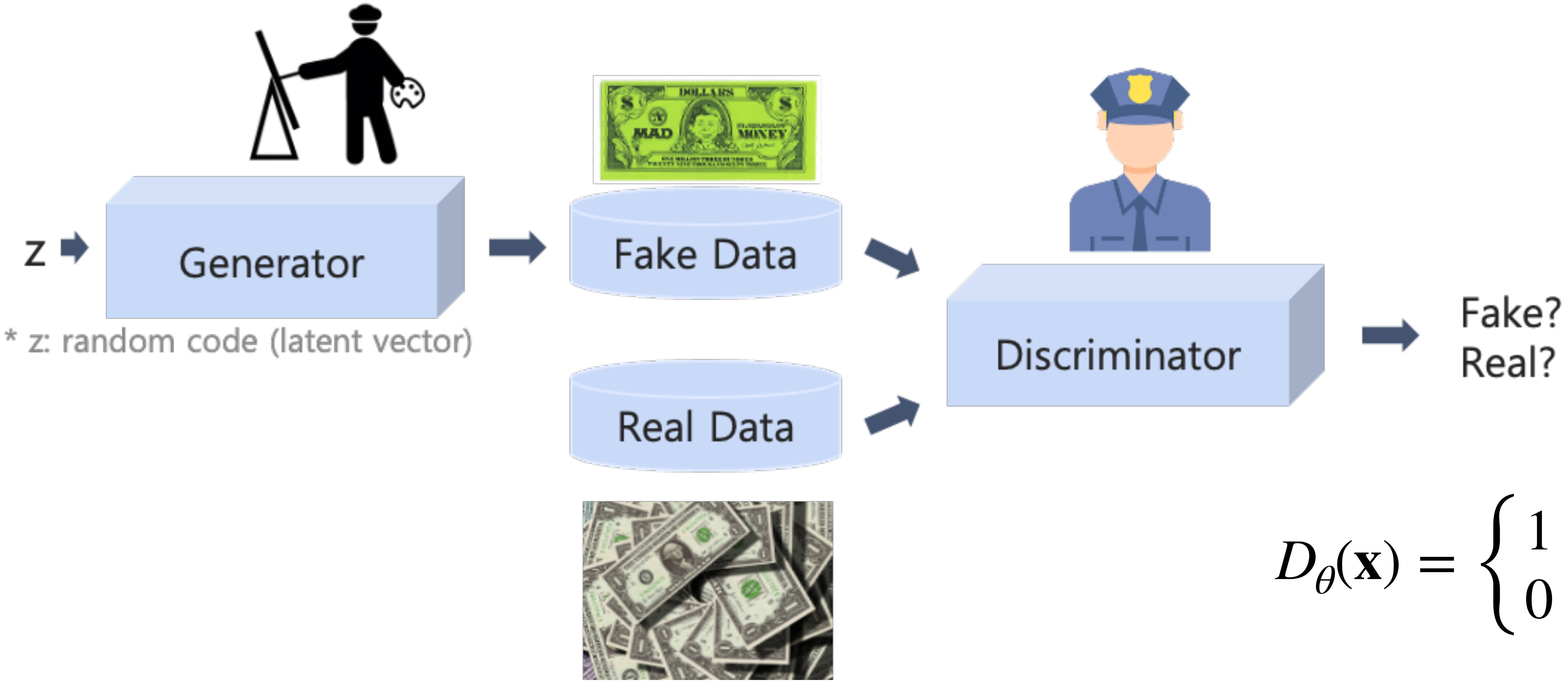
Generative Adversarial Nets

Limitations of VAEs

- **Cons.** Known to be less "sharp," with much noises
  - Clearly distinguishable from the real images
  - Question. Can we generate samples that are undistinguishable from real ones?

Generative Adversarial Nets

- **Idea.** Explicitly train for “hard to distinguish” properties, by training a distinguisher together
  - View generative process as a two-player game
    - Generator. Tries to fool the discriminator
    - Discriminator. Tries to distinguish the real / fake images



$$D_\theta(\mathbf{x}) = \begin{cases} 1 & \cdots \quad \text{if fake} \\ 0 & \cdots \quad \text{if real} \end{cases}$$

Generative Adversarial Nets

- **Training.** Jointly train the generator and discriminator
  - Objective. Minimax function

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{\mathbf{x}\sim\hat{p}} \log D_{\theta_d}(\mathbf{x}) + \mathbb{E}_{\mathbf{z}\sim p(z)} \log(1 - D_{\theta_d} \circ G_{\theta_g}(z))\right]$$

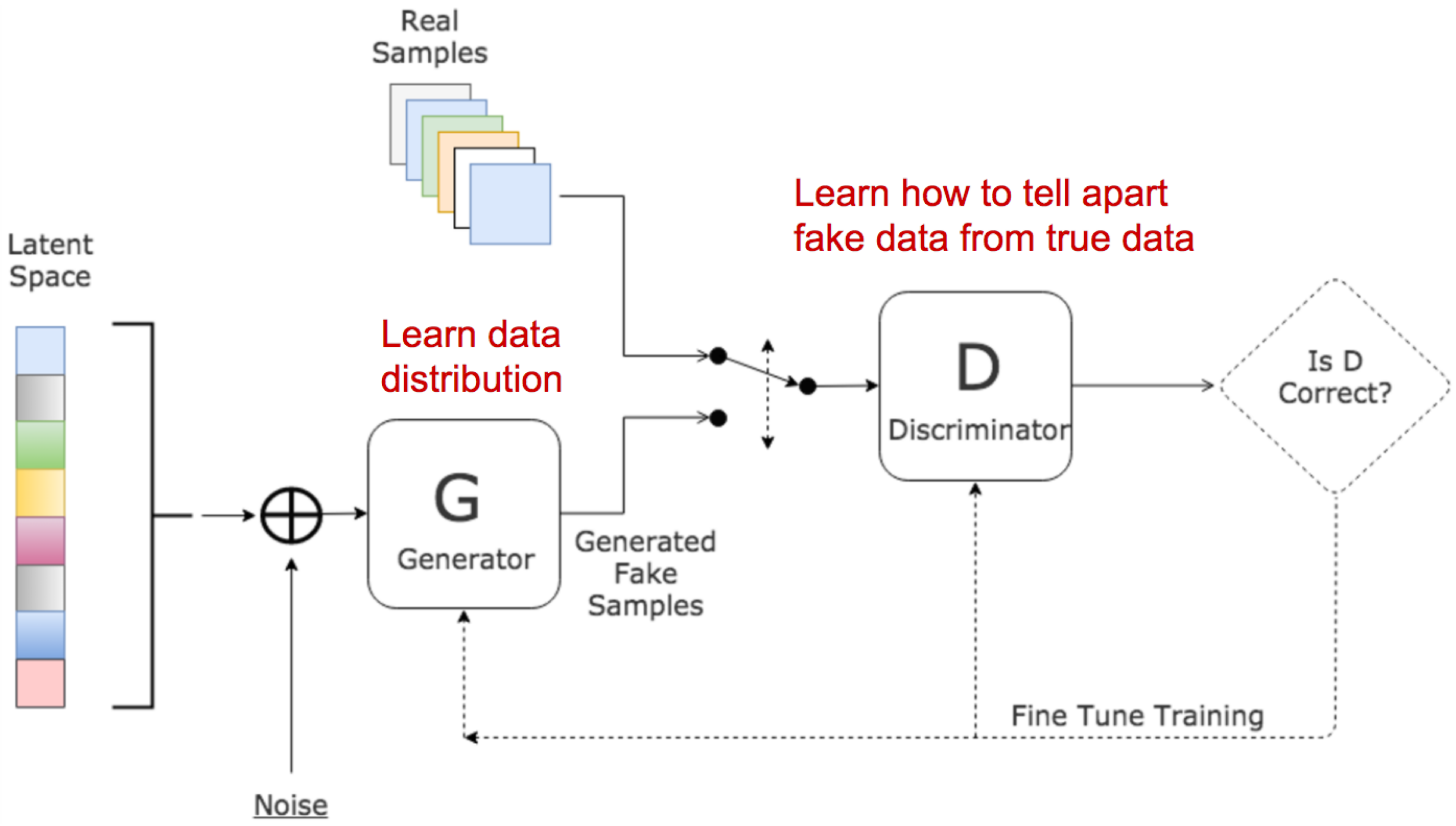
Discriminator declares real image to be real

Discriminator declares fake image to be fake

- Discriminator outputs the likelihood of being real $D_{\theta_d}(\mathbf{x}) \in [0,1]$
- This training objective is actually equivalent to the Jensen-Shannon divergence

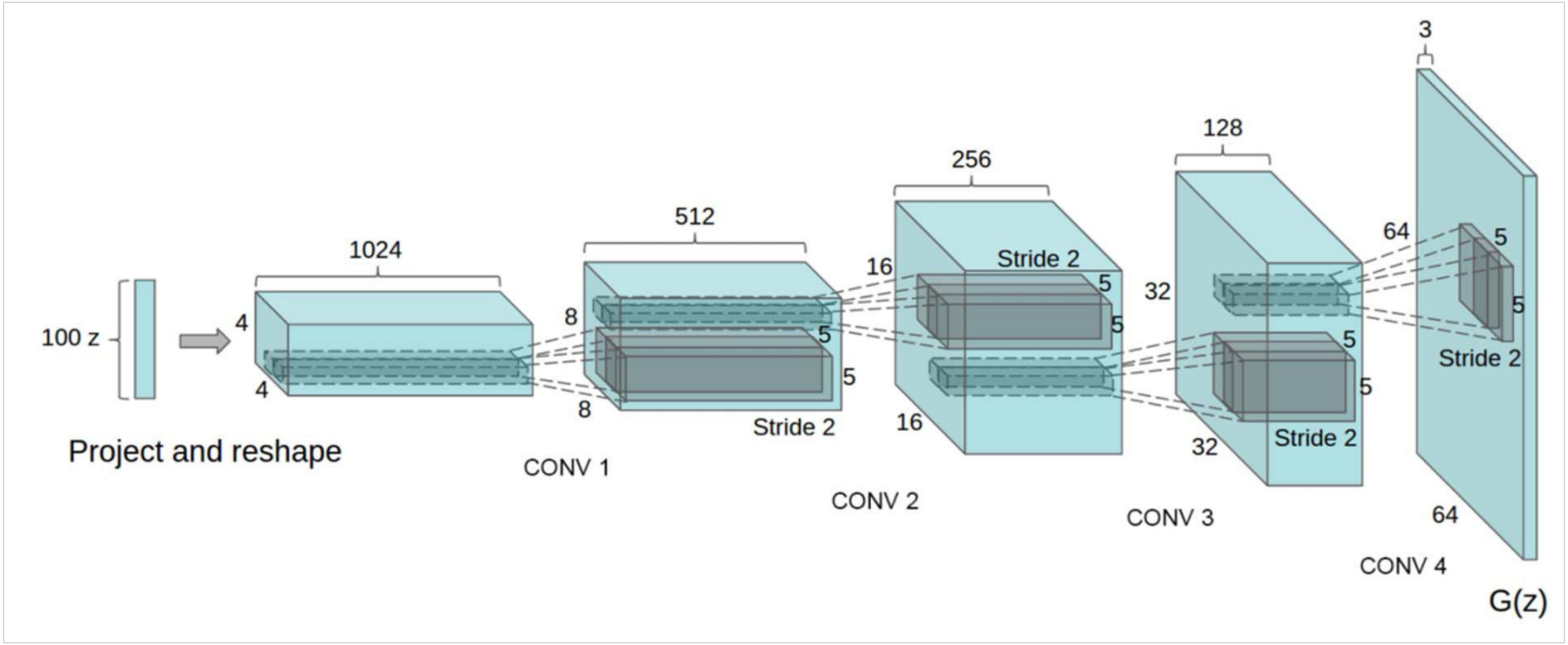
$$D\left(p_\theta \middle\| \frac{\hat{p} + p_\theta}{2}\right) + D\left(\hat{p} \middle\| \frac{\hat{p} + p_\theta}{2}\right)$$

Generative Adversarial Nets



Generative Adversarial Nets

- **Architecture.** Generator uses convolutional layers, of course.

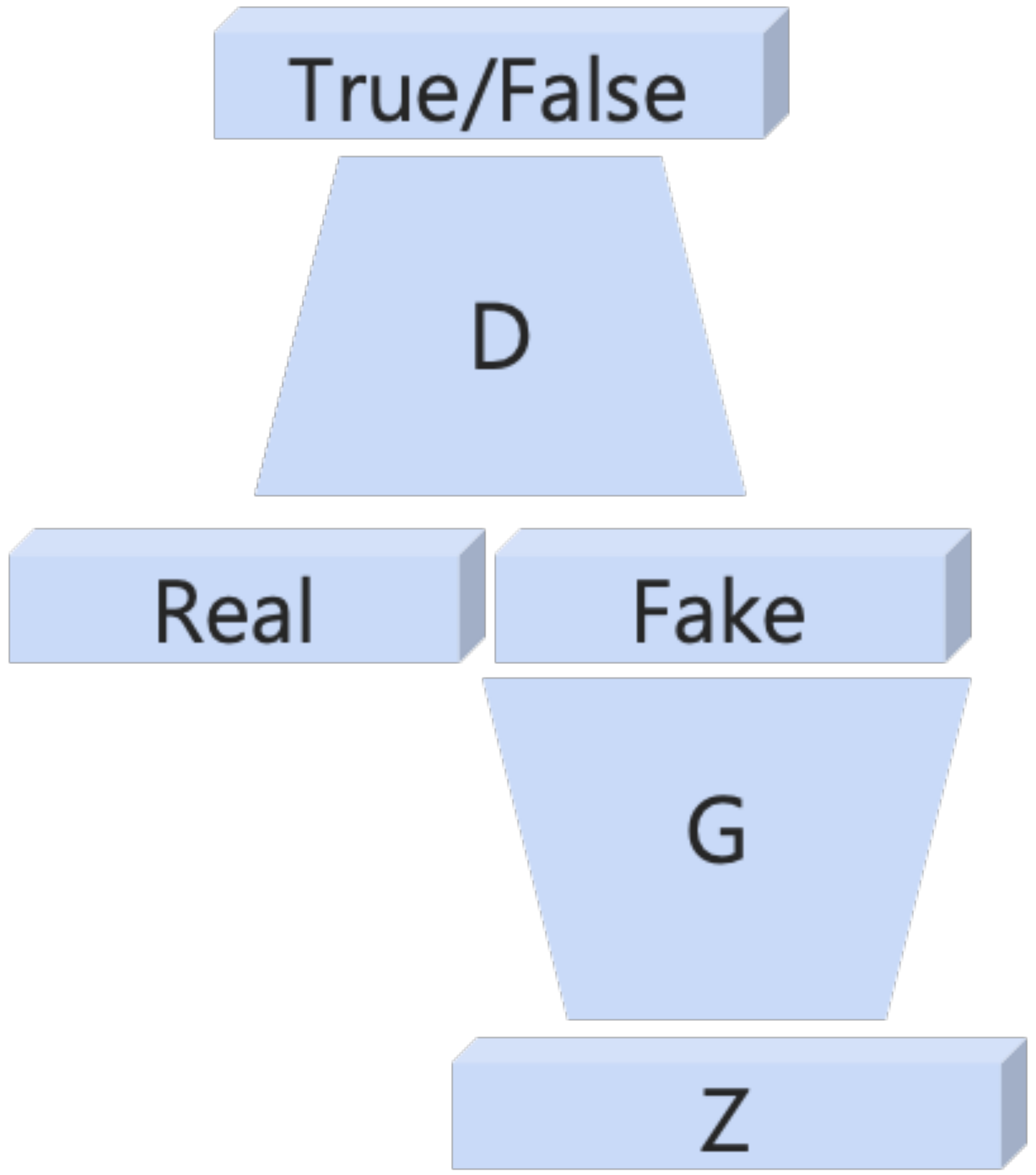


Results

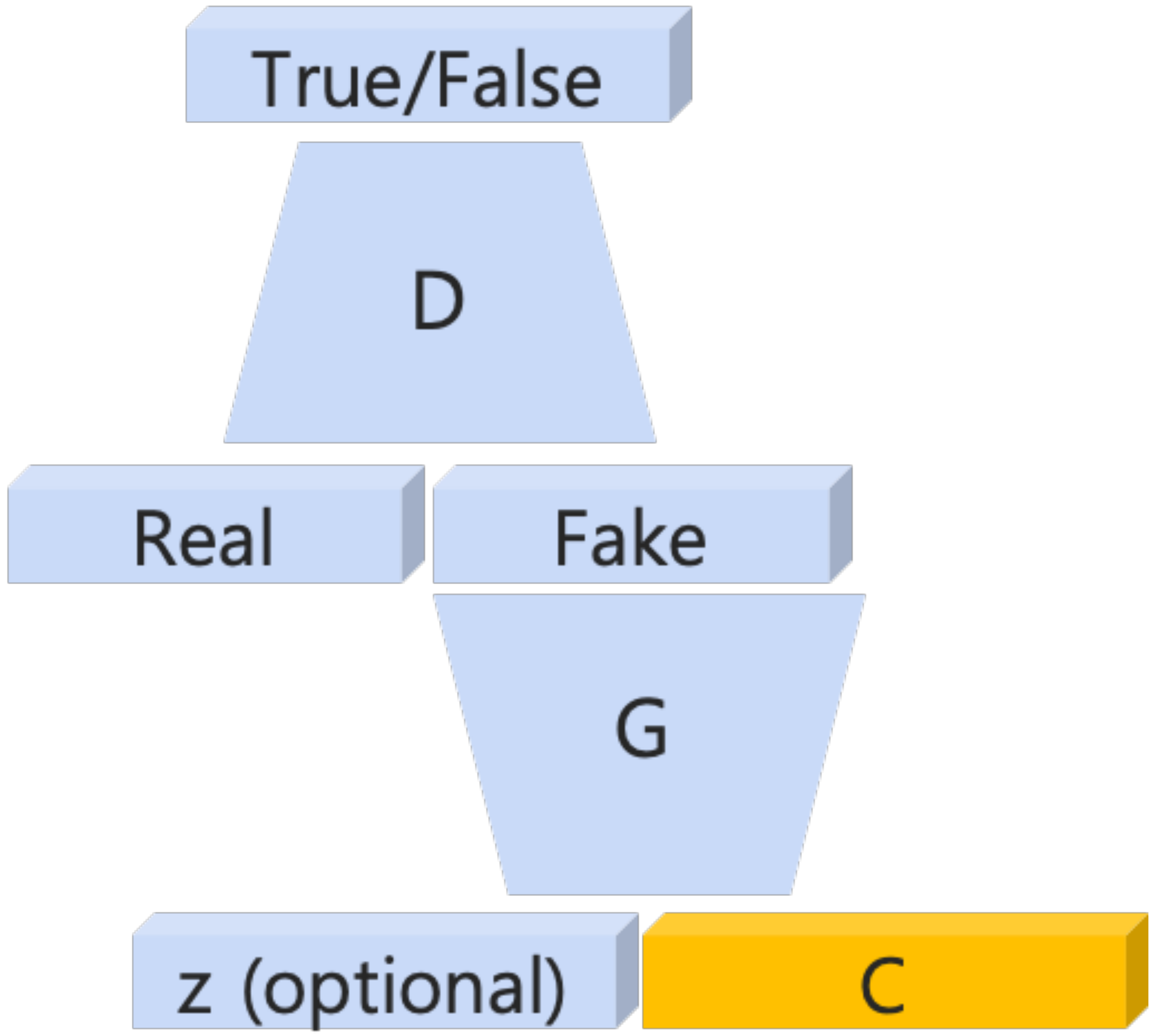
- Such training can give very sharp images

Conditional GAN

- **Idea.** Add class/text information to the latent code
    - Generate realistic images under specific conditions



GAN

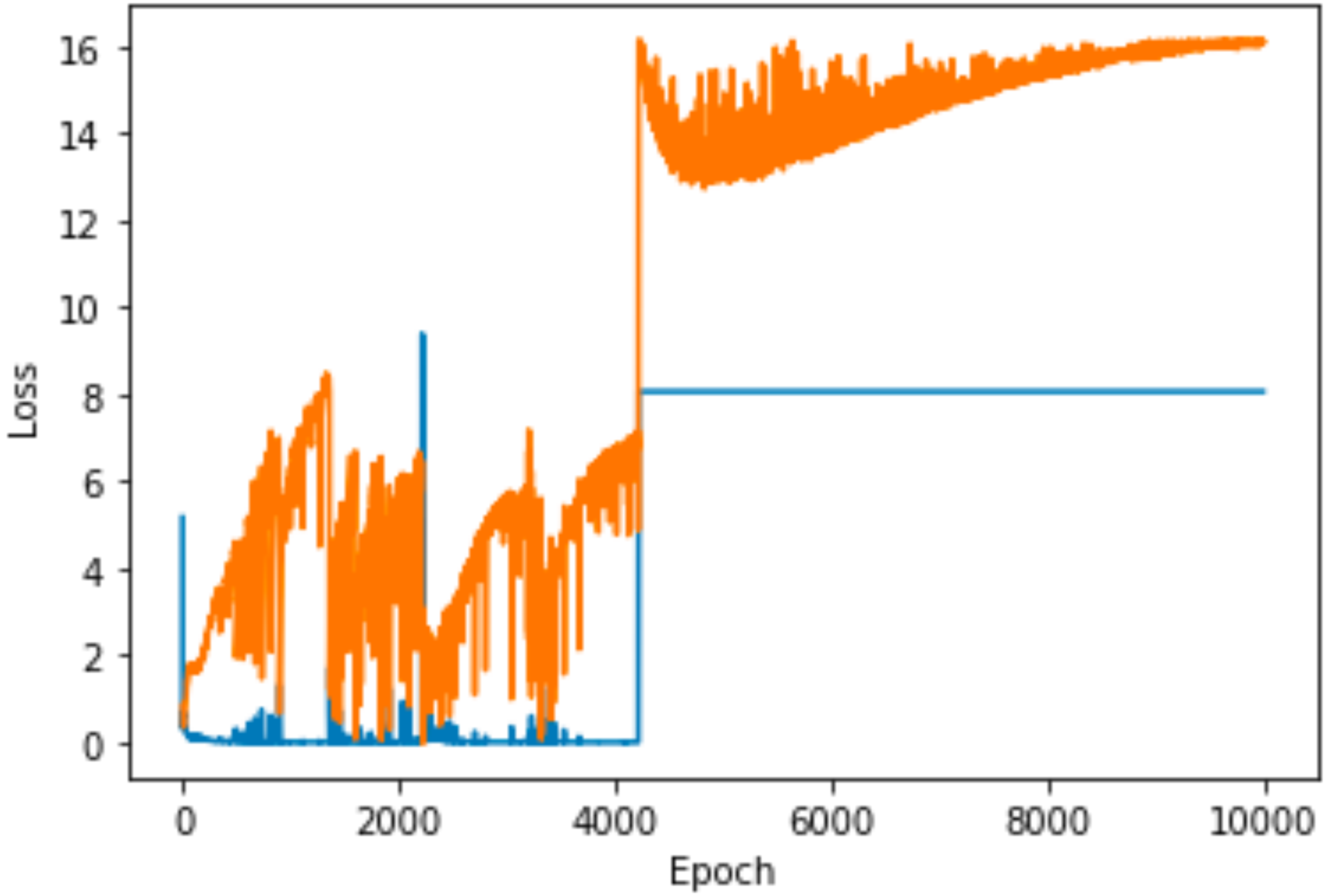


Conditional GAN

Conditional GAN

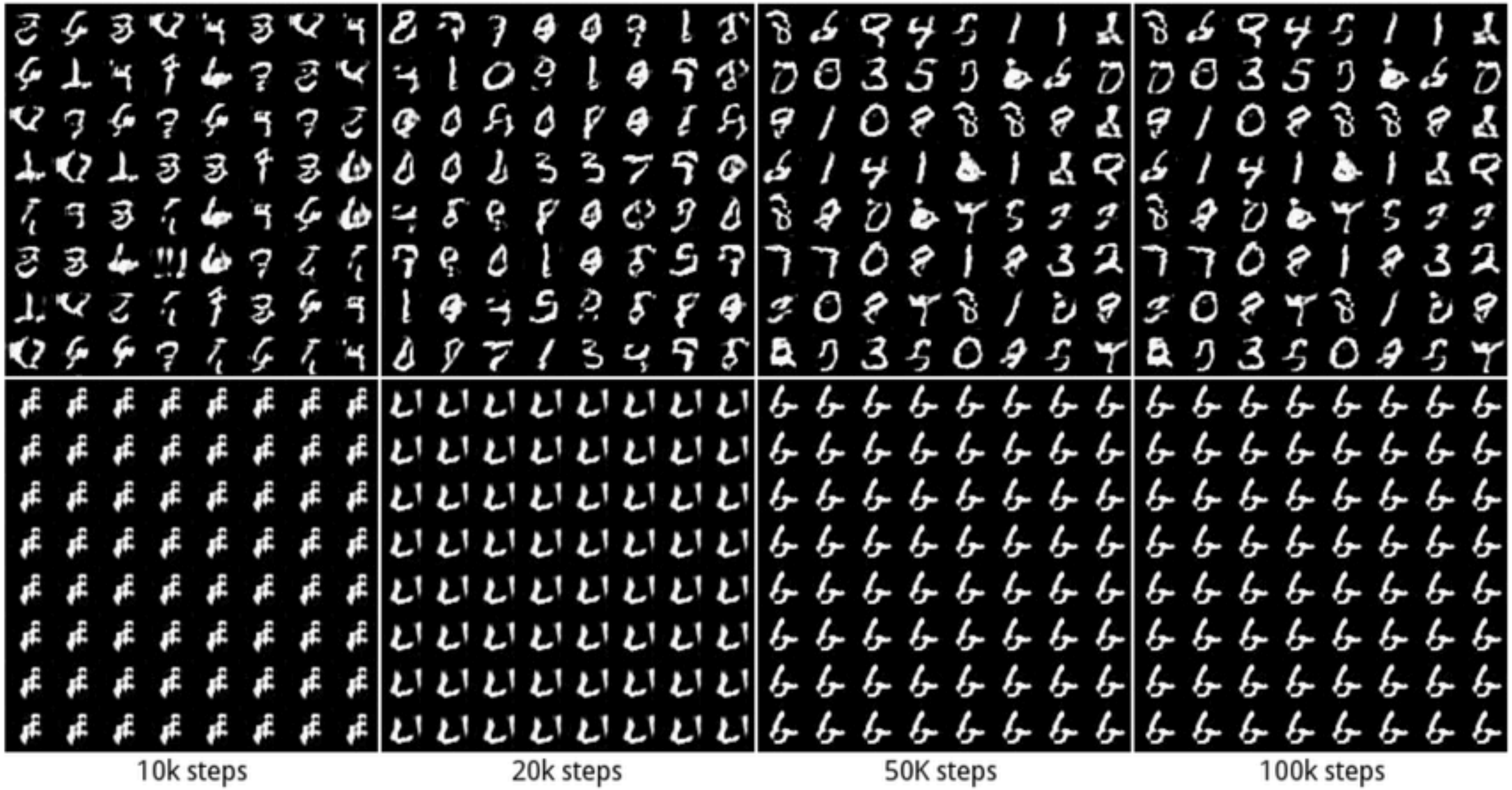
Pitfalls

- Training GANs is known to be a very unstable procedure
  - If the discriminator works too well, the generator gives up learning
  - If the generator works too well, the discriminator cannot find meaningful patterns



Pitfalls

- As a result, overfit to few good solutions (called "mode collapse")



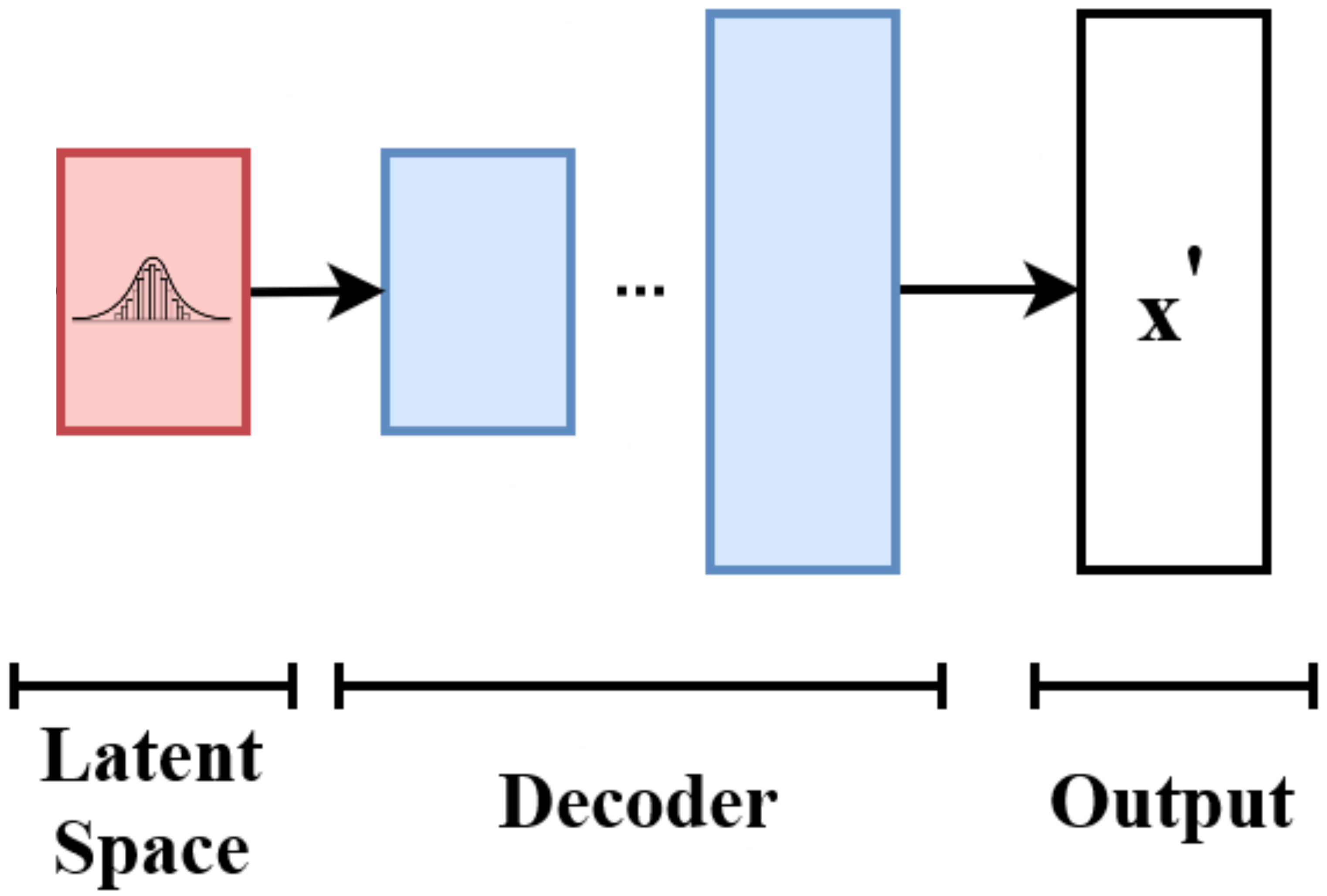
Diffusion models

VAEs

- **Recall: VAEs.** A decoder $p_\theta(\mathbf{x} \mid \mathbf{z})$ that generates samples from a code $\mathbf{z} \sim \mathcal{N}(0, I_k)$, such that

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

  - Problem. To train such model, we needed a good inverse map $p_\theta(\mathbf{z} \mid \mathbf{x})$

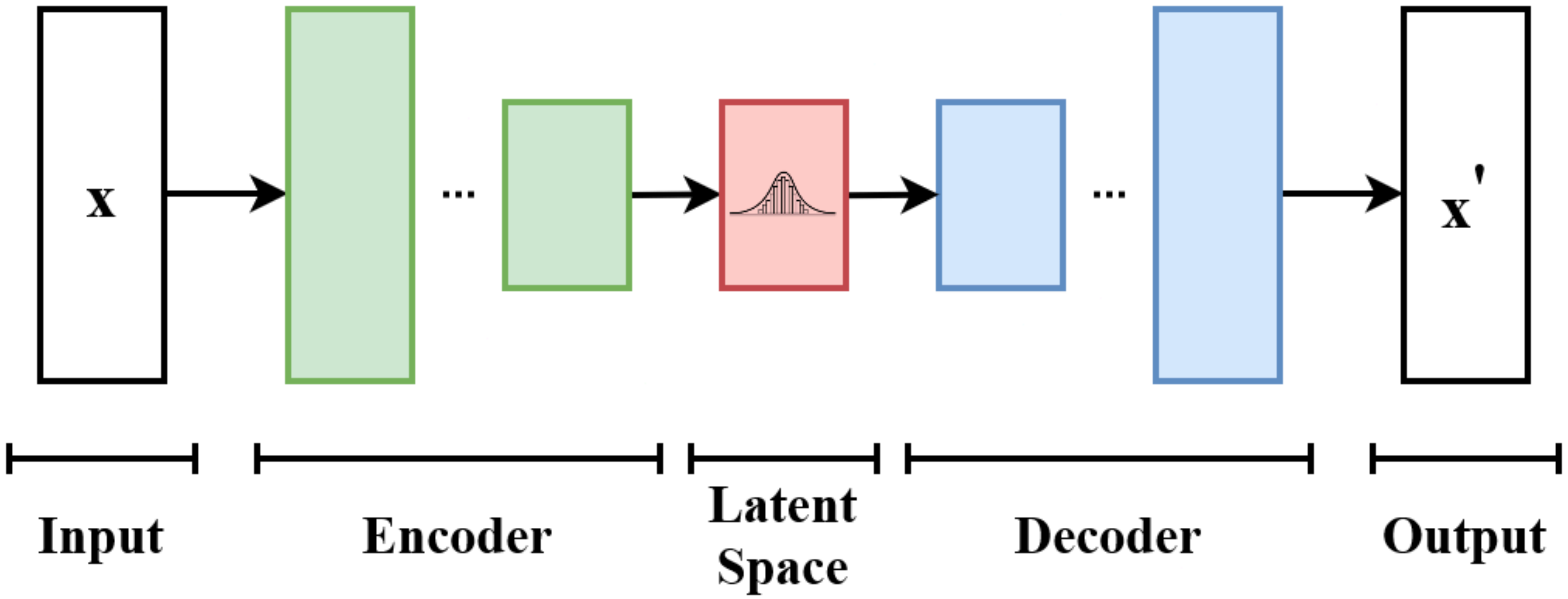


VAEs

- **Recall: VAEs.** A decoder $p_\theta(\mathbf{x}\,|\,\mathbf{z})$ that generates samples from a code $\mathbf{z} \sim \mathcal{N}(0, I_k)$, such that

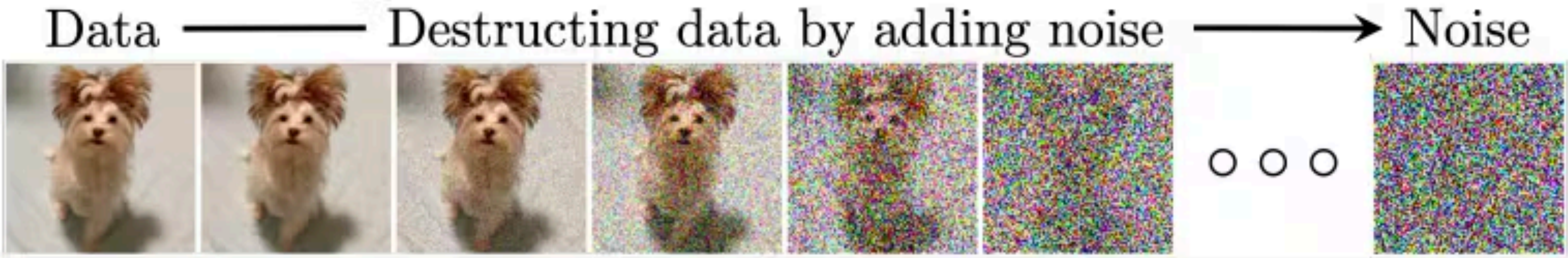
$$p_{\text{data}}(\mathbf{x}) \approx p_\theta(\mathbf{x})$$

  - Problem. To train such model, we needed a good inverse map $p_\theta(\mathbf{z}\,|\,\mathbf{x})$
  - Idea. Jointly train an encoder, which generates Gaussians from inputs
    - As the “distribution of images” is very complicated, maybe our neural nets have too low capacity to do this in a single forward…



Diffusion models

- **Observation.** Another natural way to generate Gaussian-like distribution from inputs (i.e., encode)
  - Add Gaussian noise to the input, gradually:



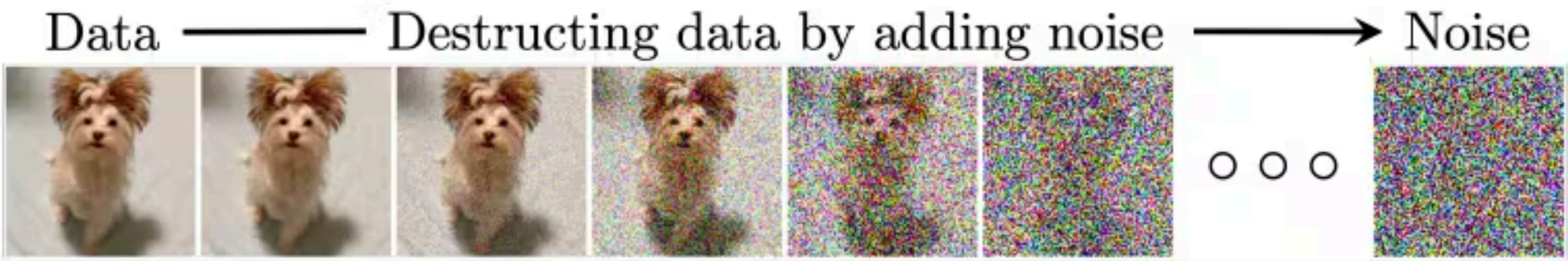
Diffusion models

- **Observation.** Another natural way to generate Gaussian-like distribution from inputs (i.e., encode)
  - Add Gaussian noise to the input, gradually:
    - Sample the data $\mathbf{x}_t$ from the distribution

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

    - That is, we do $\mathbf{x} \mapsto \sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\epsilon, \qquad \epsilon \sim \mathcal{N}(0,I)$

      (We put scaling to preserve the ℓ_2 norm)



Diffusion models

- **Observation.** Another natural way to generate Gaussian-like distribution from inputs (i.e., encode)
  - Add Gaussian noise to the input, gradually:
    - Sample the data $\mathbf{x}_t$ from the distribution

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t \mid \sqrt{\alpha_t}\mathbf{x}_{t-1}, (1-\alpha_t)I\right)$$

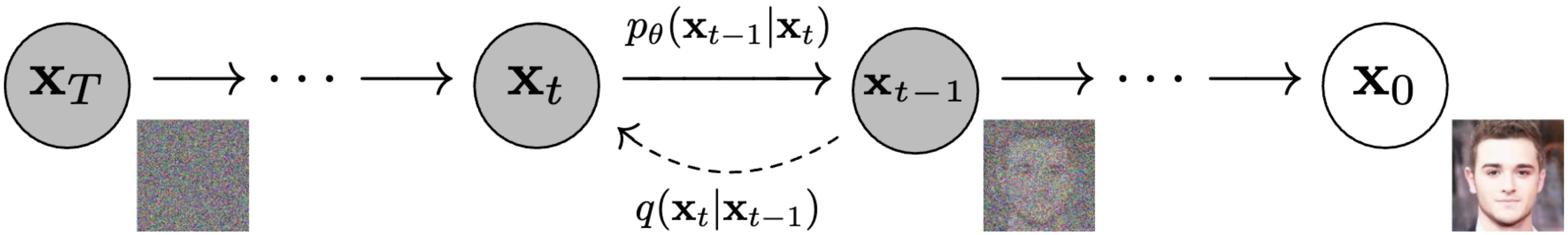
    - That is, we do $\mathbf{x} \mapsto \sqrt{\alpha_t}\mathbf{x} + \sqrt{1-\alpha_t}\epsilon, \qquad \epsilon \sim \mathcal{N}(0,I)$

      (We put scaling to preserve the ℓ_2 norm)

- **Idea.** Let this be our (probabilistic) encoder!
  - <u>Question</u>. How can we train a decoder?

Diffusion models

- **Decoder.** Train a reverse model $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ which approximates $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$

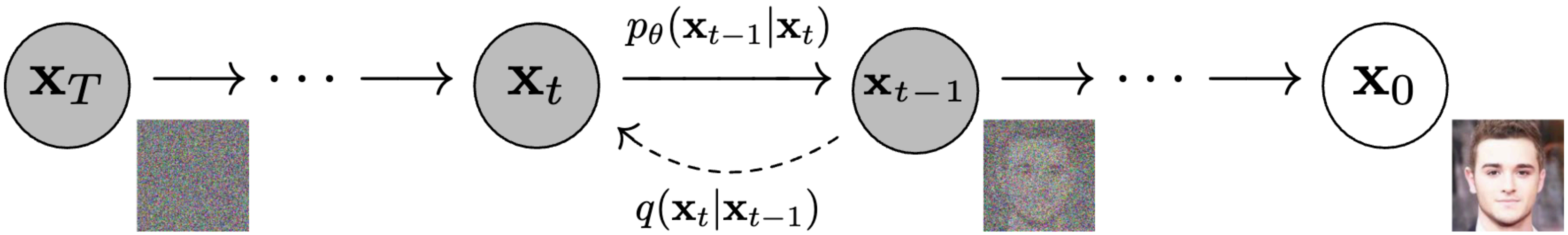


Diffusion models

- **Decoder.** Train a reverse model $p_\theta(\mathbf{x}_{t-1} \,|\, \mathbf{x}_t)$ which approximates $q(\mathbf{x}_{t-1} \,|\, \mathbf{x}_t)$
    - This reverse model will be parameterized as a Gaussian:

$$p_\theta(\mathbf{x}_{t-1} \,|\, \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1} \,|\, \mu_{\theta,t}(\mathbf{x}_t), \Sigma_{\theta,t}(\mathbf{x}_t))$$

    - That is, we train the mean predictor and variance predictor.
        - Dependent on the time t



Training a Diffusion Model

- **Training.** Suppose that we draw some sample sequence $\mathbf{x}_0, \ldots, \mathbf{x}_T$ using the forward diffusion:

$$q(\mathbf{x}_{0:T}) = q(\mathbf{x}_0) \prod_{t=1}^{T} q(\mathbf{x}_t \,|\, \mathbf{x}_{t-1})$$

Training a Diffusion Model

- **Training.** Suppose that we draw some sample sequence $\mathbf{x}_0, \ldots, \mathbf{x}_T$ using the forward diffusion:

$$q(\mathbf{x}_{0:T}) = q(\mathbf{x}_0) \prod_{t=1}^{T} q(\mathbf{x}_t \,|\, \mathbf{x}_{t-1})$$

Then, train to maximize the log probability of generating the real image

$$\mathbb{E}_{q(\mathbf{x}_0)} \left[\log p_\theta(\mathbf{x}_0) \right]$$

where the reverse diffusion process is given as:

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

Evidence Lower Bound

- As in VAE, we use the Jensen's inequality.

$$
\begin{aligned}
\mathbb{E}_{q(\mathbf{x}_0)}\left[\log p_\theta(\mathbf{x}_0)\right] &= \mathbb{E}_{q(\mathbf{x}_0)}\left[\log\left(\int p_\theta(\mathbf{x}_{0:T})\,\mathrm{d}\mathbf{x}_{1:T}\right)\right] \\
&= \mathbb{E}_{q(\mathbf{x}_0)}\left[\log\left(\int q(\mathbf{x}_{1:T}\,|\,\mathbf{x}_0)\frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}\,|\,\mathbf{x}_0)}\,\mathrm{d}\mathbf{x}_{1:T}\right)\right] \\
&= \mathbb{E}_{q(\mathbf{x}_0)}\left[\log \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}\,|\,\mathbf{x}_0)}\right]\right] \\
&\geq \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[\log\frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}\,|\,\mathbf{x}_0)}\right]
\end{aligned}
$$

Evidence Lower Bound

- The ELBO is further decomposed into:

$$\mathbb{E}_{q(\mathbf{x}_{0:T})}\left[\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}\,|\,\mathbf{x}_0)}\right] = \mathbb{E}_q\left[\log \frac{p_\theta(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)}{\prod_{t=1}^{T} q(\mathbf{x}_t\,|\,\mathbf{x}_{t-1})}\right]$$

$$= \mathbb{E}_q\left[\log p_\theta(\mathbf{x}_T) + \sum_{t=1}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)}{q(\mathbf{x}_t\,|\,\mathbf{x}_{t-1})}\right]$$

$$= \mathbb{E}_q\left[\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)}{q(\mathbf{x}_t\,|\,\mathbf{x}_{t-1})} + \log \frac{p_\theta(\mathbf{x}_0\,|\,\mathbf{x}_1)}{q(\mathbf{x}_1\,|\,\mathbf{x}_0)}\right]$$

Evidence Lower Bound

- Do additional conditioning

$$\mathbb{E}_q\left[\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^{T}\log\frac{p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)}{q(\mathbf{x}_t\,|\,\mathbf{x}_{t-1})} + \log\frac{p_\theta(\mathbf{x}_0\,|\,\mathbf{x}_1)}{q(\mathbf{x}_1\,|\,\mathbf{x}_0)}\right]$$

$$= \mathbb{E}_q\left[\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^{T}\log\left(\frac{p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)}{q(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t,\mathbf{x}_0)}\cdot\frac{q(\mathbf{x}_{t-1}\,|\,\mathbf{x}_0)}{q(\mathbf{x}_t\,|\,\mathbf{x}_0)}\right) + \log\frac{p_\theta(\mathbf{x}_0\,|\,\mathbf{x}_1)}{q(\mathbf{x}_1\,|\,\mathbf{x}_0)}\right]$$

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

Evidence Lower Bound

- Do additional conditioning

$$\mathbb{E}_q\left[\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)}{q(\mathbf{x}_t\,|\,\mathbf{x}_{t-1})} + \log \frac{p_\theta(\mathbf{x}_0\,|\,\mathbf{x}_1)}{q(\mathbf{x}_1\,|\,\mathbf{x}_0)}\right]$$

$$= \mathbb{E}_q\left[\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^{T} \log \left(\frac{p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)}{q(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t, \mathbf{x}_0)} \cdot \frac{q(\mathbf{x}_{t-1}\,|\,\mathbf{x}_0)}{q(\mathbf{x}_t\,|\,\mathbf{x}_0)}\right) + \log \frac{p_\theta(\mathbf{x}_0\,|\,\mathbf{x}_1)}{q(\mathbf{x}_1\,|\,\mathbf{x}_0)}\right]$$

$$= \mathbb{E}_q\left[\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)}{q(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t, \mathbf{x}_0)} + \sum_{t=2}^{T} \log \frac{q(\mathbf{x}_{t-1}\,|\,\mathbf{x}_0)}{q(\mathbf{x}_t\,|\,\mathbf{x}_0)} + \log \frac{p_\theta(\mathbf{x}_0\,|\,\mathbf{x}_1)}{q(\mathbf{x}_1\,|\,\mathbf{x}_0)}\right]$$

$$= \mathbb{E}_q\left[\log \frac{p_\theta(\mathbf{x}_T)}{q(\mathbf{x}_T\,|\,\mathbf{x}_0)} + \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)}{q(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t, \mathbf{x}_0)} + \log p_\theta(\mathbf{x}_0\,|\,\mathbf{x}_1)\right]$$

Evidence Lower Bound

- Tidying up, we get

$$\mathbb{E}_q\left[\log\frac{p_\theta(\mathbf{x}_T)}{q(\mathbf{x}_T\,|\,\mathbf{x}_0)}+\sum_{t=2}^{T}\log\frac{p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)}{q(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t,\mathbf{x}_0)}+\log p_\theta(\mathbf{x}_0\,|\,\mathbf{x}_1)\right]$$

$$=\mathbb{E}_q[\log p_\theta(\mathbf{x}_0\,|\,\mathbf{x}_1)]-\mathbb{E}_q D\Big(q(\mathbf{x}_T\,|\,\mathbf{x}_0)\Big\|p(\mathbf{x}_T)\Big)-\sum_{t=2}^{T}\mathbb{E}_q D\Big(q(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t,\mathbf{x}_0)\Big\|p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)\Big)$$

Evidence Lower Bound

- Tidying up, we get

$$\mathbb{E}_q\left[\log \frac{p_\theta(\mathbf{x}_T)}{q(\mathbf{x}_T\,|\,\mathbf{x}_0)} + \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)}{q(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t,\mathbf{x}_0)} + \log p_\theta(\mathbf{x}_0\,|\,\mathbf{x}_1)\right]$$

$$= \mathbb{E}_q[\log p_\theta(\mathbf{x}_0\,|\,\mathbf{x}_1)] - \mathbb{E}_q D\Big(q(\mathbf{x}_T\,|\,\mathbf{x}_0)\Big\|p(\mathbf{x}_T)\Big) - \sum_{t=2}^{T} \mathbb{E}_q D\Big(q(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t,\mathbf{x}_0)\Big\|p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t)\Big)$$

- **First term.** We know that this is the squared loss of the mean predictor.
  - Assuming that $\Sigma = I$ for simplicity, we have:

$$\mathbb{E}_q[\log p_\theta(\mathbf{x}_0\,|\,\mathbf{x}_1)] = -\frac{1}{2}\mathbb{E}_q\|\mathbf{x}_0 - \mu_{\theta,1}(\mathbf{x}_1)\|^2$$

Evidence Lower Bound

- Tidying up, we get

$$\mathbb{E}_q\left[\log \frac{p_\theta(\mathbf{x}_T)}{q(\mathbf{x}_T \mid \mathbf{x}_0)} + \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)}{q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)} + \log p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1)\right]$$

$$= \mathbb{E}_q[\log p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1)] - \mathbb{E}_q D\left(q(\mathbf{x}_T \mid \mathbf{x}_0) \middle\| p(\mathbf{x}_T)\right) - \sum_{t=2}^{T} \mathbb{E}_q D\left(q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) \middle\| p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)\right)$$

- **First term.** We know that this is the squared loss of the mean predictor.
  - Assuming that $\Sigma = I$ for simplicity, we have:

$$\mathbb{E}_q[\log p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1)] = -\frac{1}{2}\mathbb{E}_q \|\mathbf{x}_0 - \mu_{\theta,1}(\mathbf{x}_1)\|^2$$

- **Second term.** This does not involve any learnable parameters.
  - Thus, ignore!

Evidence Lower Bound

$$-\frac{1}{2}\|\mathbf{x}_0 - \mu_{t,1}(\mathbf{x}_1)\|^2 - \sum_{t=2}^{T} \mathbb{E}_q D\left(q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) \middle\| p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)\right)$$

- **Third term.** First, we look at the LHS of the KL divergence.
  - If we have the relationship

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon$$

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\epsilon'$$

(we use the shorthands $\bar{\alpha}_i = \alpha_1 \cdot \alpha_2 \cdot \cdots \cdot \alpha_i$)

Then the following relationship holds (exercise; use Bayes' theorem)

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\mathbf{x}_0, \frac{(1-\alpha_t)(1-\sqrt{\bar{\alpha}_{t-1}})}{1-\bar{\alpha}_t}I\right)$$

Evidence Lower Bound

$$-\frac{1}{2}\|\mathbf{x}_0 - \mu_{t,1}(\mathbf{x}_1)\|^2 - \sum_{t=2}^{T} \mathbb{E}_q D\Big(q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) \Big\| p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)\Big)$$

- Now, the KL-divergence between Gaussians can be written simply as:

$$D\Big(\mathcal{N}(\mu_1, \sigma_1^2 I) \Big\| \mathcal{N}(\mu_2, \sigma_2^2 I)\Big) = \log\frac{\sigma_2}{\sigma_1} - \frac{d}{2} + \frac{d\sigma_1^2 + \|\mu_1 - \mu_2\|^2}{2\sigma_2^2}$$

  - Plug this in to get the loss (ignoring the variance terms)

$$\sum_{i=2}^{T} \left\| \mu_{\theta,t}(\mathbf{x}_t) - \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)}{1-\bar{\alpha}_t}\mathbf{x}_0 \right\|^2$$

$$=: \sum_{i=1}^{T} \|\mu_{\theta,t}(\mathbf{x}_t) - \mu_q(\mathbf{x}_t, \mathbf{x}_0)\|^2$$

In a nutshell

- In a nutshell, training the reverse diffusion process is:
  - Sample an image $\mathbf{x}_0$ from the dataset
  - Sample $\mathbf{x}_1, \ldots, \mathbf{x}_T$ using $q(\cdot)$
  - Pick a time t:
    - Train $\mu_{\theta,t}(\cdot)$ to minimize $\|\mu_{\theta,t}(\mathbf{x}_t) - \mu_q(\mathbf{x}_t, \mathbf{x}_0)\|^2$
  - Repeat

In a nutshell

- In a nutshell, trainin
  - Sample an imag
  - Sample $\mathbf{x}_1, \ldots,$
  - Pick a time t:
    - Train $\mu_{\theta,t}(\,\cdot$
  - Repeat

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

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

6: **until** converged

- This is typically reparametrized as a noise prediction (i.e., residual of the prediction)

Prediction

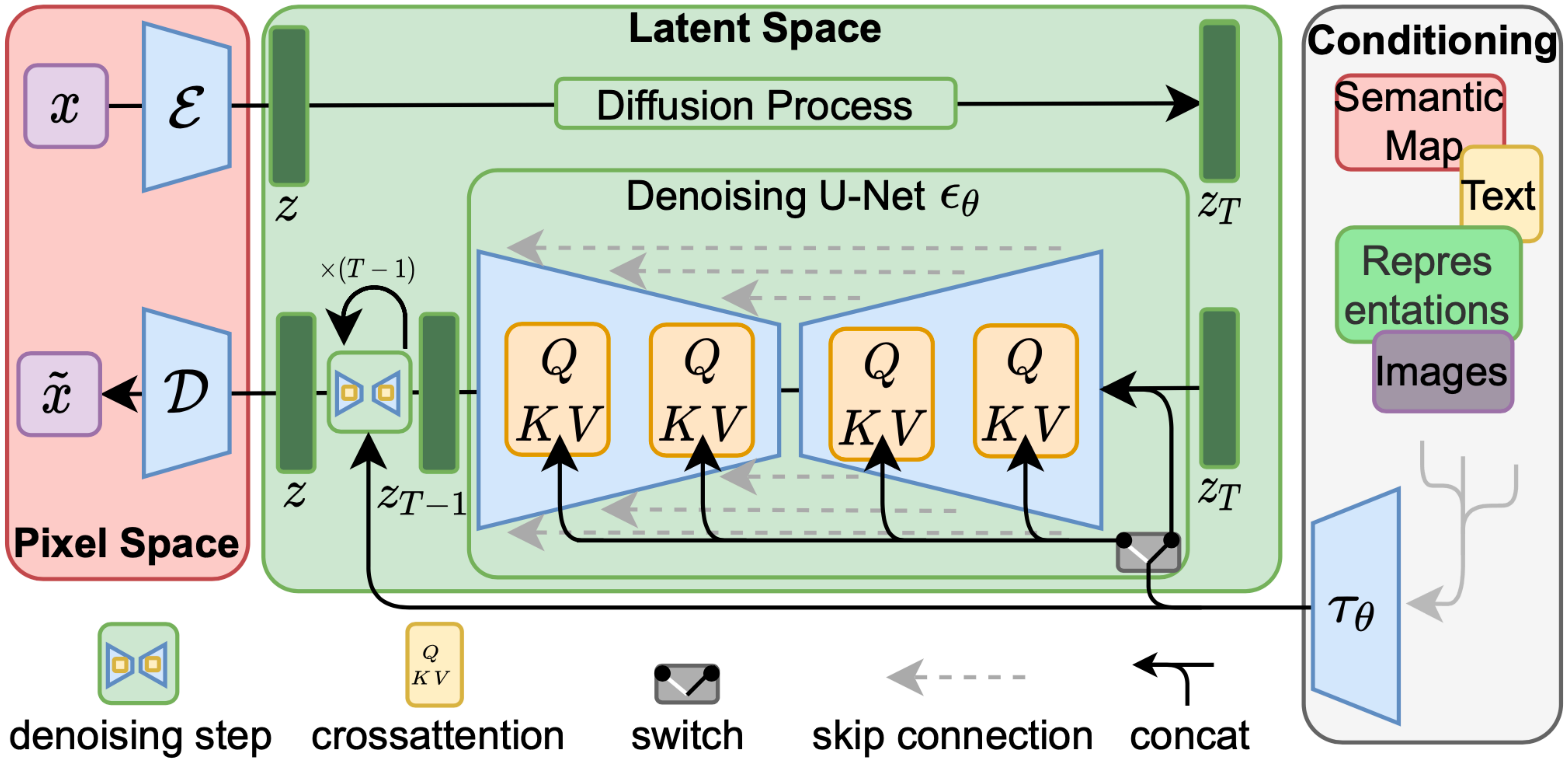
- Generation is done by starting from a Gaussian distribution, then keep denoising…

Algorithm 2 Sampling

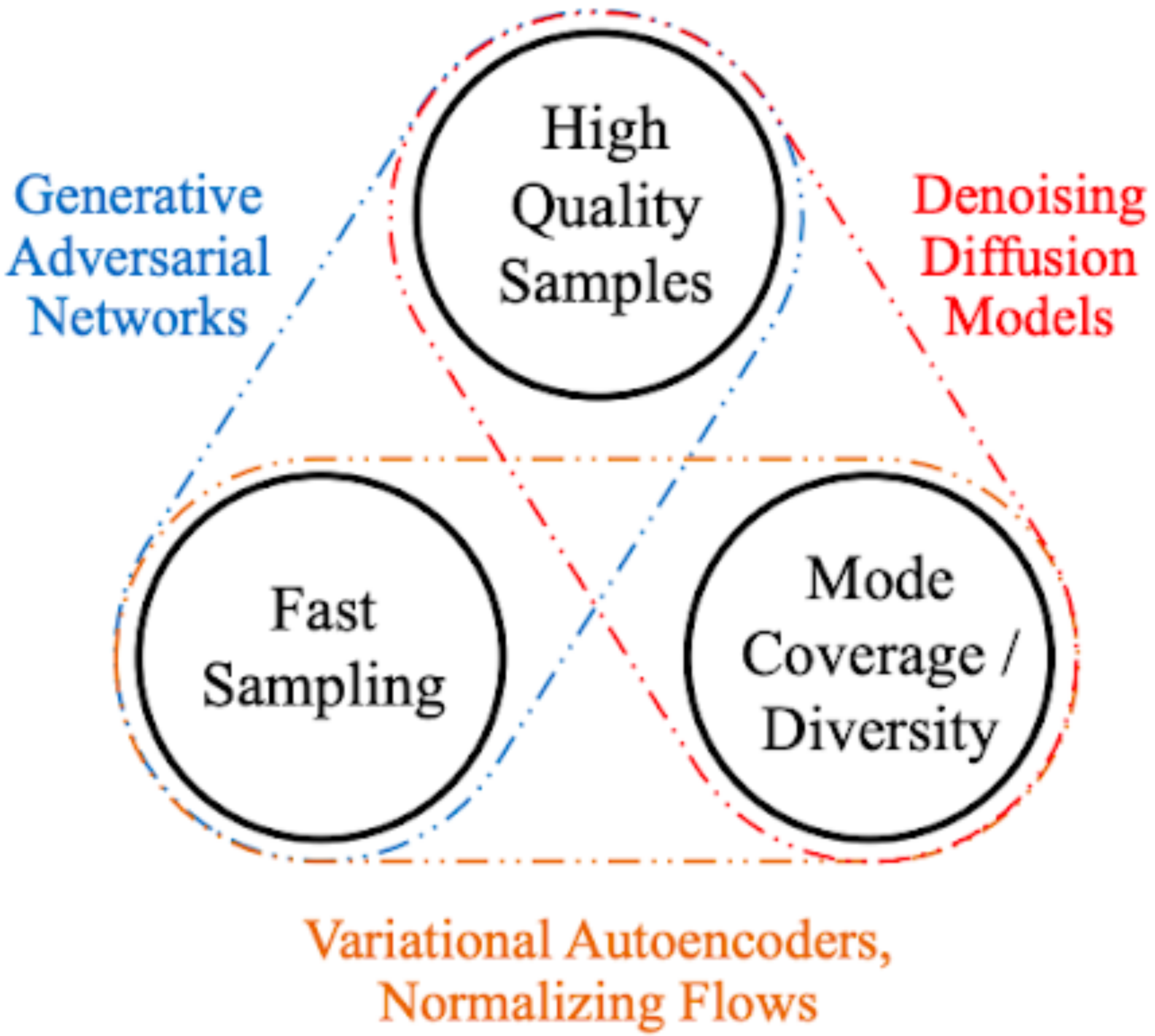
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\quad \mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}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

Latent Diffusion

- We use diffusion in some latent space.
  - Combine with the ideas of VAE
- Plus, we do some conditioning



Pros & Cons


High Quality Samples
Generative Adversarial Networks
Denoising Diffusion Models
Fast Sampling
Mode Coverage / Diversity
Variational Autoencoders, Normalizing Flows

More references

- https://huggingface.co/blog/annotated-diffusion
- https://lilianweng.github.io/posts/2021-07-11-diffusion-models/
- https://arxiv.org/abs/2403.18103

Cheers