Vision:
Generative Modeling - 1

EECE454 Intro. to Machine Learning Systems

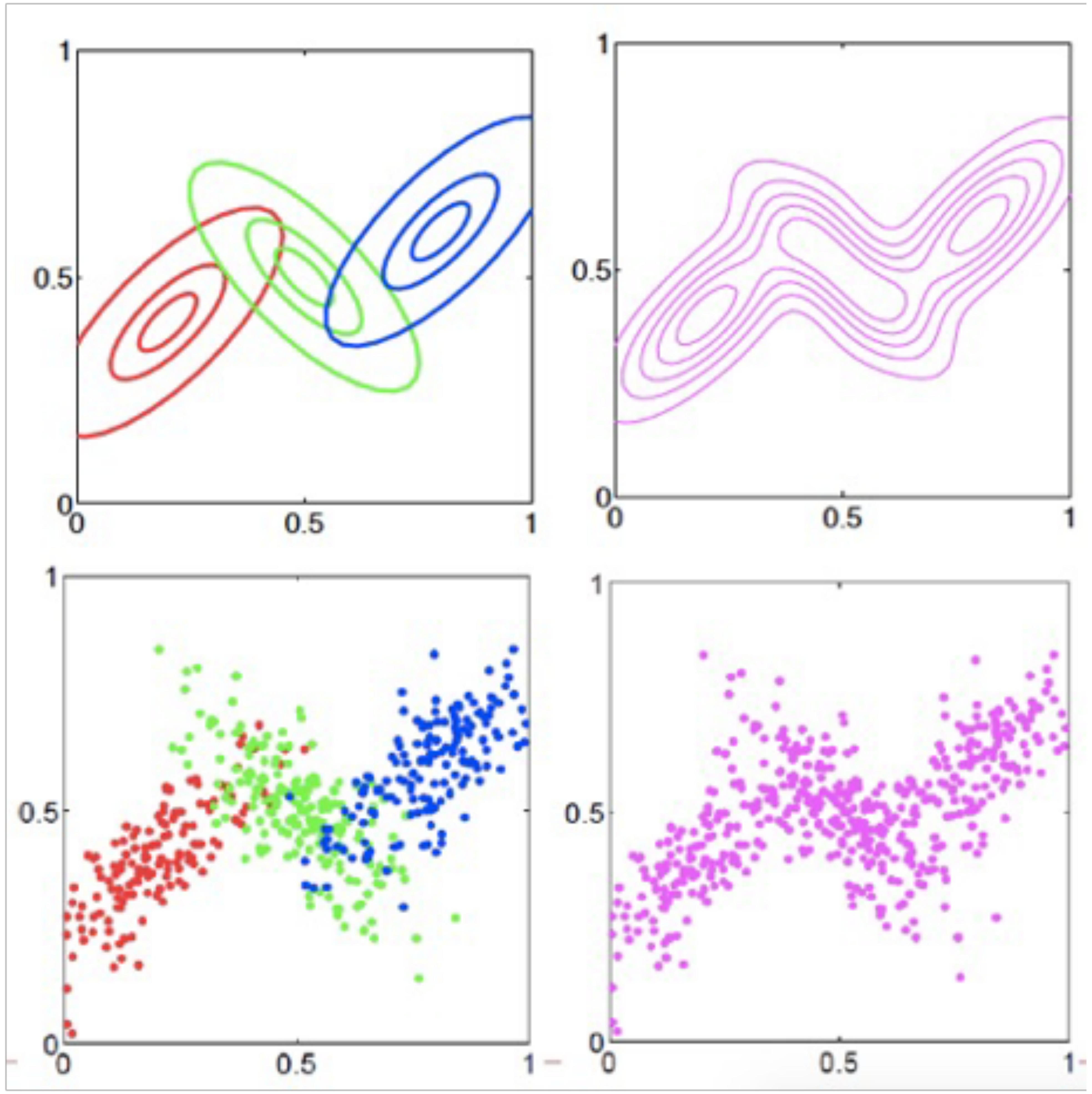
Fall 2024

Recap

- **Generative Modeling.**
  Using unlabeled training data $\mathbf{x}_1, \ldots, \mathbf{x}_n \sim p_{\text{data}}(\mathbf{x})$, approximate the data-generating distribution such that

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

- **Classic Example.** Gaussian mixture models
  - The cluster information may help performing the downstream classification
  - Helps you evaluate how likely each data is:
    - Anomaly detection, novelty detection



Generative Modeling

- In modern contexts, generative modeling has extended boundaries.

- **Modern example.** Suppose that we have learned a good model on the joint distribution

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

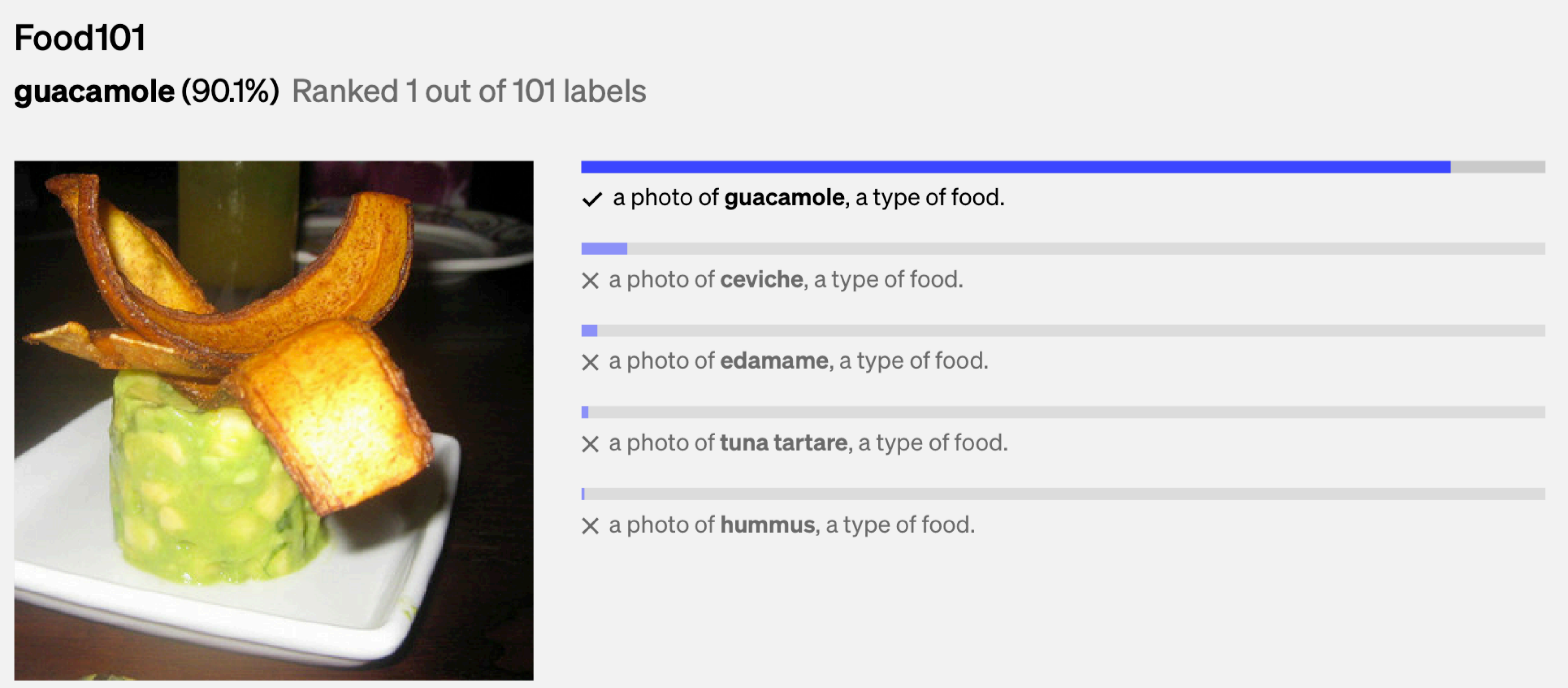
from the image-text pairs $\{(\mathbf{x}, y)\}_{i=1}^{n}$ crawled from web (treat $\mathbf{z} = (\mathbf{x}, y)$ as an unlabeled data)

Generative Modeling

- With a good generative model, we can do the following things:

- **(Generative) Classification.** Use the Bayes rule to do

$$p_\theta(y \mid \mathbf{x}) = \frac{p_\theta(\mathbf{x}, y)}{p_\theta(\mathbf{x})}$$

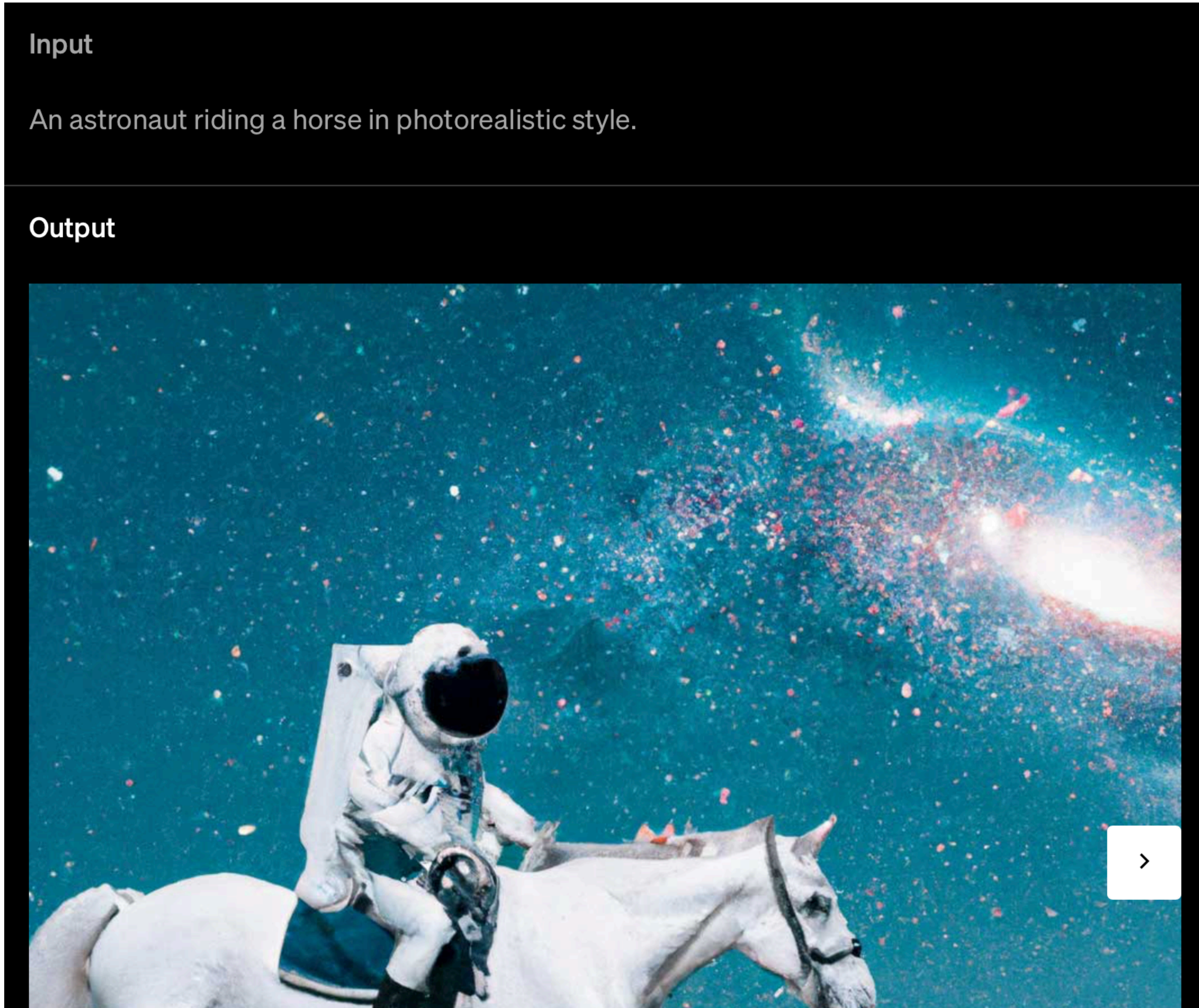


Generative Modeling

- **Text-conditional Generation.**
  Use the Bayes rule the other way—

  Generate an image that correspond to an arbitrary text

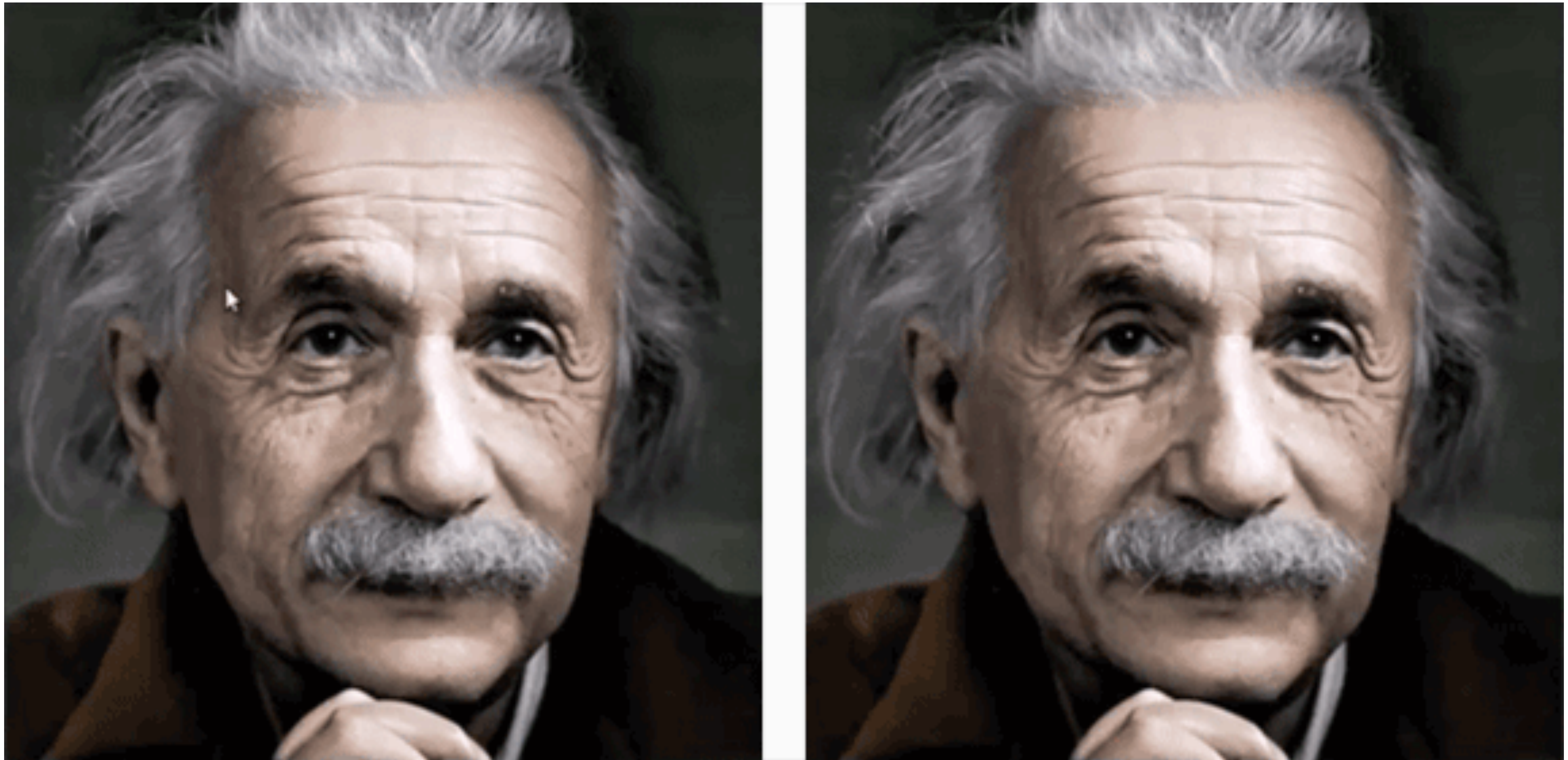
$$p_\theta(\mathbf{x} \mid y) = \frac{p_\theta(\mathbf{x}, y)}{p_\theta(y)}$$



Generative Modeling

- **Image inpainting.** Generate some part of the image that correspond to other parts:

$$p_\theta(x_i \mid x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_d)$$



Generative Modeling

- **Text generation.** Generate the next text word that correspond to the given text

$$p_\theta(y_{n+1} \mid y_1, \ldots, y_n)$$

JA **You**

To study AI, where should I go? Please give a detailed answer.

ChatGPT

To study AI, there are several excellent institutions and universities around the world that offer comprehensive programs and research opportunities. POSTECH is one of them, but there are many others as well. The choice of where to study AI depends on various factors such as your academic interests, location preferences, budget, and career goals.

Here are some top institutions known for their AI programs and research:

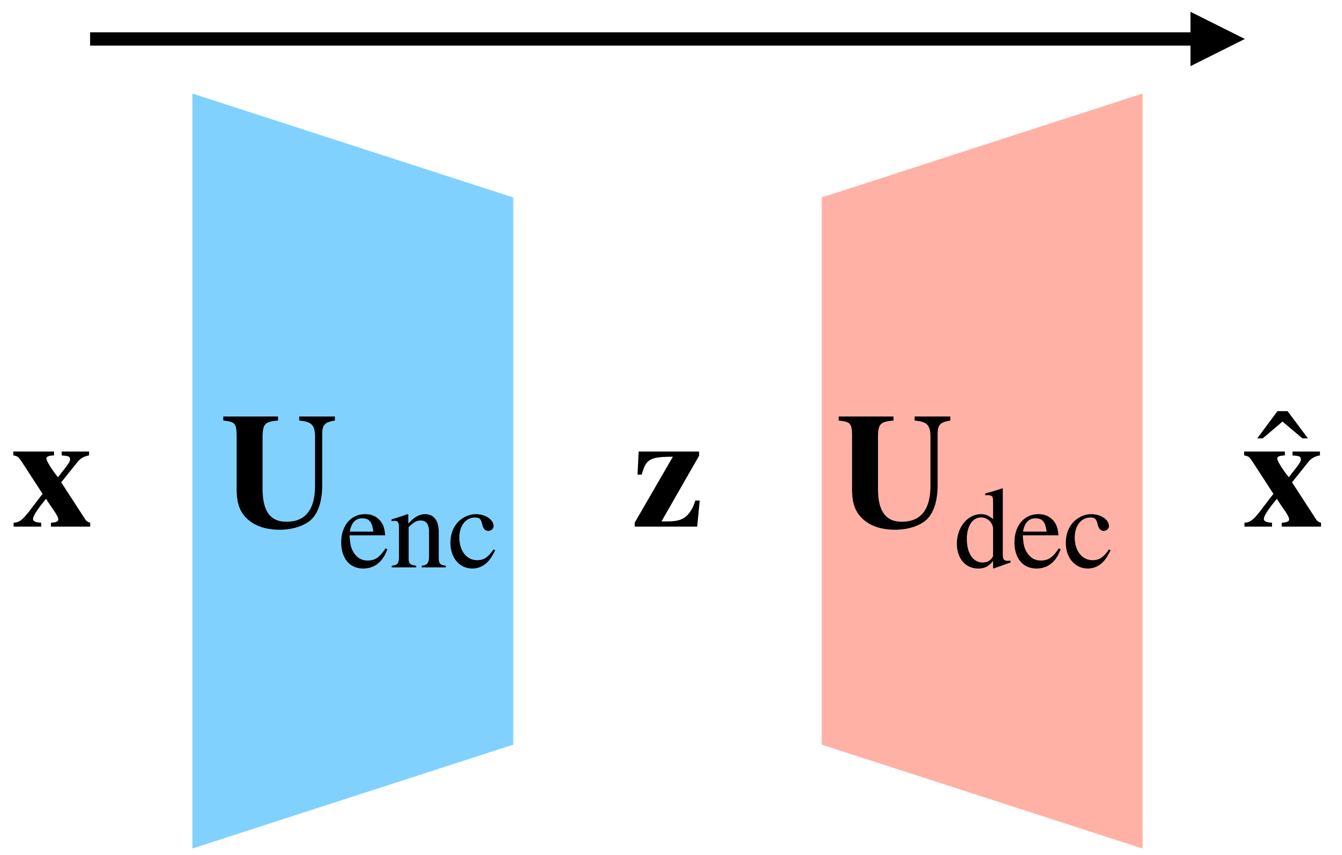
Today

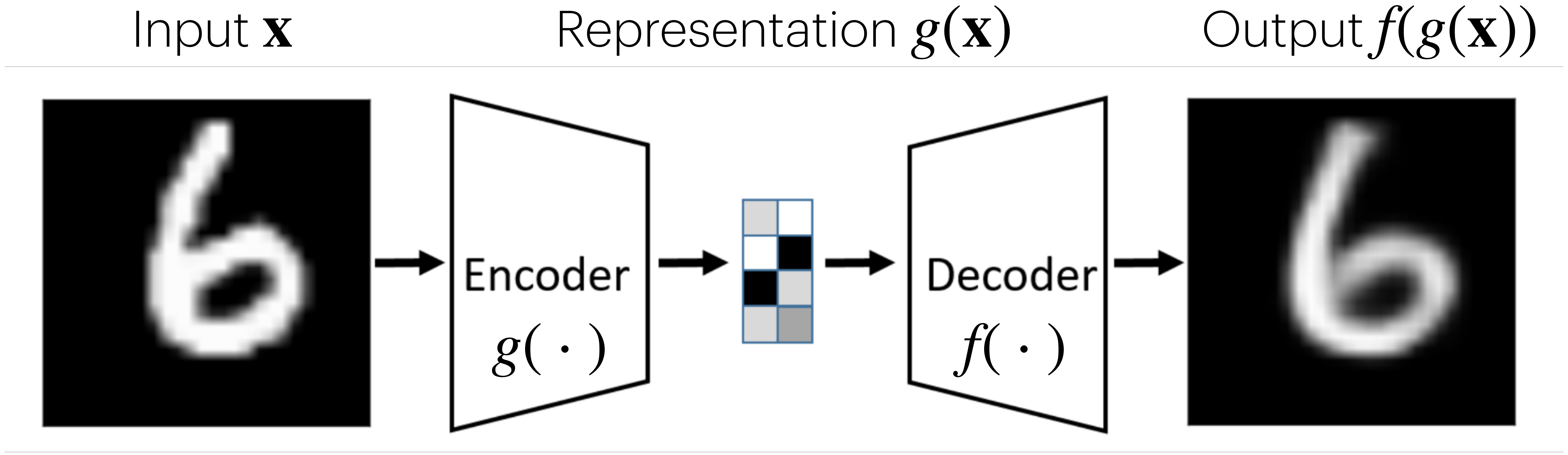
- Focusing on the generative modeling for images
  - Will cover multimodal cases later
  - **vs. Language.**
    - Need to generate many pixels for high-resolution images (Challenging to generate "realistic" ones)
    - More locality involved, with 2D/3D geometry

Autoencoders

Basic autoencoders

- An approach that has been used for representation learning, initially
- **Idea.** Train a neural network that can do PCA
  - Replace the linear model with neural networks
  - Use SGD to solve $\min_{f,g} \mathbb{E}_{\mathbf{x}} \|\mathbf{x} - f(g(\mathbf{x}))\|^2$

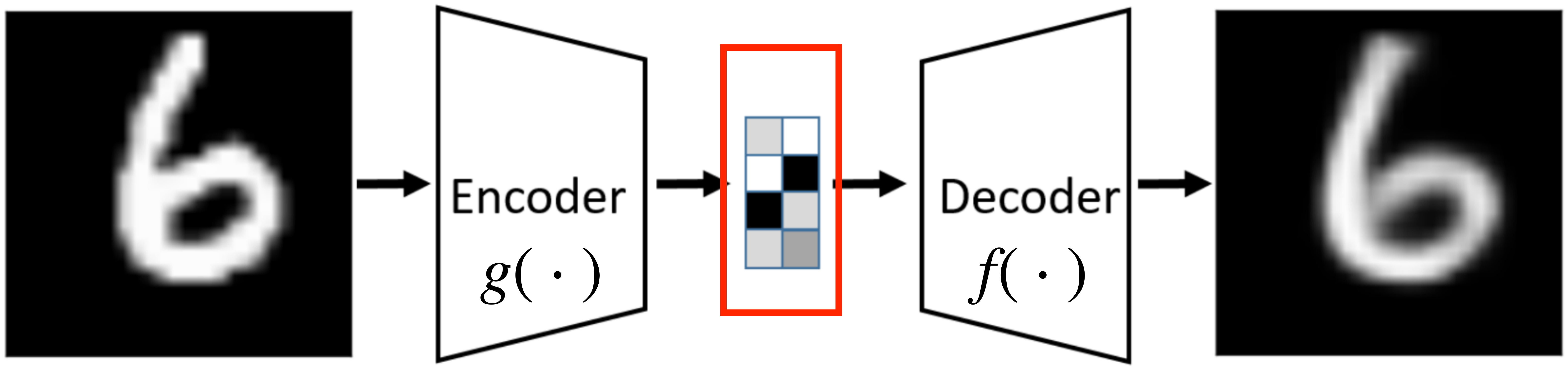




Basic autoencoders

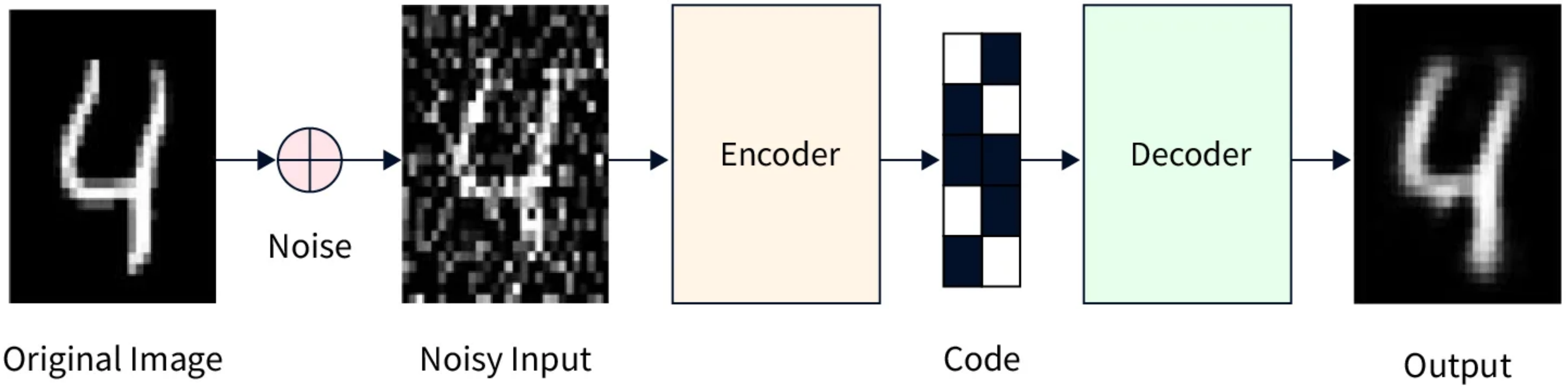
- An approach that has been used for representation learning, initially
- **Idea.** Train a neural network that can do PCA
  - Replace the linear model with neural networks
  - Use SGD to solve $\min_{f,g} \mathbb{E}_{\mathbf{x}} \|\mathbf{x} - f(g(\mathbf{x}))\|^2$
  - Note. A trivial solution $f(\cdot) = g(\cdot) = \mathbf{Identity}$ can happen
    - Avoidable with the hourglass structure (or other regularizations)

Input $\mathbf{x}$ Representation $g(\mathbf{x})$ Output $f(g(\mathbf{x}))$



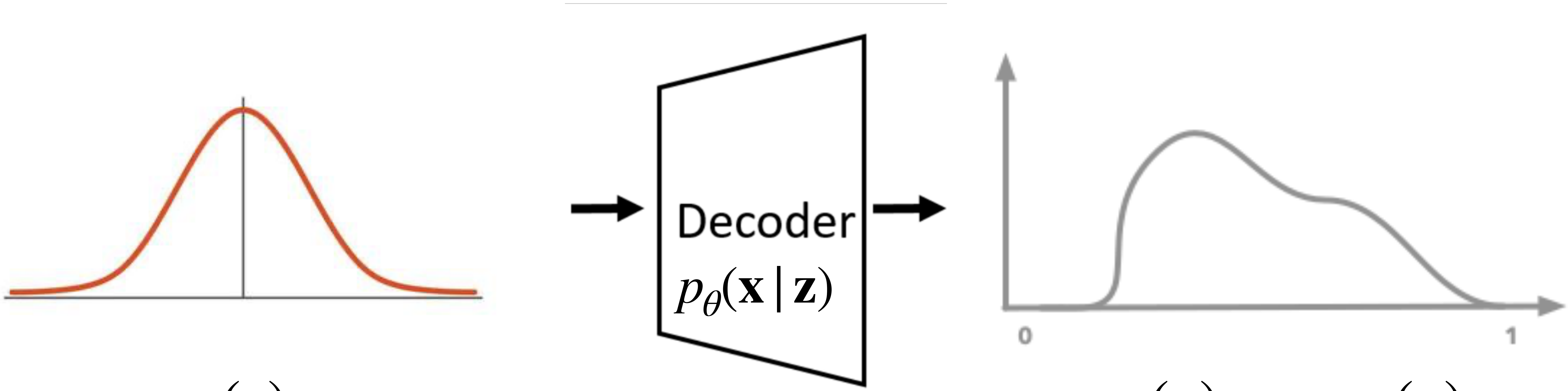
Denoising autoencoders

- A regularization technique to avoid learning trivial solutions.
- **Idea.** Add noise to the input image, an train to recover a clean image
  - Never solved by identity functions
  - Requires understanding how real images look like:
    - Tell apart from the random noise
- **Other examples.** Sparse autoencoders ...



Variational autoencoders

- Takes a similar structure, but quite different from the typical autoencoders
- **Goal.** Train a decoder and a distribution such that:
  - Input. We send in a distribution $p_\theta(\mathbf{z})$
  - Output. We get a data-generating distribution $p_\theta(\mathbf{x}) \approx p_{\text{data}}(\mathbf{x})$



$p_\theta(\mathbf{z})$

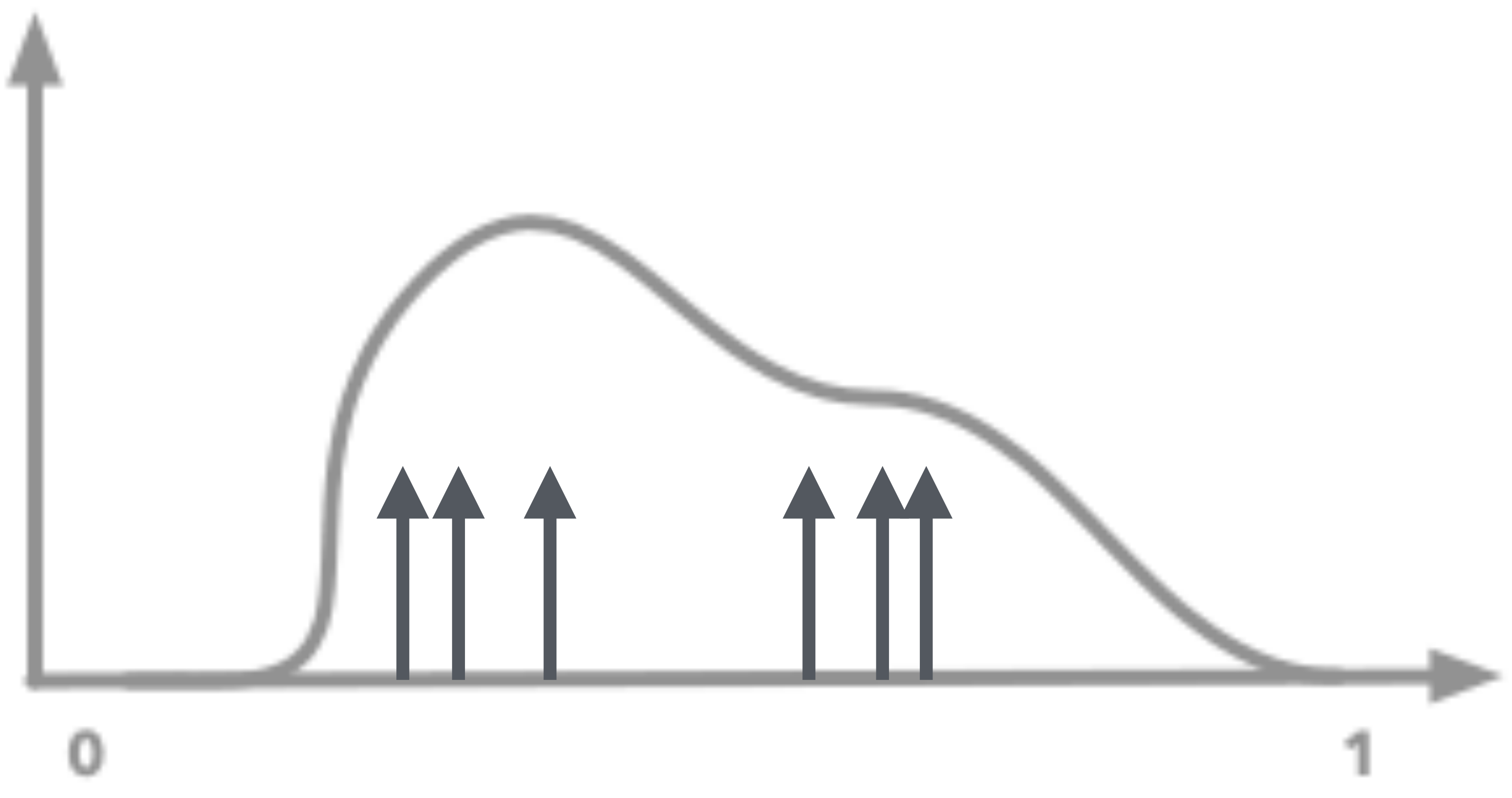
$p_\theta(\mathbf{x}) \approx p_{\text{data}}(\mathbf{x})$

Variational autoencoders

- **Training.** Optimize the log probability

$$\max_{\theta} \sum_{i=1}^{n} \log p_\theta(\mathbf{x}_i) \qquad \Leftrightarrow \qquad \min_{\theta} D(\hat{p} \| p_\theta)$$

  - Equivalent to minimizing some distance measure (called Kullback-Leibler divergence) $D(p \parallel q) = \mathbb{E}_p \log(p/q)$ between the p_θ and the empirical distribution $\hat{p}$



$$p_\theta(\mathbf{x}) \approx \hat{p}(\mathbf{x})$$

Variational autoencoders

- **Training.** Optimize the log probability

$$\max_{\theta} \sum_{i=1}^{n} \log p_\theta(\mathbf{x}_i) \qquad \Leftrightarrow \qquad \min_{\theta} D(\hat{p} \| p_\theta)$$

  - Equivalent to minimizing some distance measure (called Kullback-Leibler divergence) $D(p \,\|\, q) = \mathbb{E}_p \log(p/q)$ between the p_θ and the empirical distribution $\hat{p}$

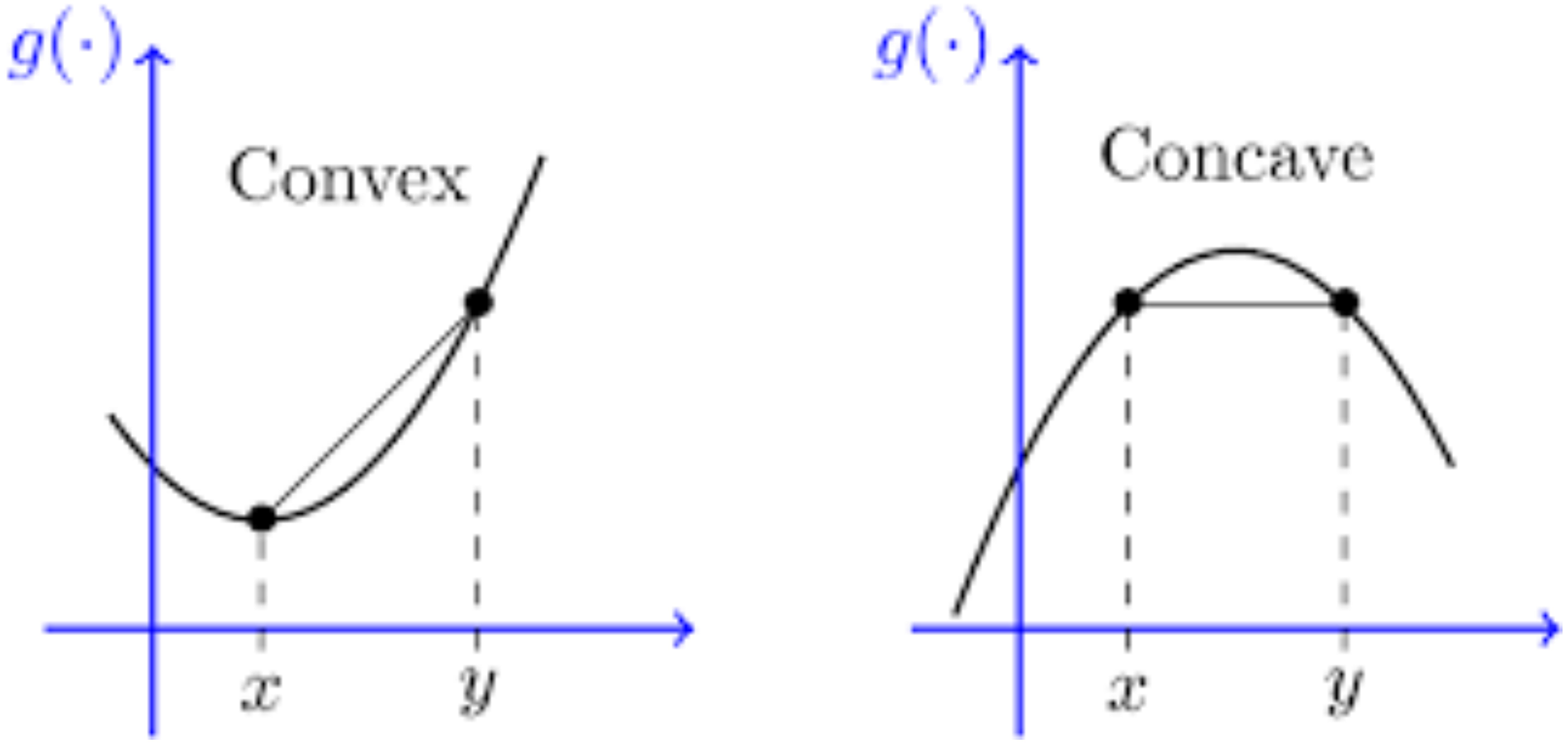
- **Problem.** Computing the marginal distribution is intractible:

$$p_\theta(\mathbf{x}_i) = \int p_\theta(\mathbf{x}_i \,|\, \mathbf{z}) p_\theta(\mathbf{z})\, \mathrm{d}\mathbf{z}$$

  - Idea. We maximize some lower bound of $p_\theta(\mathbf{x})$, not itself
    - Called "evidence lower bound," or simply ELBO

Evidence lower bound

- **Tool.** Jensen's inequality
  - For concave function $f(\cdot)$, we have $\mathbb{E}[f(X)] \leq f(\mathbb{E}[X])$
  - For convex function $f(\cdot)$, we have $\mathbb{E}[f(X)] \geq f(\mathbb{E}[X])$



Evidence lower bound

- For some arbitrary $q_\phi(\mathbf{z})$, we can proceed as:

$$
\begin{aligned}
\log p_\theta(\mathbf{x}) &= \log \int p_\theta(\mathbf{z}) p_\theta(\mathbf{x} \mid \mathbf{z}) \, \mathrm{d}\mathbf{z} \\
&= \log \int q_\phi(\mathbf{z}) \frac{p_\theta(\mathbf{z})}{q_\phi(\mathbf{z})} p_\theta(\mathbf{x} \mid \mathbf{z}) \, \mathrm{d}\mathbf{z} \\
&\geq \int q_\phi(\mathbf{z}) \cdot \log \left[\frac{p_\theta(\mathbf{z})}{q_\phi(\mathbf{z})} p_\theta(\mathbf{x} \mid \mathbf{z}) \right] \, \mathrm{d}\mathbf{z} \\
&= - D(q_\phi(\mathbf{z}) \| p_\theta(\mathbf{z})) + \mathbb{E}_{\mathbf{z} \sim q_\phi}[\log p_\theta(\mathbf{x} \mid \mathbf{z})]
\end{aligned}
$$

Evidence lower bound

- For some arbitrary $q_\phi(\mathbf{z})$, we can proceed as:

$$\begin{aligned}
\log p_\theta(\mathbf{x}) &= \log \int p_\theta(\mathbf{z}) p_\theta(\mathbf{x} \mid \mathbf{z}) \, \mathrm{d}\mathbf{z} \\
&= \log \int q_\phi(\mathbf{z}) \frac{p_\theta(\mathbf{z})}{q_\phi(\mathbf{z})} p_\theta(\mathbf{x} \mid \mathbf{z}) \, \mathrm{d}\mathbf{z} \\
&\geq \int q_\phi(\mathbf{z}) \cdot \log \left[\frac{p_\theta(\mathbf{z})}{q_\phi(\mathbf{z})} p_\theta(\mathbf{x} \mid \mathbf{z}) \right] \, \mathrm{d}\mathbf{z} \\
&= \boxed{- D(q_\phi(\mathbf{z}) \| p_\theta(\mathbf{z})) + \mathbb{E}_{\mathbf{z} \sim q_\phi}[\log p_\theta(\mathbf{x} \mid \mathbf{z})]}
\end{aligned}$$

Sample from $q_\phi(\mathbf{z})$ and measure the loss

- Holds for any $q_\phi(\mathbf{z})$ — takes the maximum lower bound!
  - The optimal $q_\phi(\mathbf{z})$ depends on $\mathbf{x}$... thus write as $q_\phi(\mathbf{z} \mid \mathbf{x})$

Evidence lower bound

- In a nutshell, we are doing

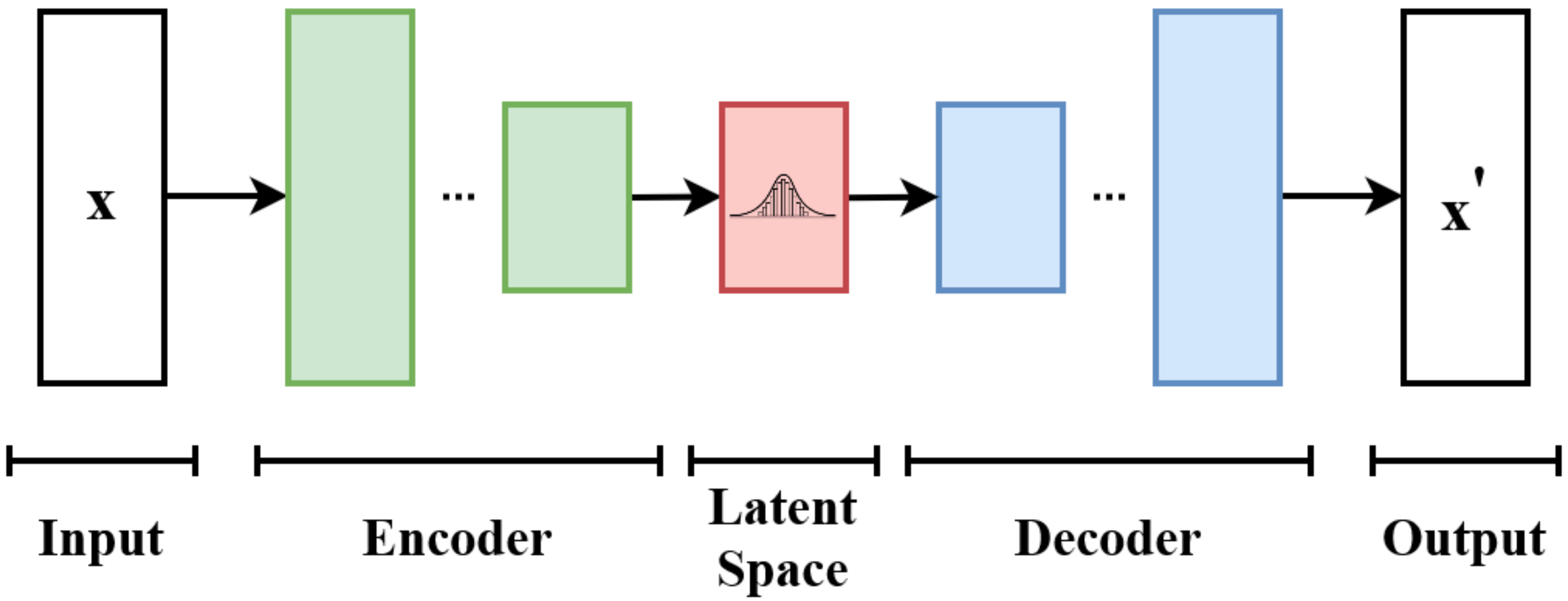
$$\max_{\theta} \log p_\theta(\mathbf{x}_i) \geq \max_{\theta} \max_{\phi} \left(-D(q_\phi(\mathbf{z} \,|\, \mathbf{x}_i) \| p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\cdot|\mathbf{x}_i)}[\log p_\theta(\mathbf{x}_i \,|\, \mathbf{z})] \right)$$

Evidence lower bound

- In a nutshell, we are doing

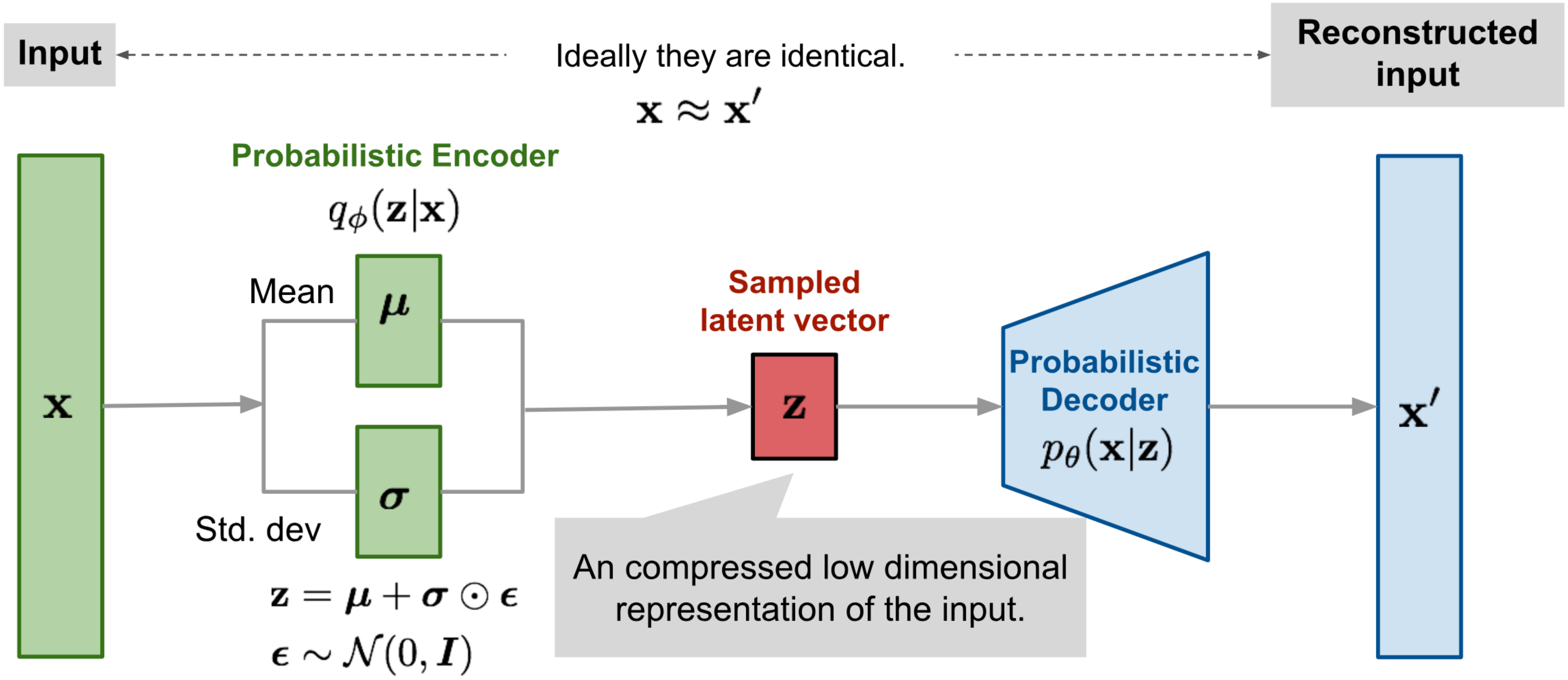
$$\max_{\theta} \log p_\theta(\mathbf{x}_i) \geq \max_{\theta} \max_{\phi} \left(-D(q_\phi(\mathbf{z}\,|\,\mathbf{x}_i) \| p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\cdot|\mathbf{x}_i)}[\log p_\theta(\mathbf{x}_i\,|\,\mathbf{z})] \right)$$

- **Question.** How do we model $q_\phi(\mathbf{z}\,|\,\mathbf{x})$?
  - Answer. Jointly train a probabilistic encoder that expresses $q_\phi(\mathbf{z}\,|\,\mathbf{x})$
  - Question. How do we implement a probabilistic encoder?



Reparametrization Trick

- **Idea (Reparametrization Trick).** Model $q_\phi(\mathbf{z} \mid \mathbf{x})$ as a conditional Gaussian $\mathcal{N}(\mu_\mathbf{x}, \sigma_\mathbf{x}^2)$
  - $\mu_\mathbf{x}, \sigma_\mathbf{x}$ are learned with a neural network, instead.



Reparametrization Trick

- **Idea (Reparametrization Trick).** Model $q_\phi(\mathbf{z} \mid \mathbf{x})$ as a conditional Gaussian $\mathcal{N}(\mu_\mathbf{x}, \sigma_\mathbf{x}^2)$
  - $\mu_\mathbf{x}, \sigma_\mathbf{x}$ are learned with a neural network, instead
- Now. look at the optimization problem

$$\max_\theta \max_\phi \left(-D(q_\phi(\mathbf{z} \mid \mathbf{x}_i) \| p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\cdot \mid \mathbf{x}_i)}[\log p_\theta(\mathbf{x}_i \mid \mathbf{z})] \right)$$

- Let us look at the 2nd term, and then the 1st term:
  - If we use the model $p_\theta(\mathbf{x}_i \mid \mathbf{z}) = \mathcal{N}(f_\theta(\mathbf{z}), \eta \cdot I_d)$, the 2nd term becomes

$$-\mathbb{E}_{q_\phi(\cdot \mid \mathbf{x}_i)} \left[\frac{1}{2\eta} \|\mathbf{x}_i - f_\theta(\mathbf{z}_i)\|^2 \right] + \text{const}.$$

    - That is, simply use the squared loss!
      (a bit more complicated if variances are also trained for each dimension)

Reparametrization Trick

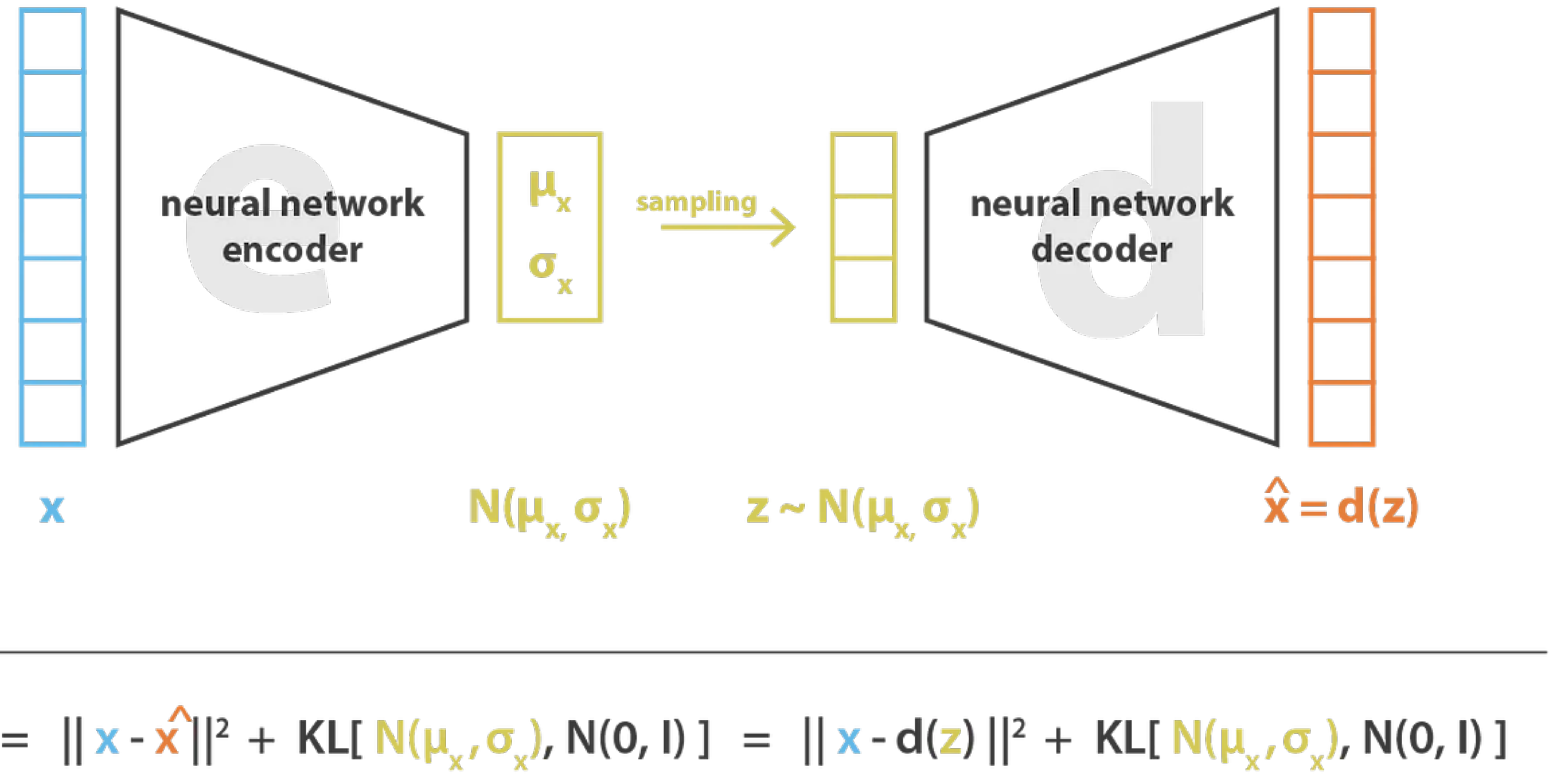
$$\max_{\theta} \max_{\phi} \left(-D(q_\phi(\mathbf{z}\,|\,\mathbf{x}_i) \| p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\cdot|\mathbf{x}_i)}[\log p_\theta(\mathbf{x}_i\,|\,\mathbf{z})] \right)$$

- If we use the Gaussian encoder

$$q_\phi = \mathcal{N}(\mu_{\mathbf{x}_i}, \sigma_{\mathbf{x}_i} \cdot I_k)$$

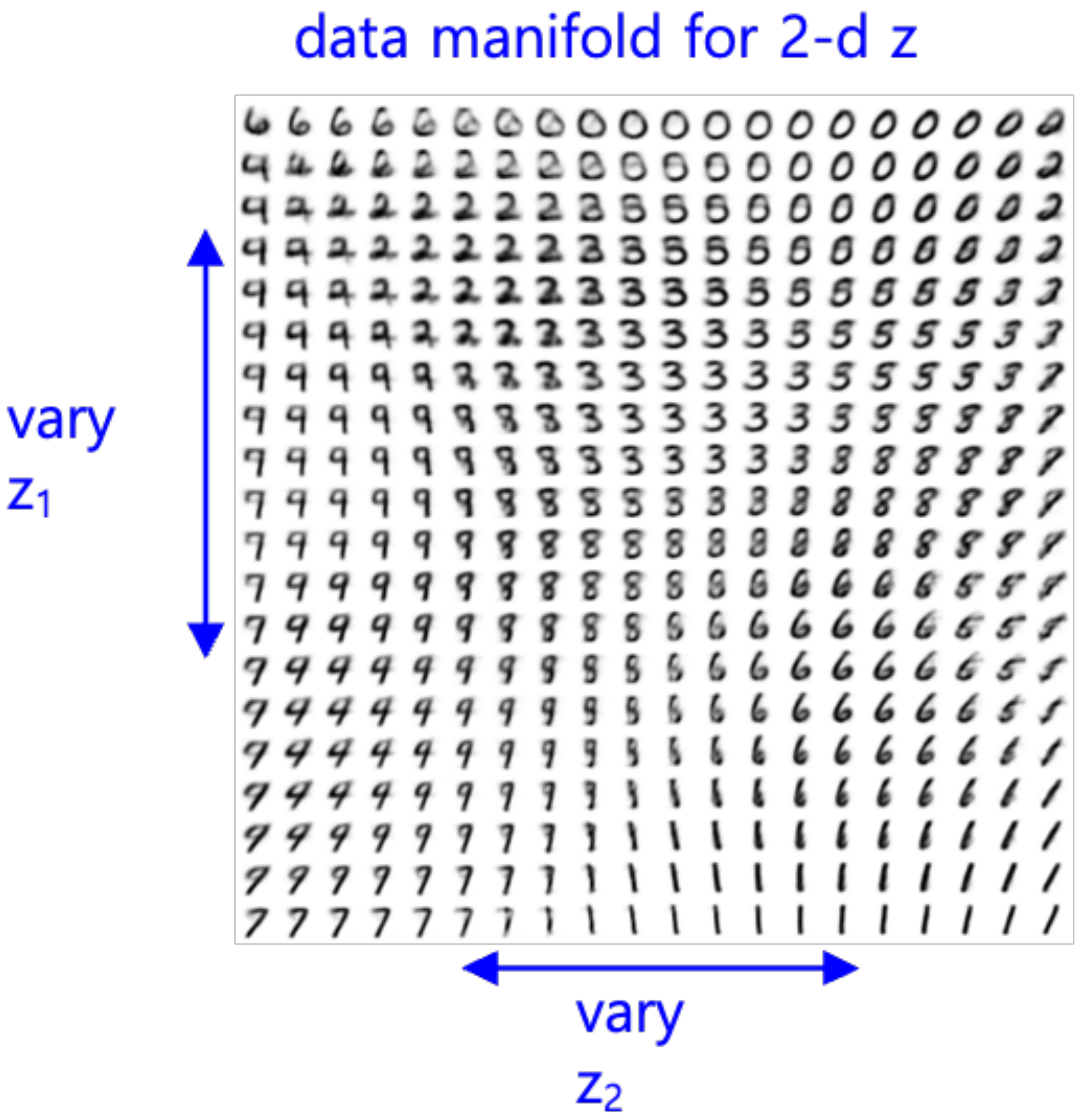
  then this simply becomes the squared regularizers on μ and $\boldsymbol{\sigma}$
  - Check by yourself!

- Thus, a squared loss and a squared regularizer



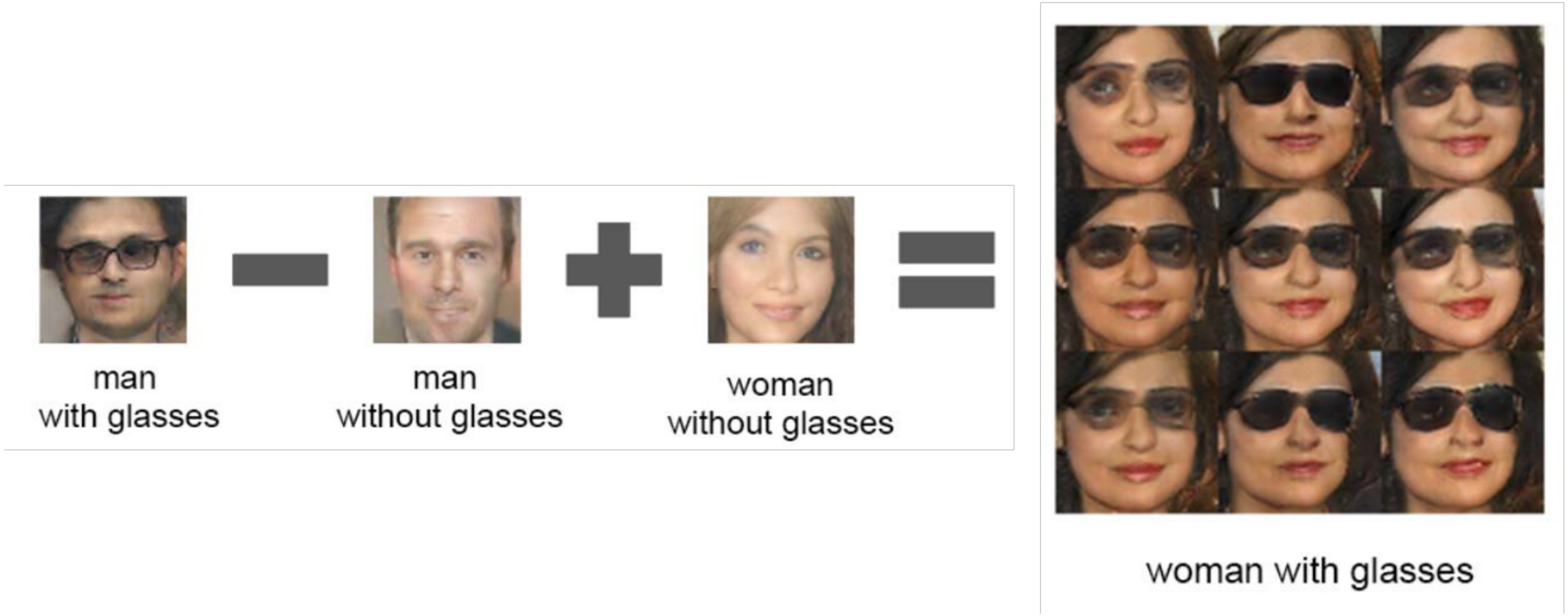
Properties

- **Pros.** Known to enjoy nice disentanglement



Properties

- **Pros.** Known to enjoy nice disentanglement



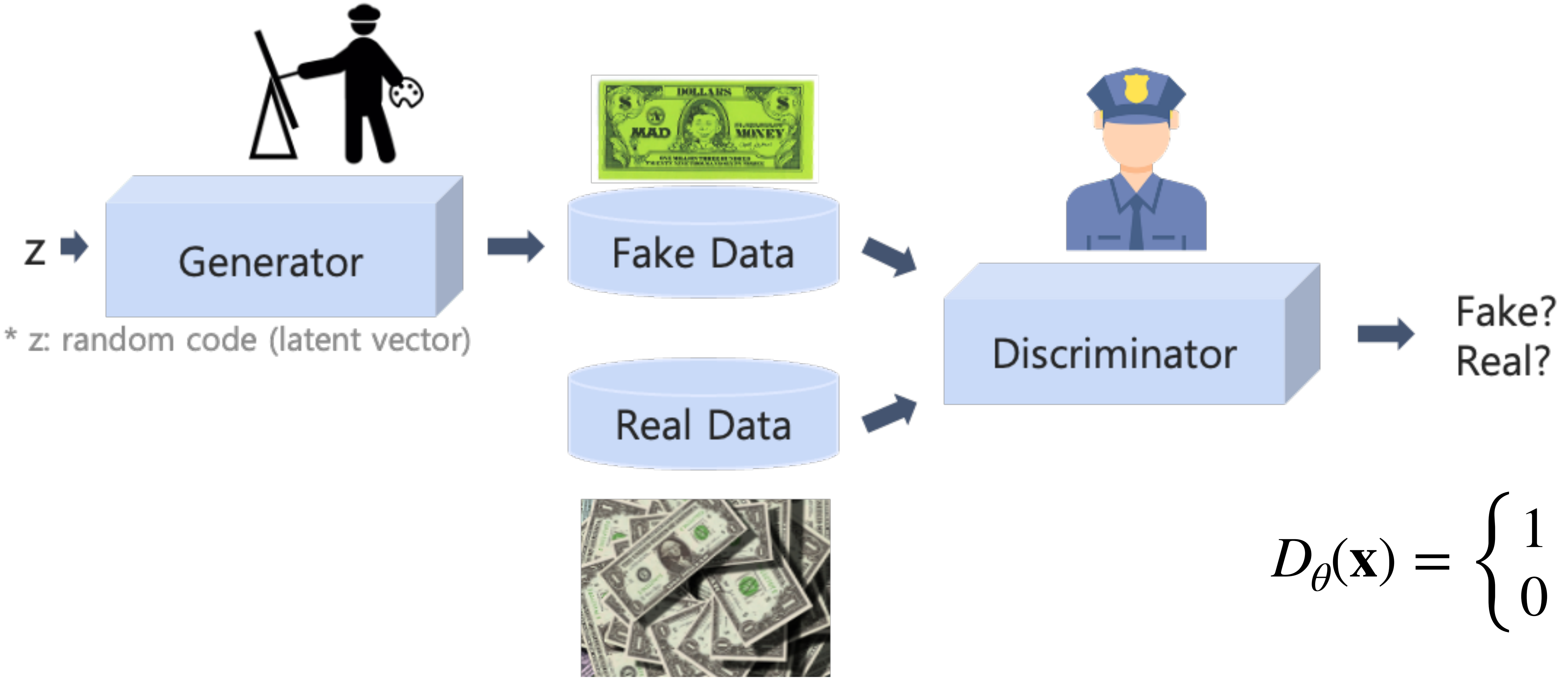
Properties

- **Pros.** Known to enjoy nice disentanglement
- **Cons.** Known to be less "sharp," with much noises
  - Clearly distinguishable from the real images
    (Take these with a grain of salt, as technologies advance fast!)

Generative Adversarial Nets

Generative Adversarial Nets

- **Idea.** Train explicitly for “hard to distinguish” properties
  - View generative process as a two-player game
    - <u>Generator</u>. Tries to fool the discriminator
    - <u>Discriminator</u>. 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

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

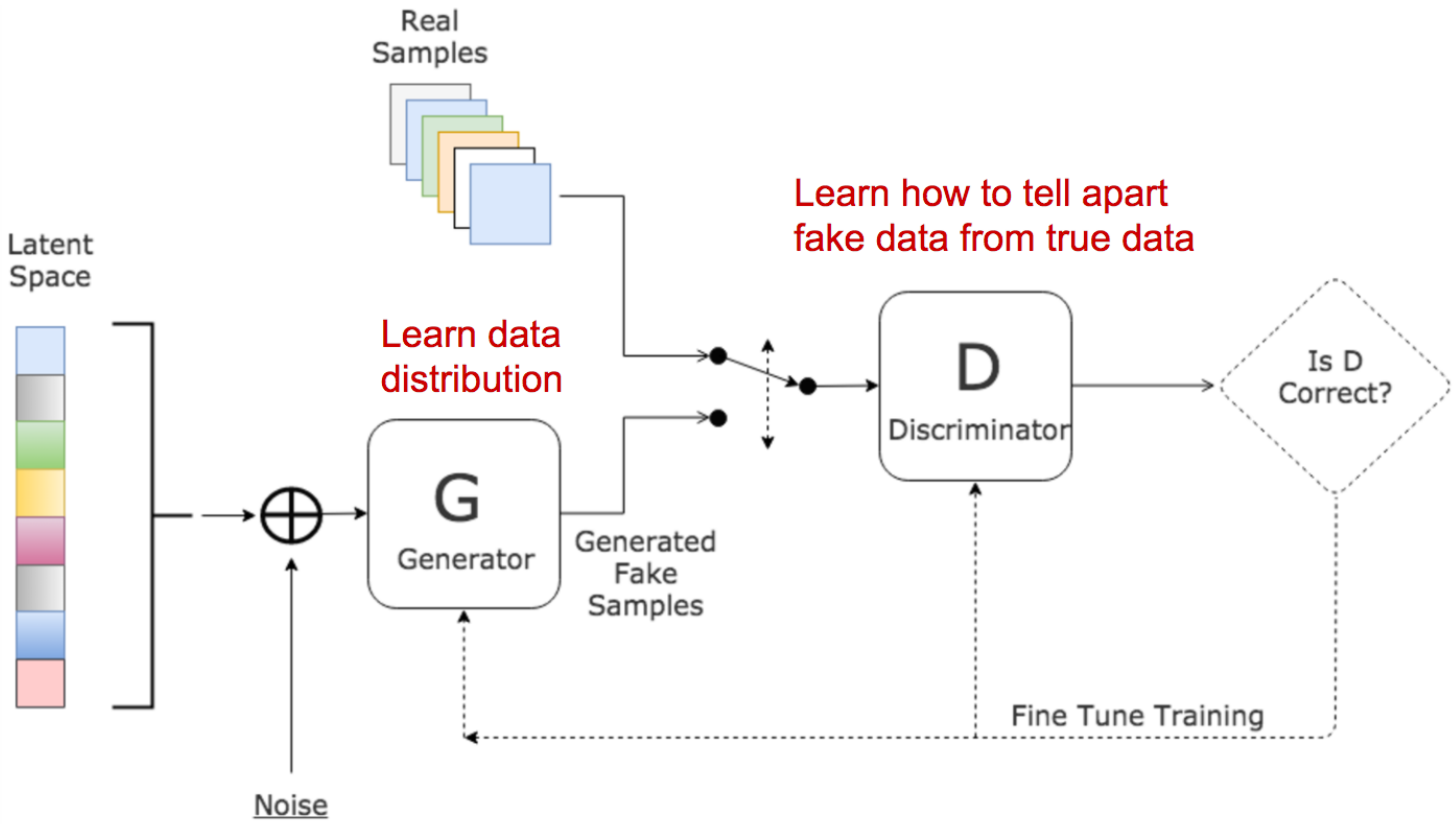
Discriminator declares real image to be real

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

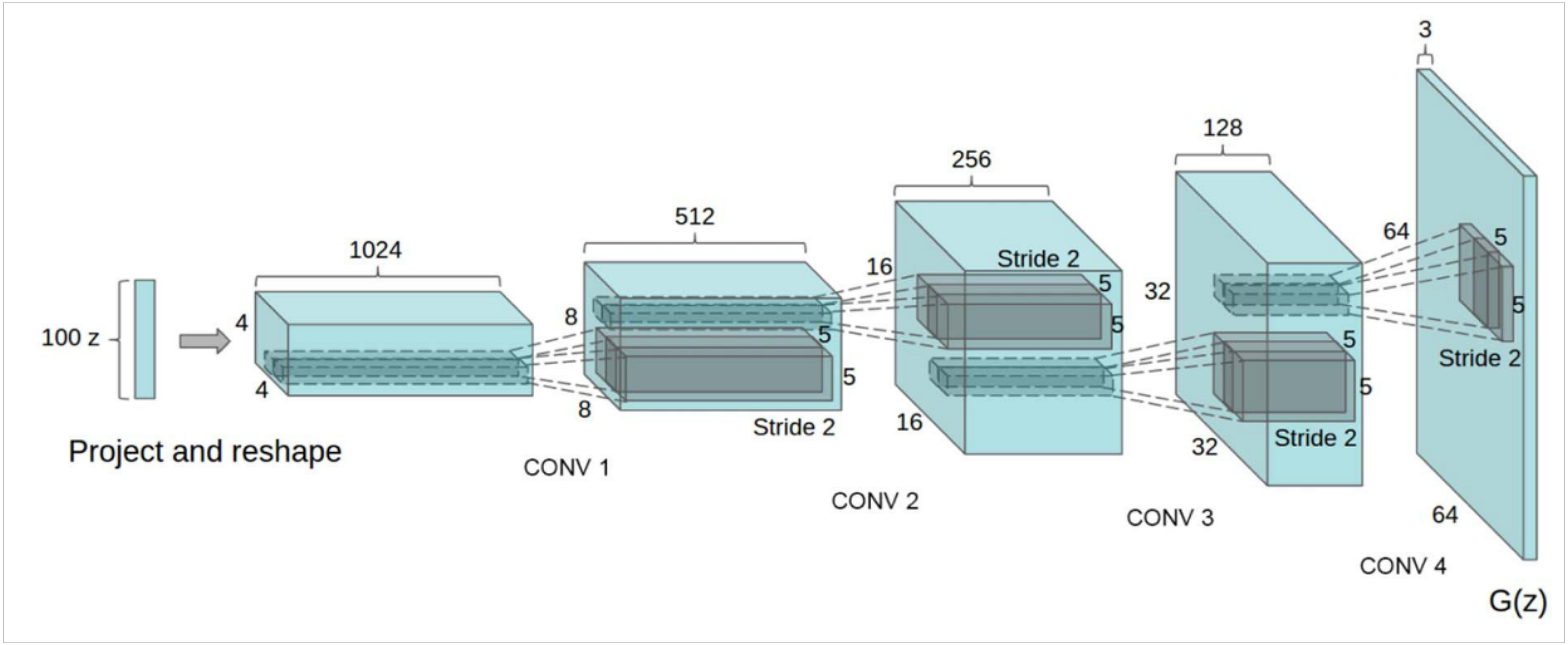
$$D\left(p_\theta \middle\| \frac{\hat{p} + p_\theta}{2}\right) + D\left(\hat{p} \middle\| \frac{\hat{p} + p_\theta}{2}\right)$$

Generative Adversarial Nets



Generative Adversarial Nets

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

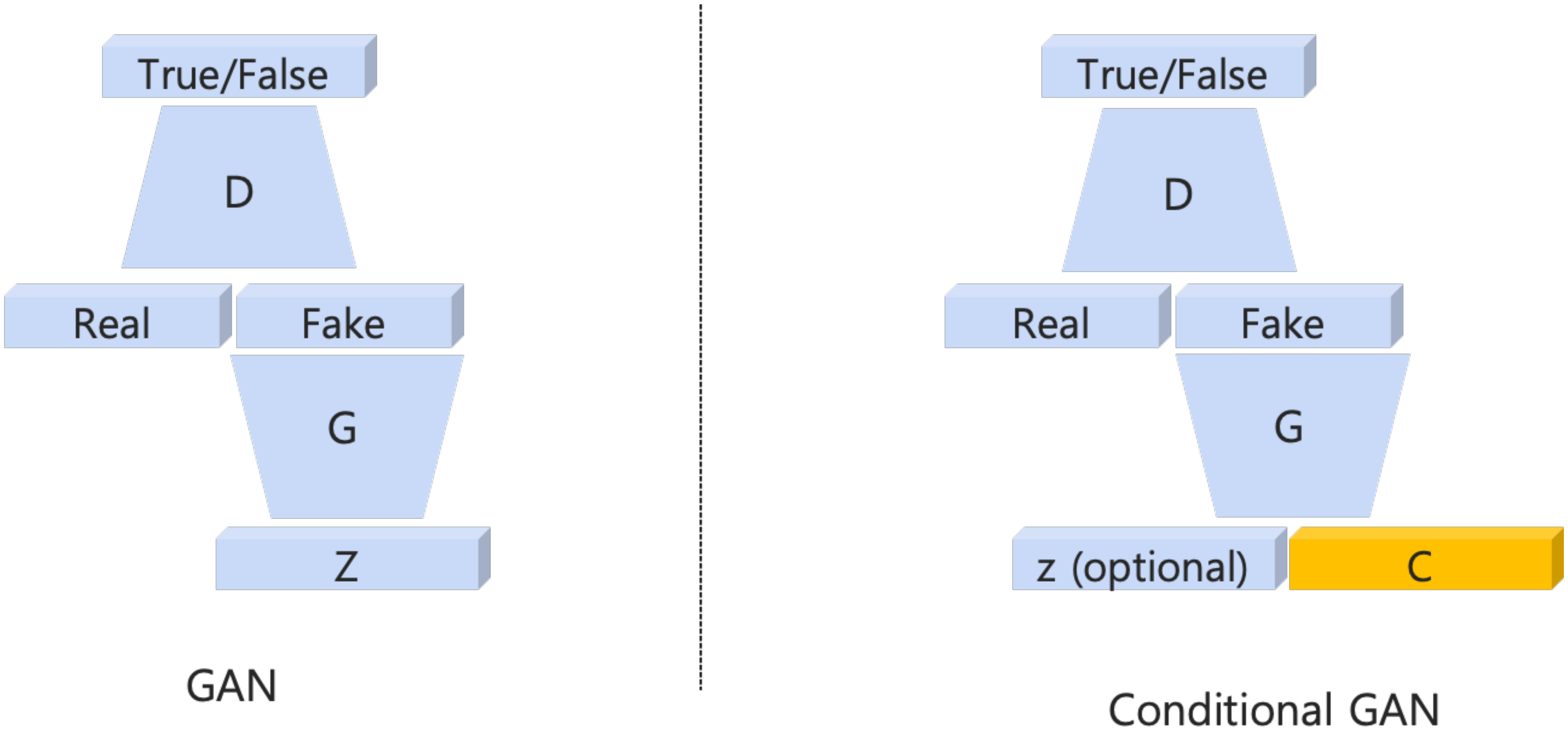


Results

- Such training can give very sharp images

Conditional GAN

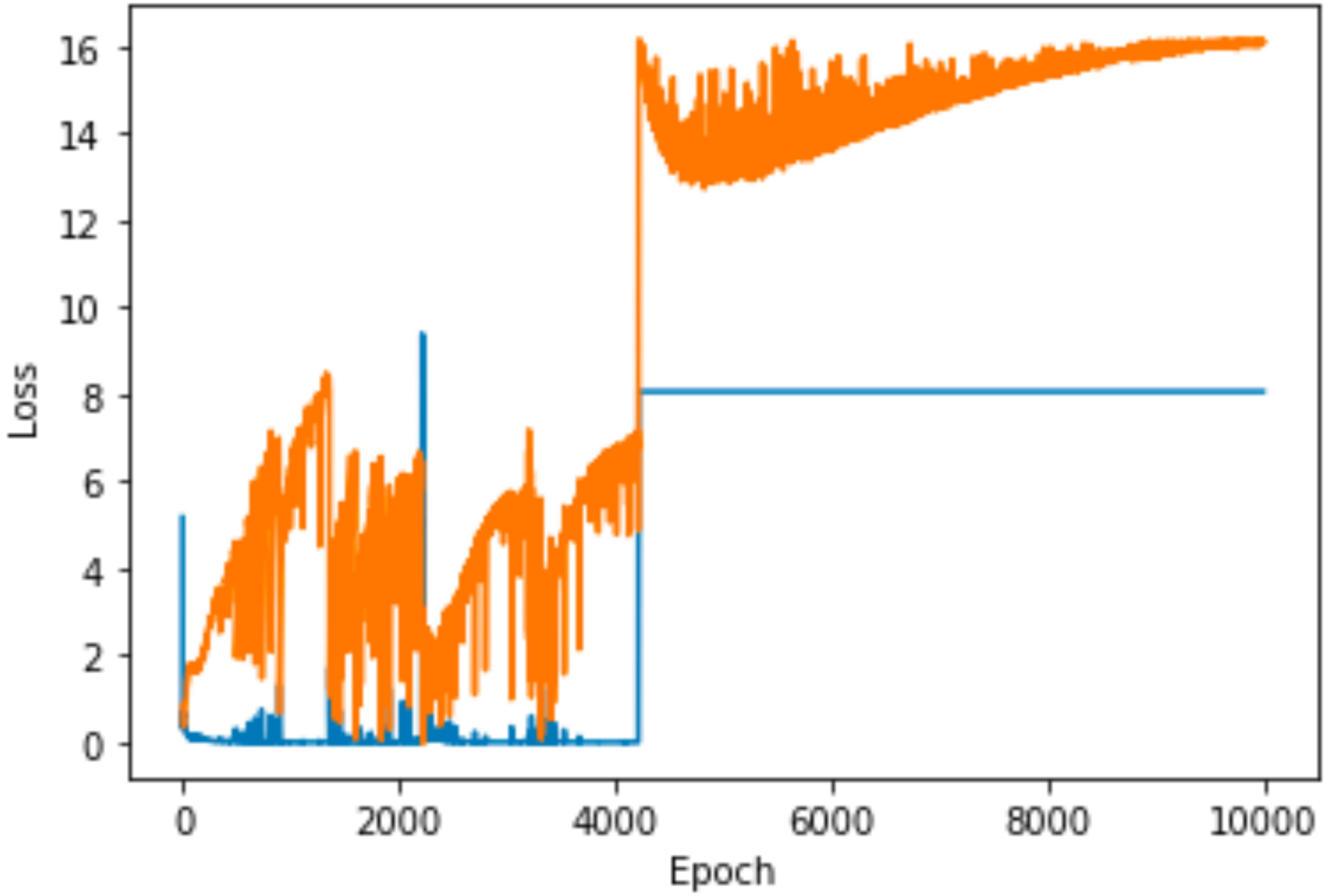
- **Idea.** Add class/text information to the latent code
  - Generate realistic images under specific conditions



Conditional GAN

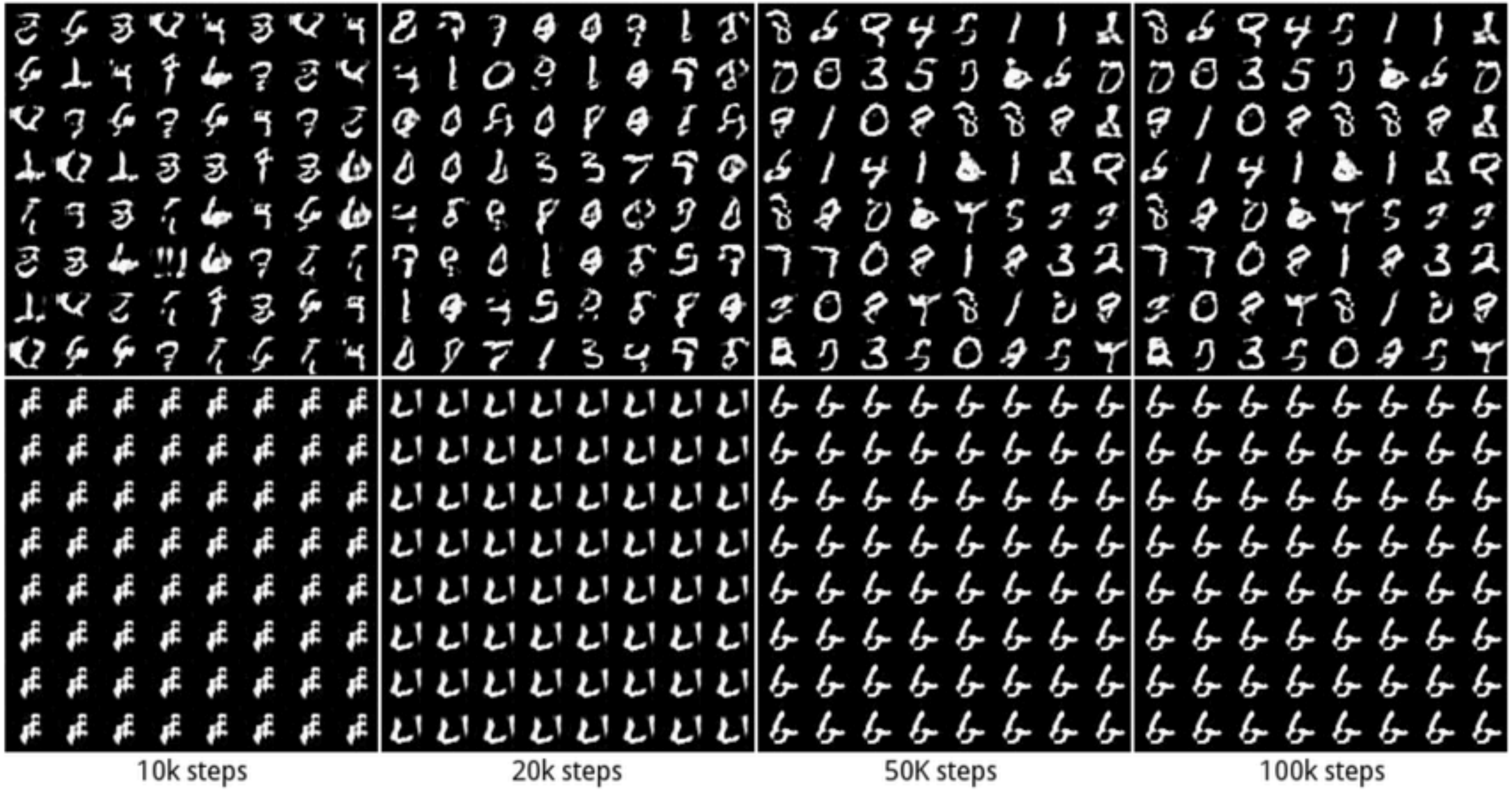
Pitfalls

- 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”



Next class

- Diffusion model

Cheers