MACHINE LEARNING

Generative Models

What is a Generative Model?

- Approach for unsupervised learning analysis of data
- Model that learn a simulator of data - a **source** that produce data
- Model that allow for density estimation

Techniques for **Generative Modeling**

- **Explicit density estimation**: Directly estimating the distribution of training data.
- **Implicit density estimation**: Indirect estimation of the data distribution through direct learning of the data generation mechanism.

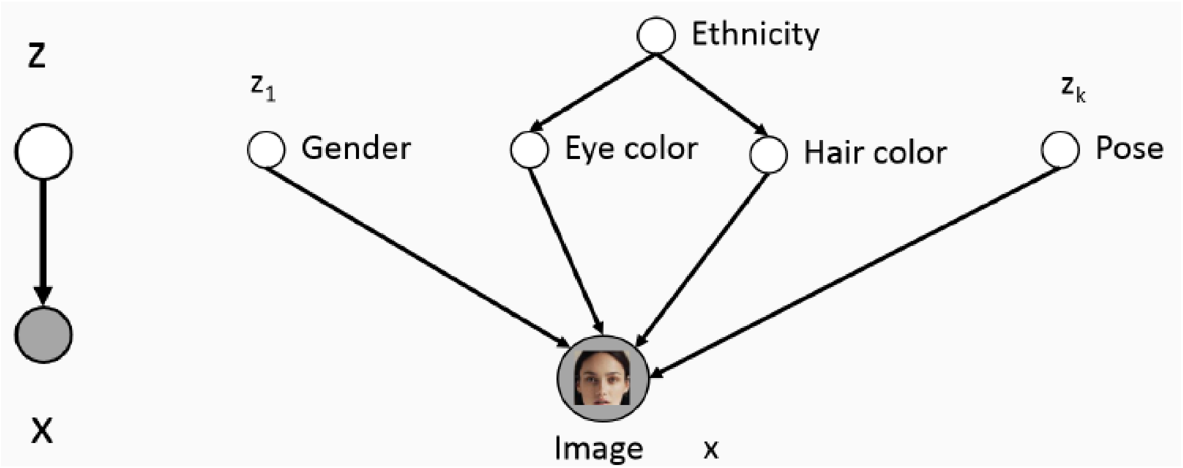
Deep Generative Models

- **Task**: Given a dataset of samples $\boldsymbol{D} = \{x_1, \dots, x_N\}$ find the underlying (*unknown*) data **distribution *p(x)***

- **Goal:** Approximate the true data distribution *p(x)* with a parameterized **Neural Network** $p_\theta(x)$ using the ***D***

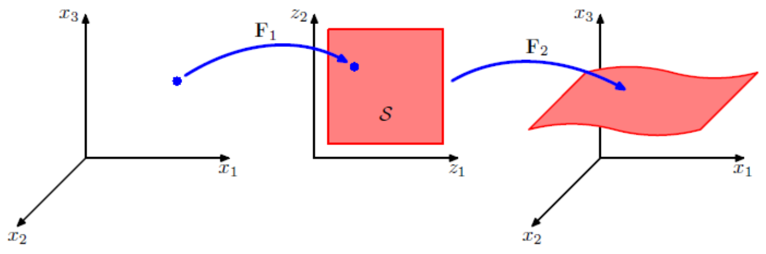
- **Learning problem**: Find the (*unknown*) **parameters** θ of the Neural Network that **best fit** the given input data D

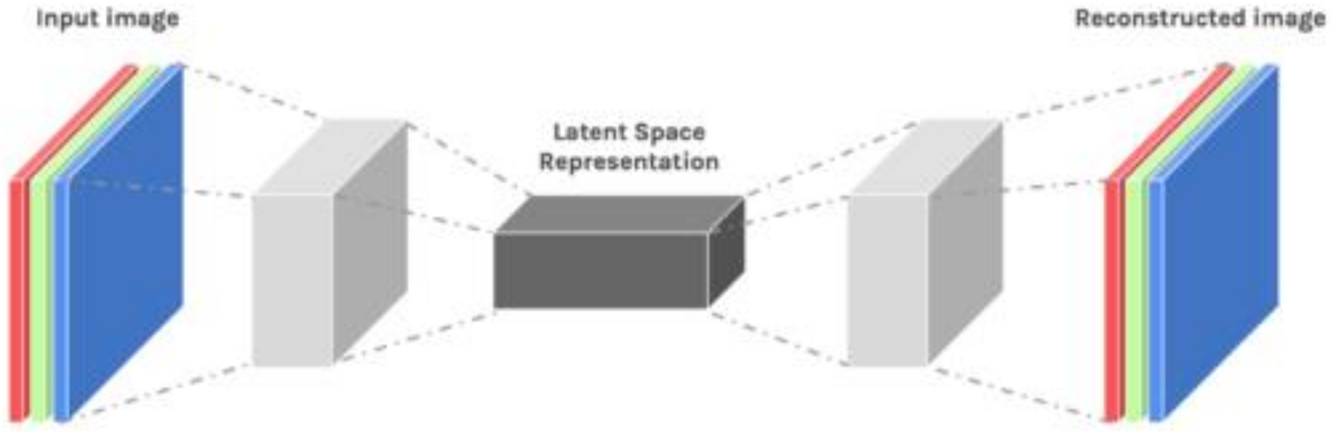
Latent Variable Models



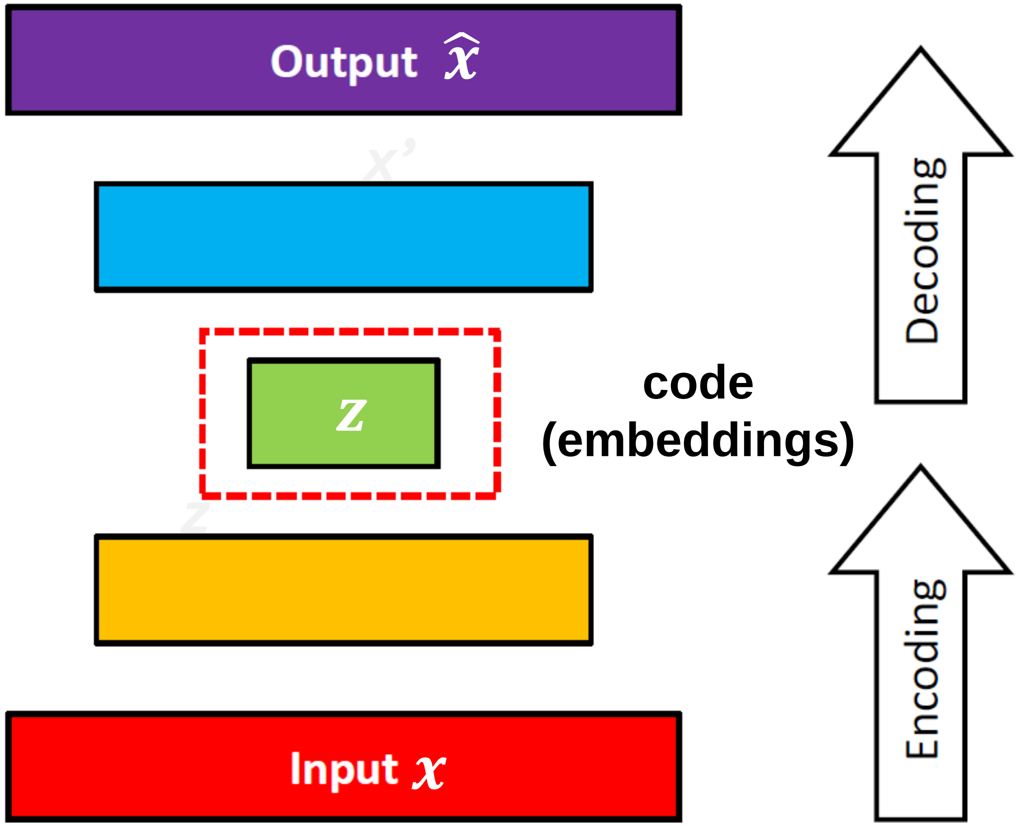
- Only variables x are observed in the data
- **Hypothesis**: Existence of latent variables **z** that correspond to high level features
  - If **z** can be found, $p(\boldsymbol{x}|\boldsymbol{z})$ could be much simpler that $p(\boldsymbol{x})$
  - If we train this model, then we can identify features via $p(\boldsymbol{z}|\boldsymbol{x})$

Autoencoders





- **Auto-associative** neural network
- A neural network with identical input and output
- Perform dimensionality reduction and data compression using a smaller number of hidden neurons than the input
- The hidden layer captures the compressed **latent coding**



Autoencoder: a two-parts neural network structure

- **Encoder** (or *recognition network*): compress the input and converts it to a latent representation (**code**), $\boldsymbol{z} = f(x)$
- **Decoder** (or *generation network*): regenerates the input, converts the internal representation to the outputs $\hat{\boldsymbol{x}} = g(\boldsymbol{z})$

- The encoder compresses the input, and the decoder (*conditionally*) reproduces it

- **Learning** is achieved by minimizing the **reconstruction error**, which is the *Loss function*:

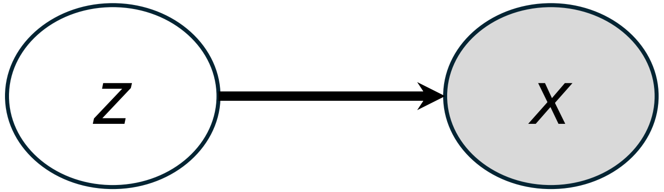
$$\boldsymbol{L}\Big(x, g\big(f(x)\big)\Big) = \boldsymbol{L}\big(x, g_w(x)\big)$$

- *Mean squared error*, ***MSE*** is a common loss function:

$$\boldsymbol{L}(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N} \|x_i - \underbrace{\boldsymbol{g_w(x_i)}}_{\hat{\boldsymbol{x}}_i}\|^2$$

Stochastic Autoencoders

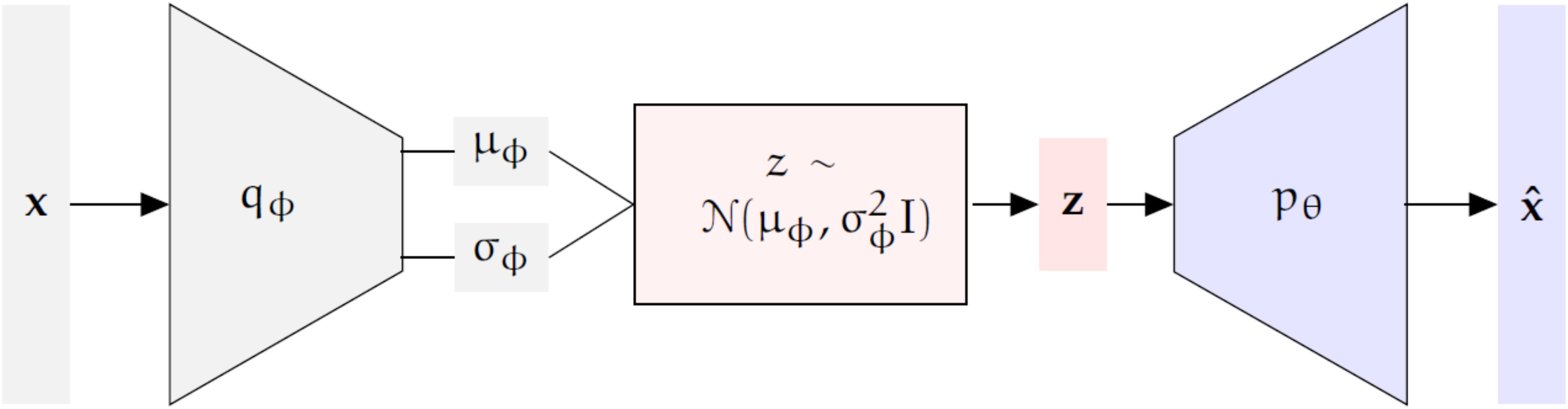
- Consider Autoencoders as ***Generative Models***
- **Goal**: capture the distribution of observed data
  - Introduce **latent variables** $z \sim p(z)$ (typically of lower dimension), which are responsible to generate the observed data.



- **Idea**: model the joint distribution $p(x, \boldsymbol{z})$ and integrate out the latent variable $\boldsymbol{z}$ to obtain the marginal distribution of data $p(x)$:

$$p_\theta(x) = \int_z p(x,z)dz = \int_z \boldsymbol{p_\theta(x|z)} p_z(z)dz$$

Variational Autoencoders (VAE)

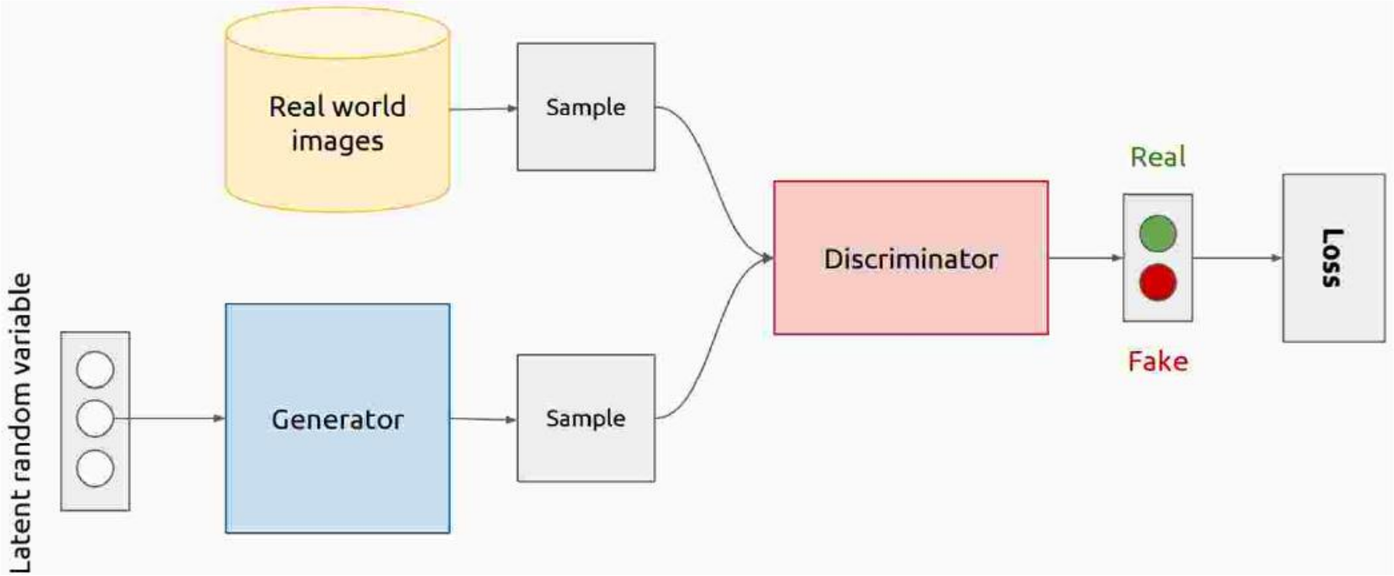


- The distribution of latent variables $p(z) = \boldsymbol{q(z|x)}$ is chosen to be *Gaussian* with parameters ***μ, σ***

Stages

1. The **Encoder** produces the **mean** (μ) and **standard deviation** (σ).
2. Normal distribution ***N****(μ, σ^2)* is used to **produce** a sample latent vector ***z***
3. This becomes input to the **Decoder network** for reproducing input

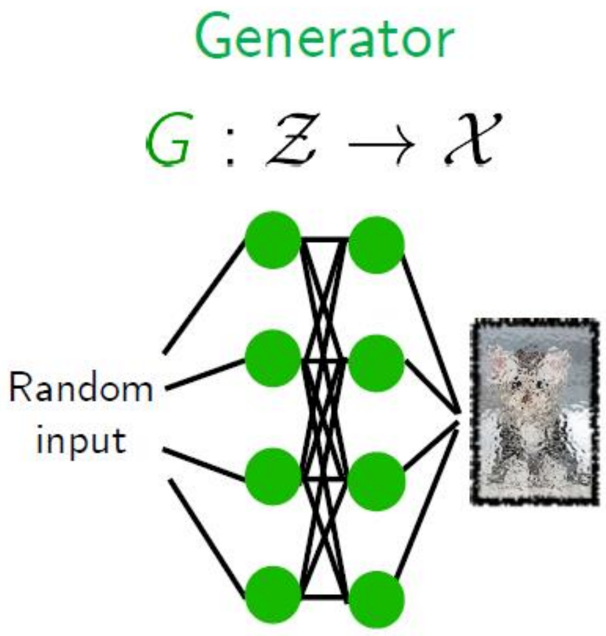
Generative Adversarial Networks (**GANs**) [Goodfellow et. al. 2014]

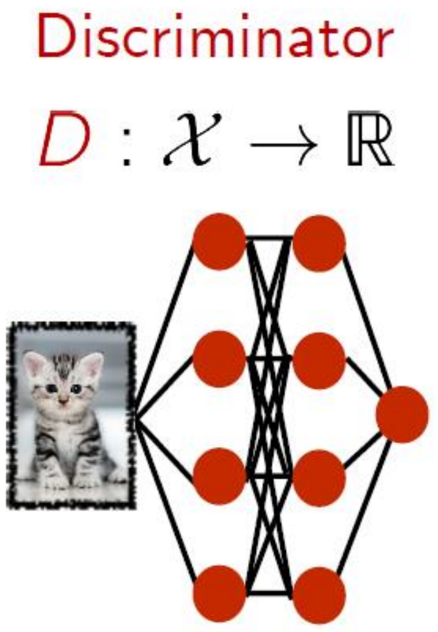


GAN structure

- Generative model consists of two neural networks that compete each other
  - Make a sampling through $p(\boldsymbol{z})$ and map it using a Deep Generator net to $\boldsymbol{x} = G_\theta(\boldsymbol{z})$
  - Instead of evaluating $p_\theta(\boldsymbol{x})$, use a **classifier** $D_\varphi(\boldsymbol{x})$ to decide if it is ***real*** or ***fake***

Generative Adversarial Networks (GAN)



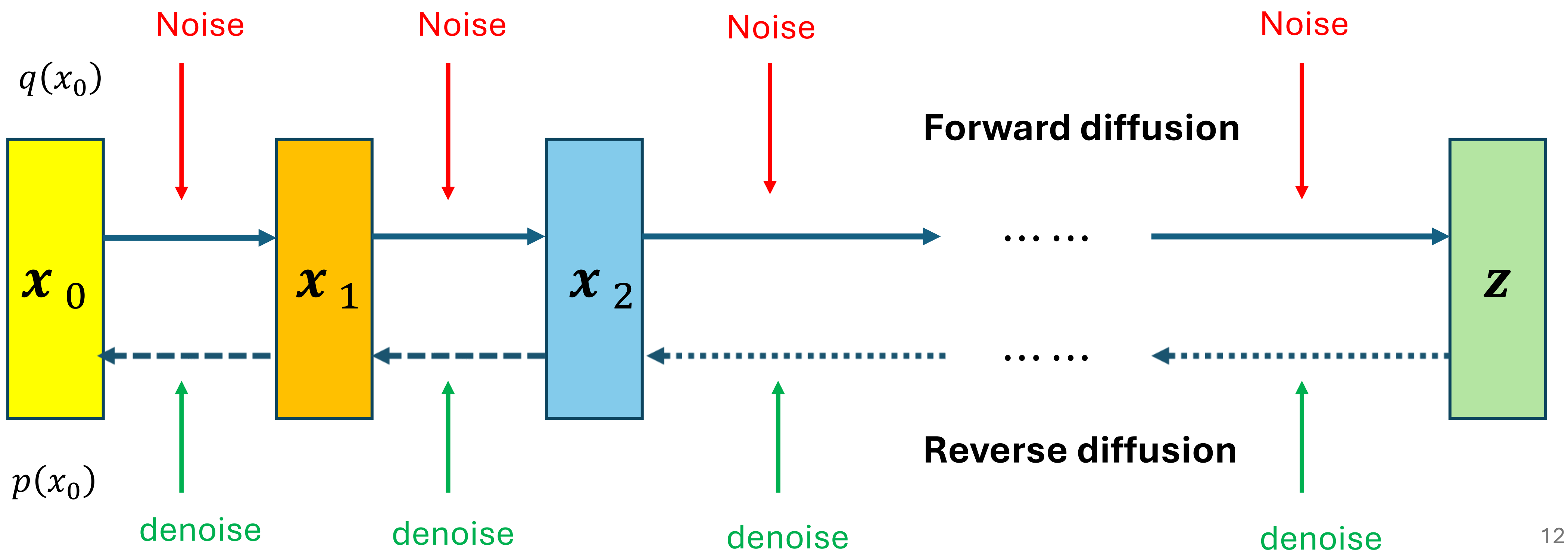


Two collaborative and competitive neural networks:

- **Discriminator** tries to distinguish real from fake data created by the **Generator**
- **Generator** turns **random noise** into imitations of the data, in an attempt to fool the Discriminator by creating more realistic samples

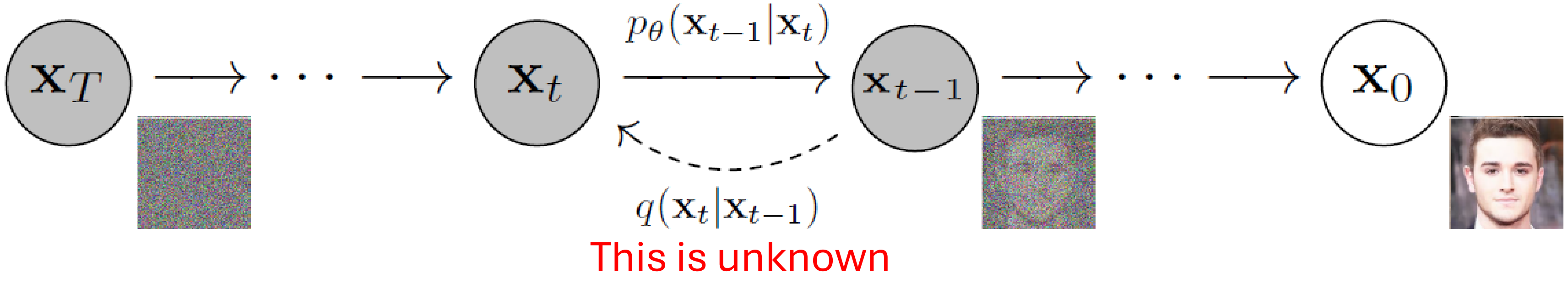
Diffusion Models

- **Idea**: Estimate and analyze small step sizes (instead of a single step) that gradually inserts noise to data using a Markov chain $q(x_0,x_1,\ldots,x_N)=q(x_0)q(x_1|x_0)\ldots q(x_N|x_{N-1})$ until reaching a final latent space that is a standard Gaussian, i.e. **noise**



Reverse diffusion: remove noise

- Reverse diffusion is a denoising process



- **Goal of diffusion model** is to learn the reverse denoising process using information from the forward process
- In this way, the reverse process can be used as a **generative model** of new data from random noise!
- $p_\theta(x_{t-1}|x_t)$ modeled as $\mathcal{N}(x_{t-1}|\mu_\theta(x_t), \Sigma_\theta(x_t))$ where μ_θ and Σ_θ are **neural networks** with θ parameters