Bits of Vision:
Generative Modeling - 2

Recap

- **Last class.** Generative model for images
  - VAE (Variational Autonencoder)
  - GAN (Generative Adversarial Net)

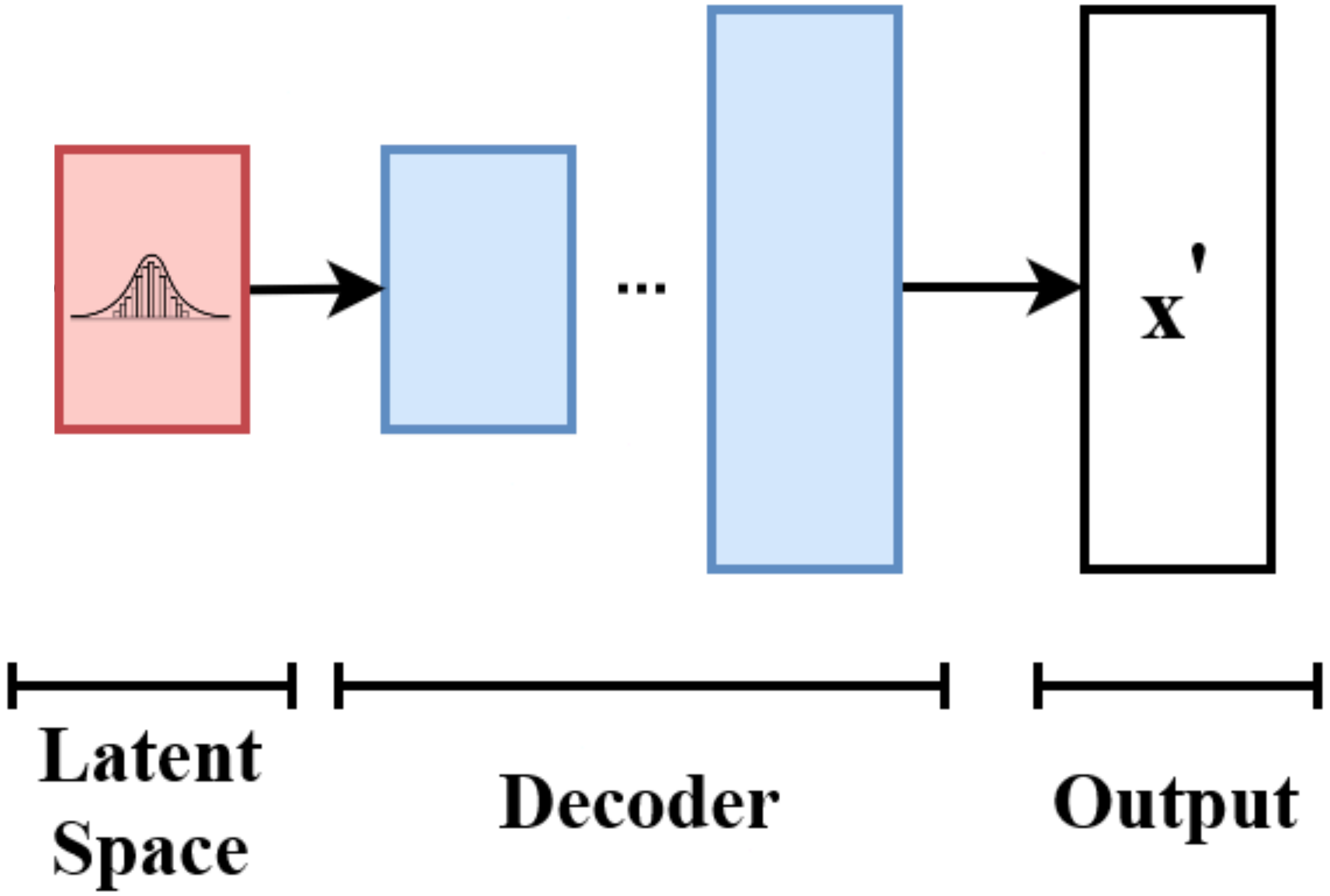
- **Today.** Diffusion model

Recap: VAE

- In VAE, the decoder $p_\theta(\mathbf{x} \mid \mathbf{z})$ generates a samples from a random latent code $\mathbf{z} \sim \mathcal{N}(0, I_k)$, such that

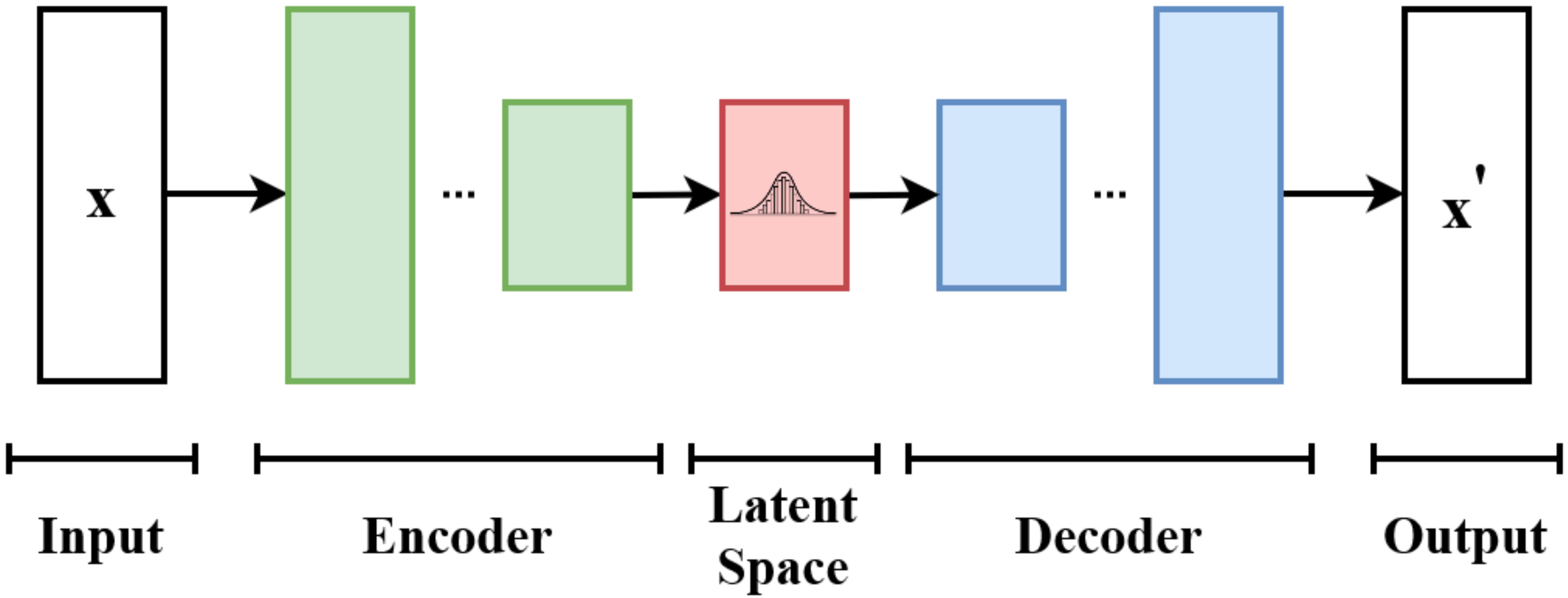
$$p_{\text{data}}(\mathbf{x}) \approx p_\theta(\mathbf{x})$$

- **Problem.** For training such a model, we need a good inverse map



Recap: VAE

- The solution was to jointly train an encoder which generates Gaussian from inputs
- **Problem.** As the “distribution of images” is extremely complicated,
  a single forward of neural network may not have a sufficient
  capacity to do this…



Diffusion model

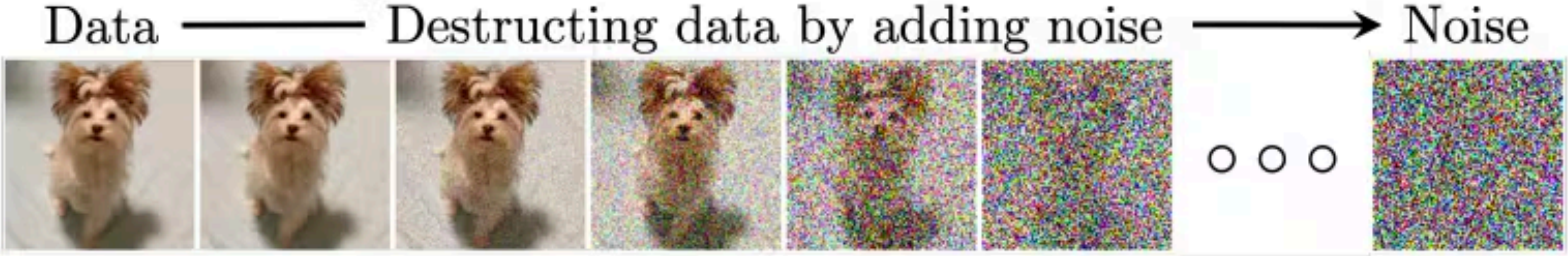
- **Observation.** We can also generate Gaussian-like latent codes from the input (i.e., encode), via gradually adding Gaussian noise to the input
  - i.e., sample the data $\mathbf{x}_t$ from the distribution

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

  - i.e., mix with the Gaussian noise:

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

(scaling factors for preserving the ℓ_2 norm)

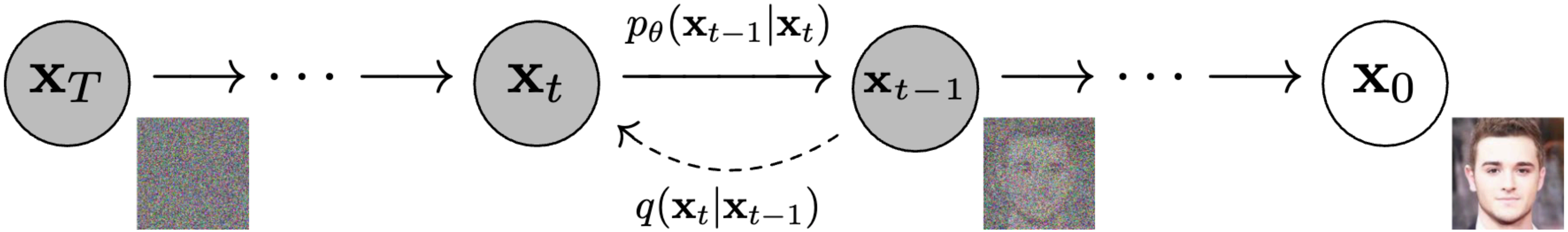


Diffusion model

- Use this noise-adding as our probabilistic encoder!
    - How do we train the corresponding decoder?

- **Idea.** Train a step-by-step model $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ which approximates the posterior of the noise addition, i.e., $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$
    - Parameterized as Gaussian, with trainable mean & variance

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



Training

- Draw a 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, 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

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

Training

- To evaluate the log probability, we use ELBO (math alert!)
  - Use Jensen's inequality to get:

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

Training

- Decomposing further, we get

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

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

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

  - Step-by-step operations, for each $\mathbf{x}_0, \ldots, \mathbf{x}_T$

Training

- We apply an additional conditioning

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

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

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

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

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

Training

- Then proceed with decomposition

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

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

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

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

Training

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

- **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.** Does not have any learnable parameter
  - Thus, ignore

Training

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

- **Third term.** First, we look at the LHS of the KL divergence
  - If we have the relationships $\# \; \bar{\alpha}_i := \alpha_1 \cdot \alpha_2 \cdot \cdots \cdot \alpha_i$

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon, \qquad \mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\epsilon'$$

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

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

Training

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

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

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

- 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

- Training the reverse diffusion process is simply:
    - Sample an image $\mathbf{x}_0$ from the dataset
    - Using $q(\,\cdot\,)$, sample $\mathbf{x}_1, \ldots, \mathbf{x}_T$.
    - 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

- This is typically reparametrized as a noise prediction
  - i.e., predict the residual of the prediction
  - Recent works suggest that this reparametrization may be flawed… (https://arxiv.org/abs/2511.13720)

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

In a nutshell

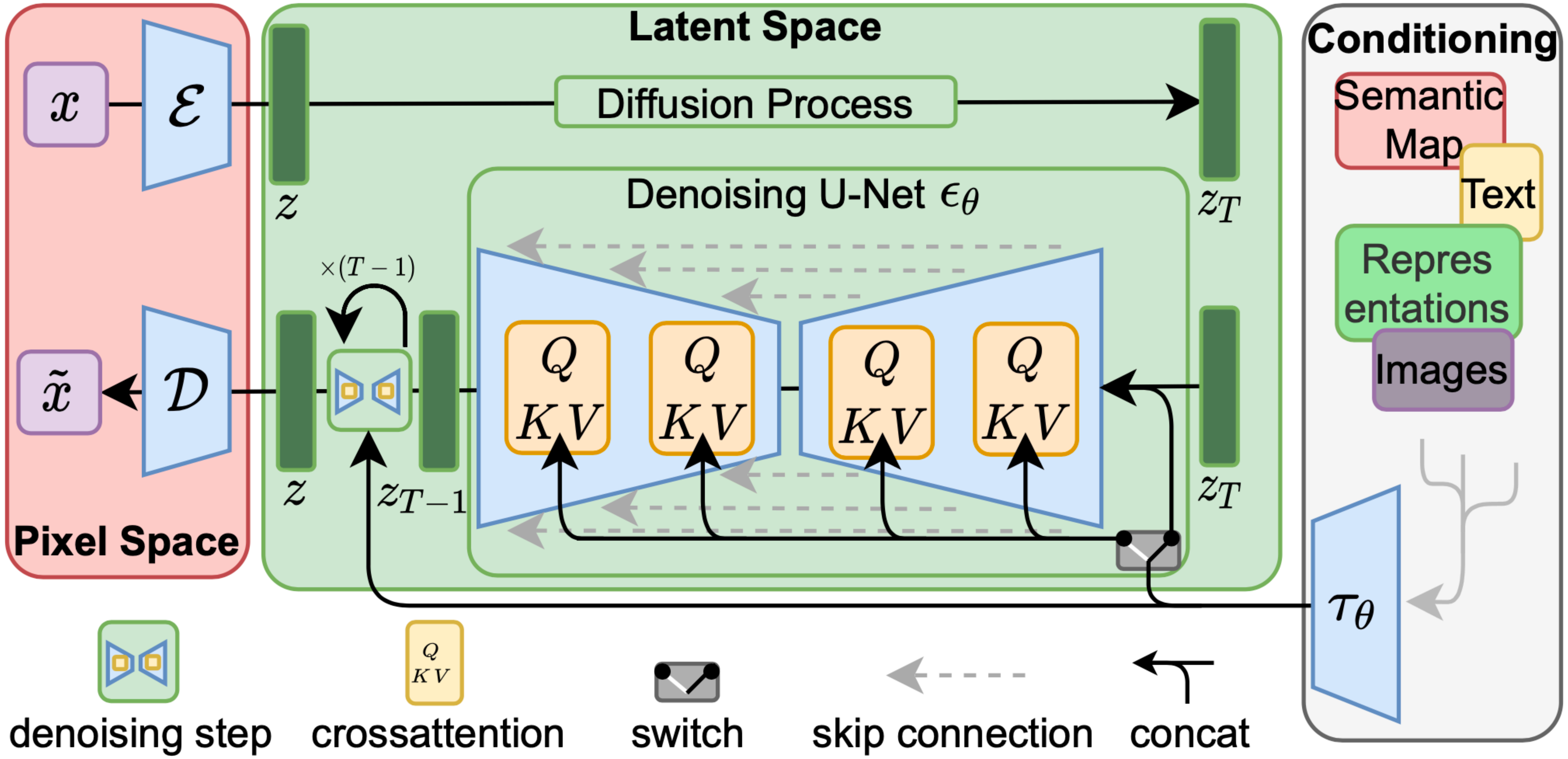
- The inference is done by starting from a Gaussian distribution
  - Then, keep denoising

Algorithm 2 Sampling

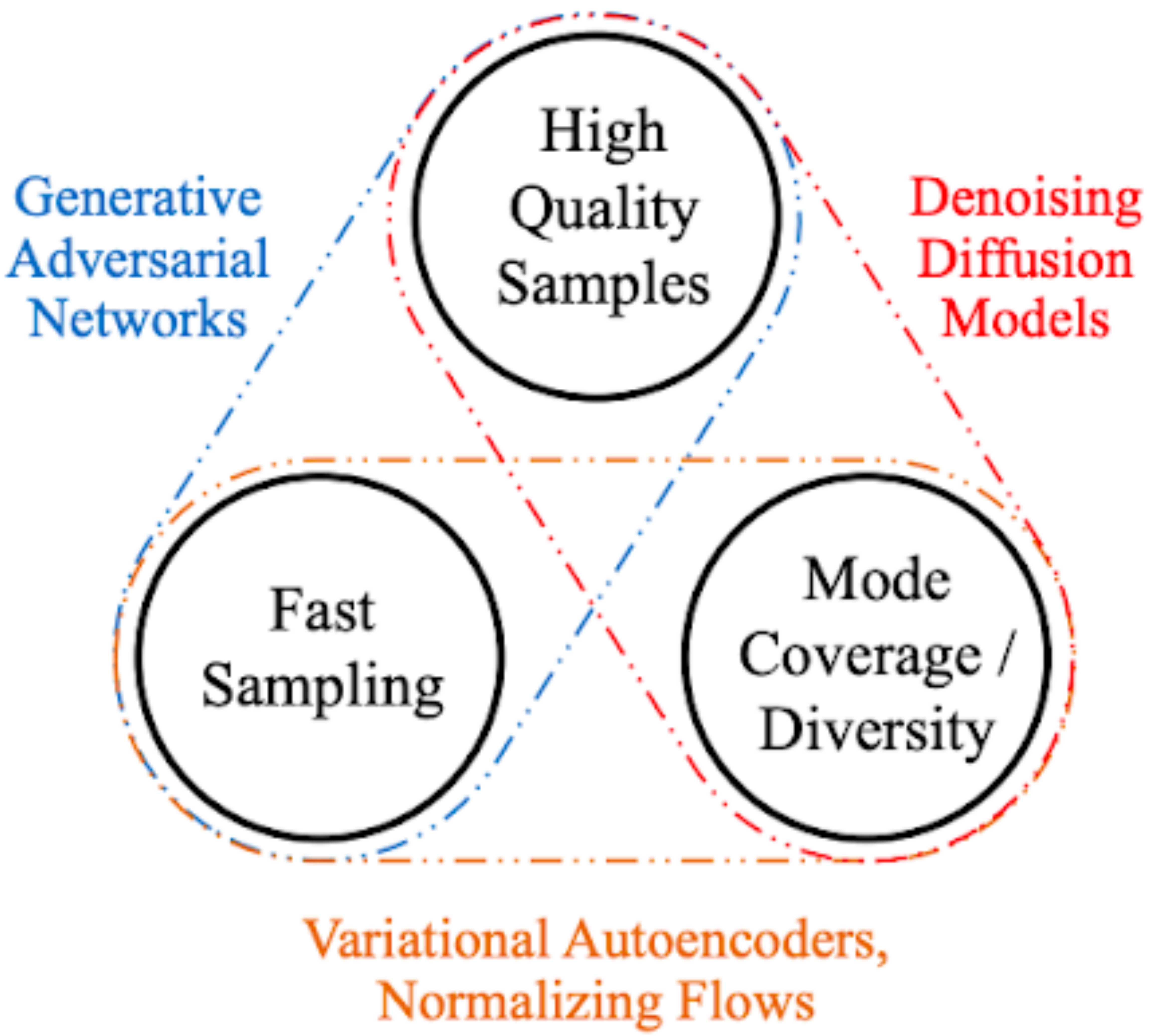
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\quad \mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}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

- **Beginner**
  - https://huggingface.co/blog/annotated-diffusion
  - https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

- **Advanced**
  - https://arxiv.org/abs/2403.18103
  - https://arxiv.org/abs/2510.21890 <- highly recommended

</lecture 18>