AI504: Programming for Artificial Intelligence

Week 14: Deep Diffusion Probabilistic Model

Edward Choi
Grad School of AI
edwardchoi@kaist.ac.kr

Today’s Topic

- Generative models recap
- Deep diffusion probabilistic model (DDPM)
- Deep diffusion implicit model (DDIM)
- Classifier-guided diffusion

Generative Models Recap

VAE

- Objective
  - Compress $\mathbf{x}$ to $\mathbf{z}$ which follows $P(\mathbf{Z} \mid \mathbf{X})$
  - Decompress $\mathbf{z}$ to reconstruct $\mathbf{x}$

Encoding (Compression)

$$q_\theta(z \mid x_i)$$

-1.2
3.1
0.2
-0.9

Decoding (Decompression)

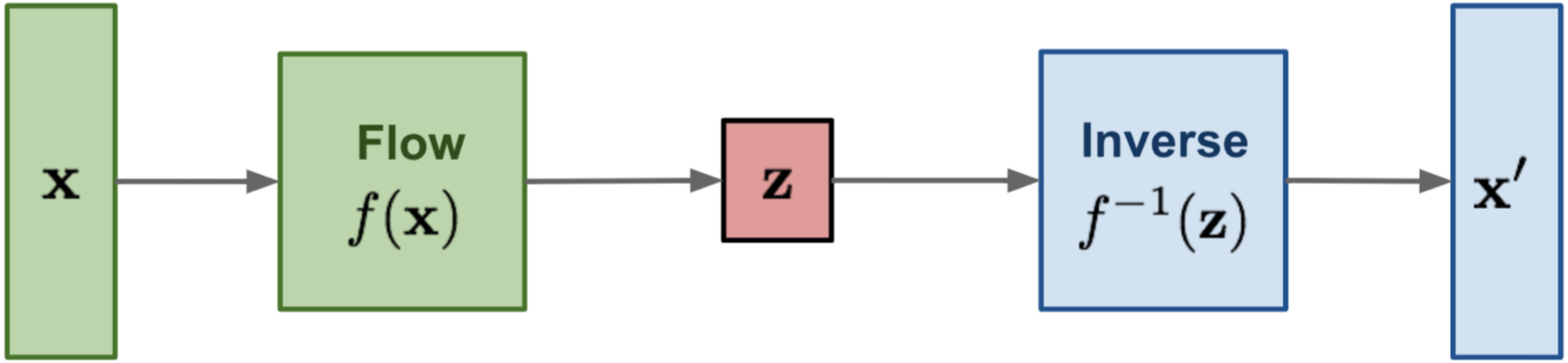
$$p_\phi(x_i \mid z)$$

This follows the distribution P (e.g. Gaussian N(0, 1))

$$\mathbb{KL}(q_\theta(z \mid x_i) \mid\mid p(z))$$

Flow-based Models

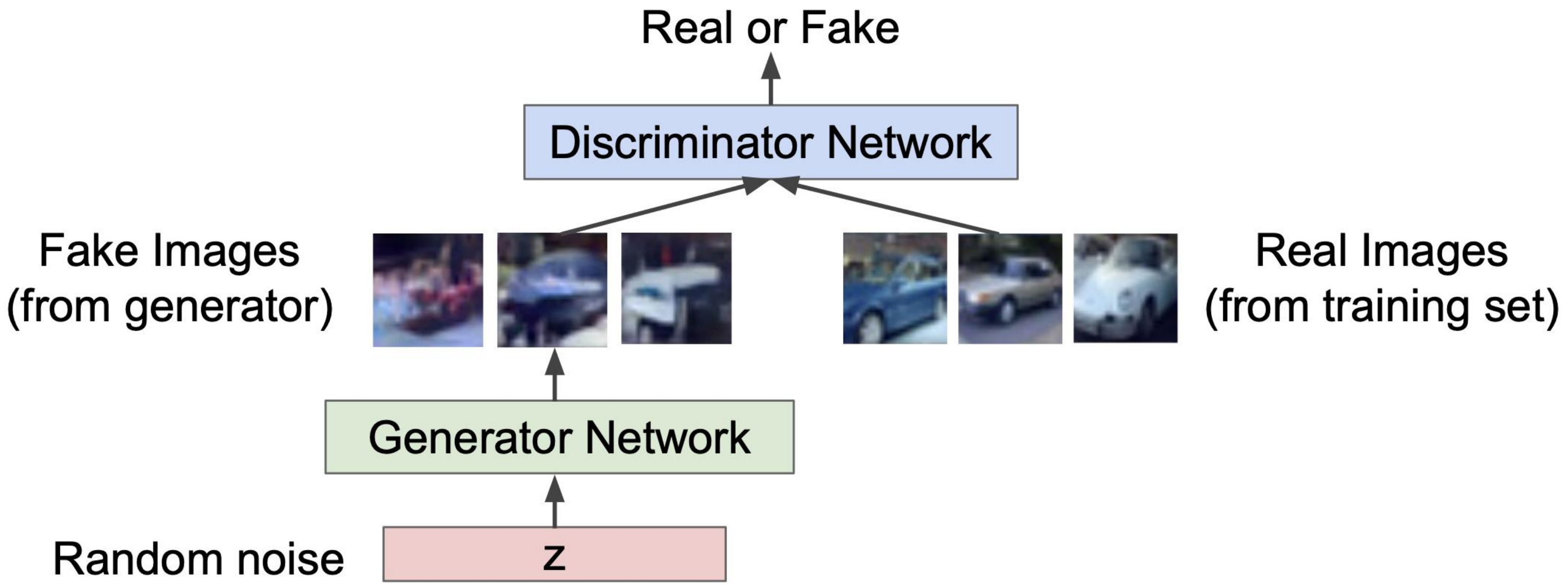
- Consistent mapping between **x** and **z**



  - Overcomes the limitation of VAE
  - Must use (a sequence of) invertible functions only
  - Can be trained via log-likelihood (no variational inference)

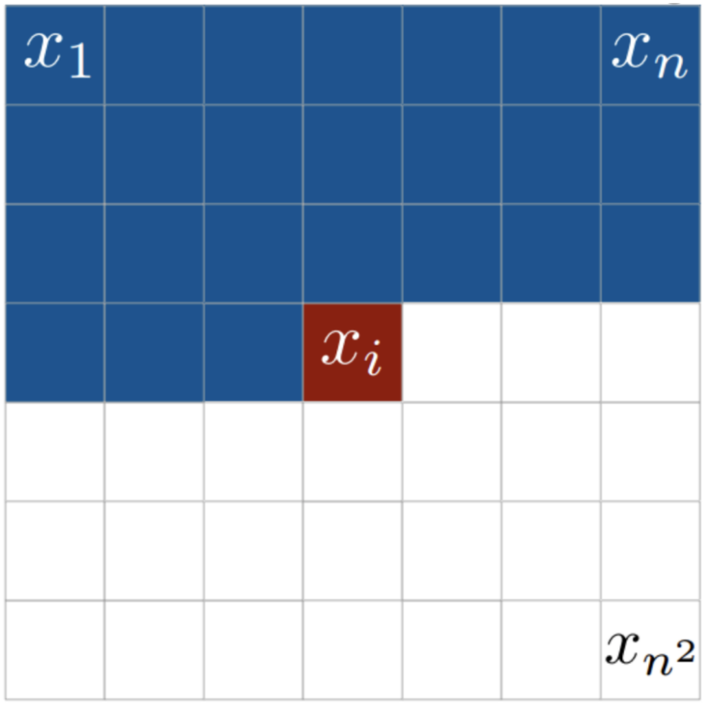
Generative Adversarial Network

- Generator (G)
  - Tries to fool D with fake samples x’
- Discriminator (D)
  - Tries to discriminate between real samples x and fake samples x’



Autoregressive Models

- Pixel-CNN
  - Generate images one pixel at a time.
- WaveNet
  - Generate audio one frame at a time
- GPT-3
  - Generate text one word at a time
- DALL-E 1
  - Generate one visual “code" at a time

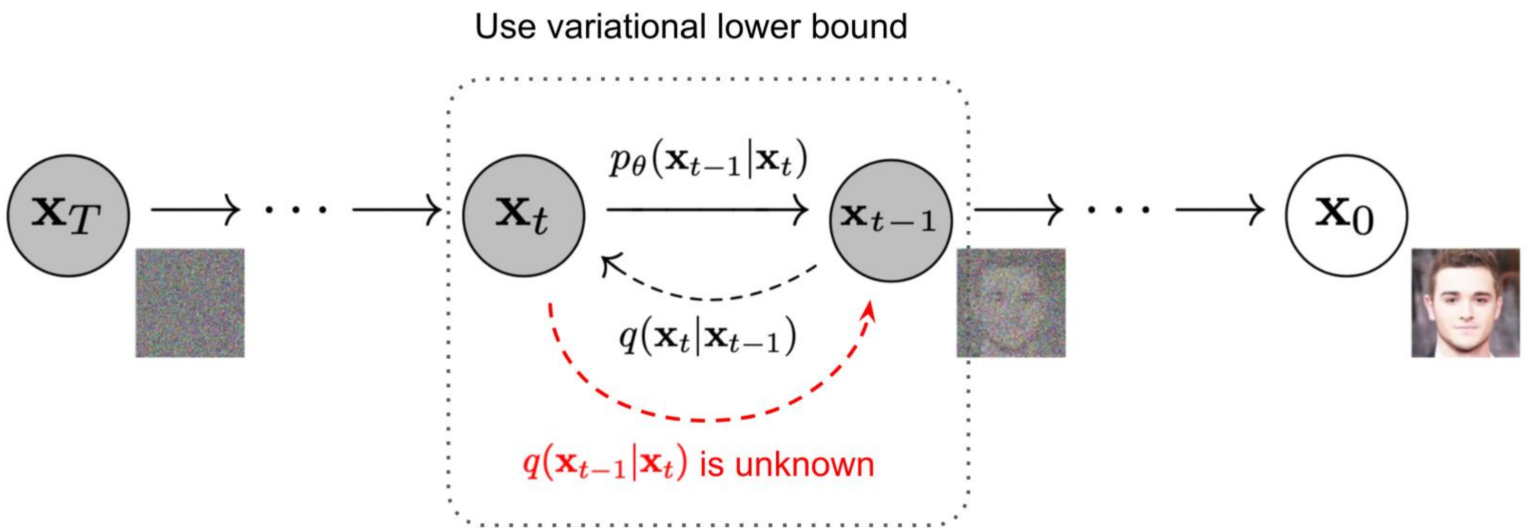


PixelCNN generates one pixel at a time

https://towardsdatascience.com/auto-regressive-generative-models-pixelrnn-pixelcnn-32d192911173

Diffusion

- Gradually add/remove noise



- Can be seen as a multi-step VAE
- Bayes rule, variational inference, reparameterization trick…

https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

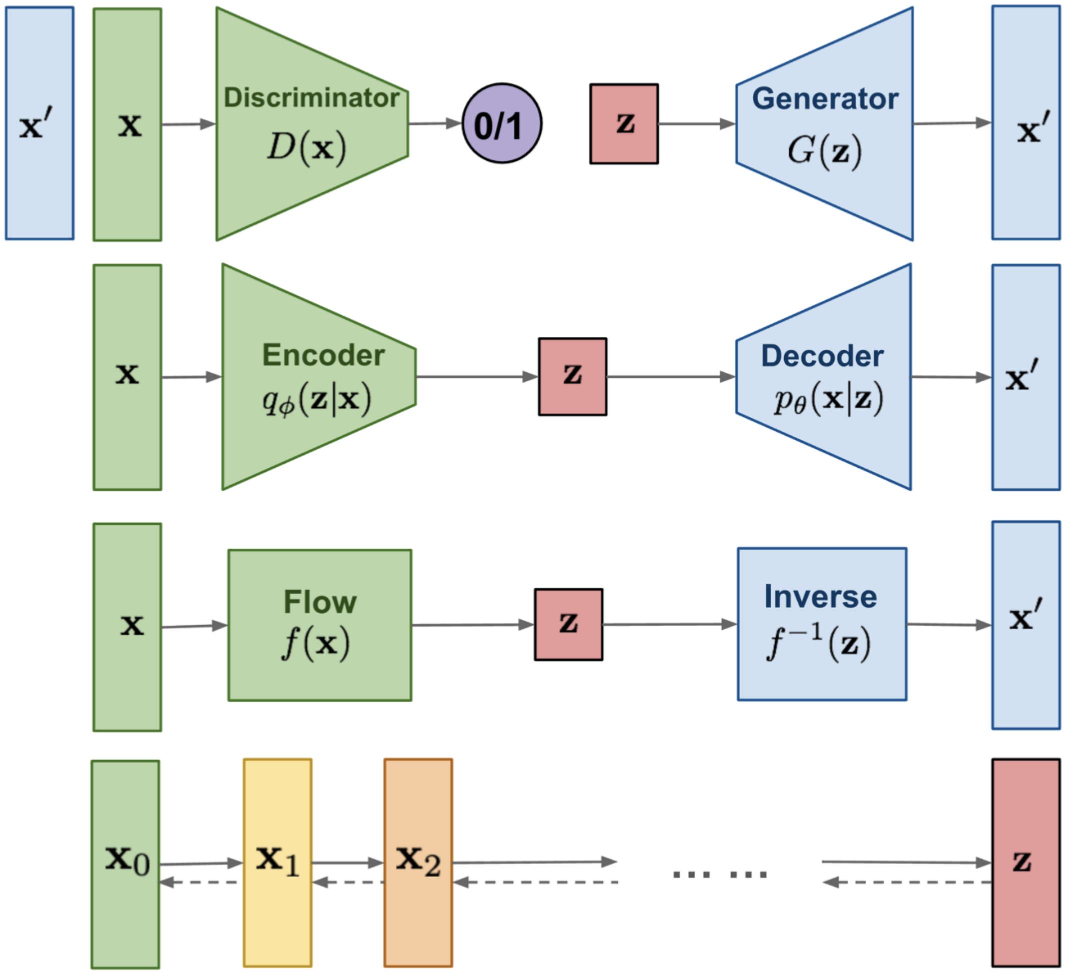
Comparison Overview

GAN: Adversarial training

VAE: maximize variational lower bound

Flow-based models: Invertible transform of distributions

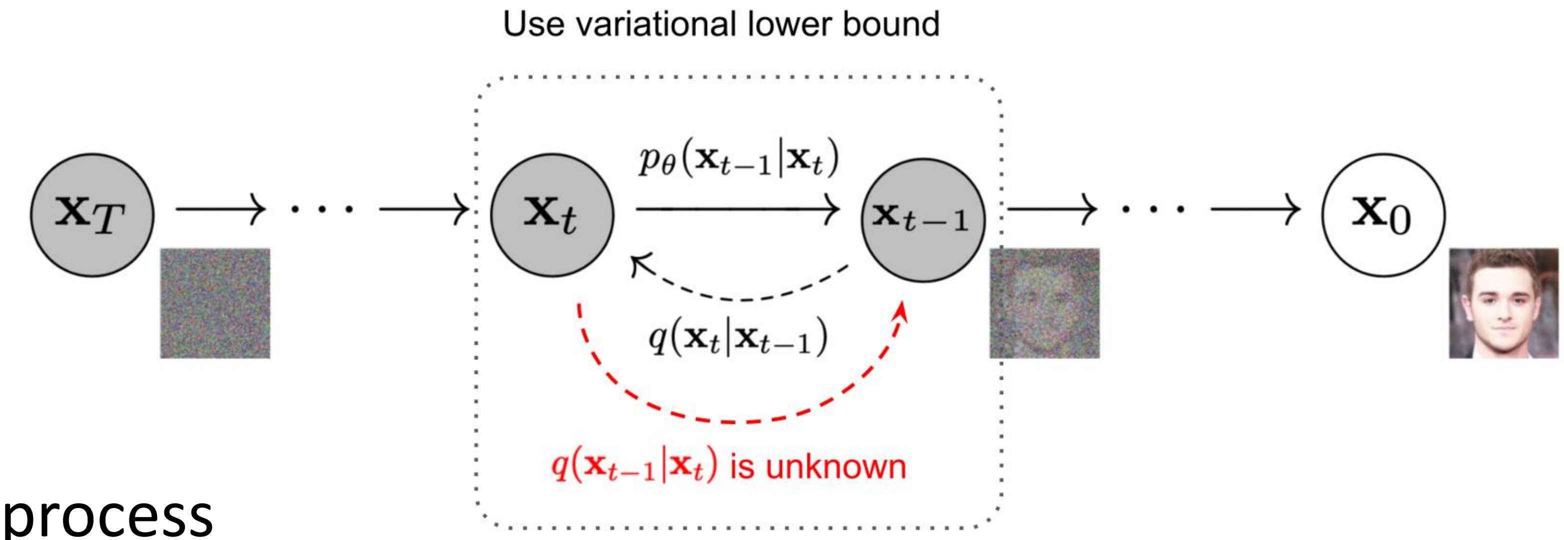
Diffusion models: Gradually add Gaussian noise and then reverse



https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

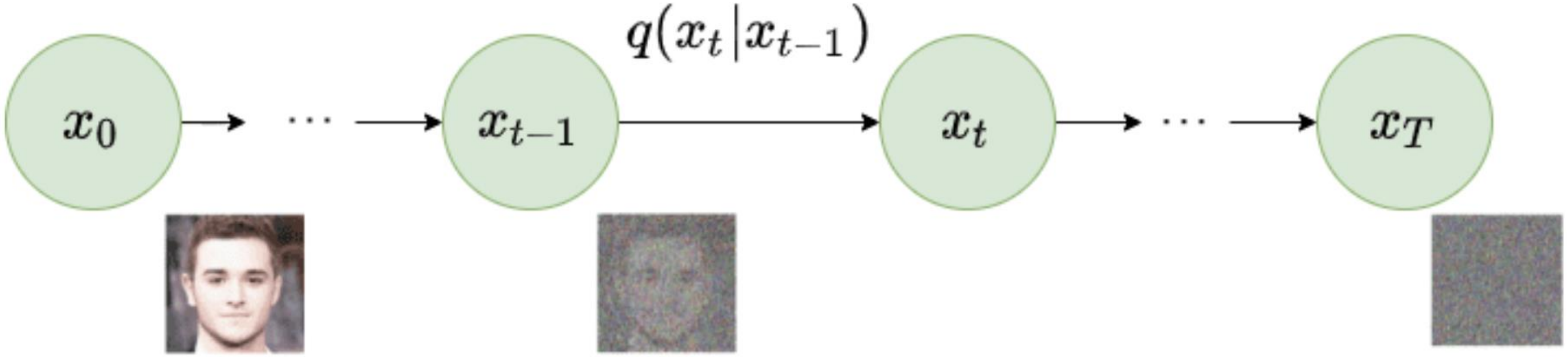
DDPM

Diffusion



- Forward process
  - Keep adding noise to the original data $\mathbf{x}_0 \sim p_{data} \rightarrow \mathbf{x}_T \sim \mathcal{N}(0, I)$ when $T \rightarrow \infty$
- Reverse process
  - Remove noise from $\mathbf{x}_T \rightarrow \mathbf{x}_0$
- Both processes are Markov chains
  - $\mathbf{x}_t$ is determined only by $\mathbf{x}_{t-1}$

Forward Process

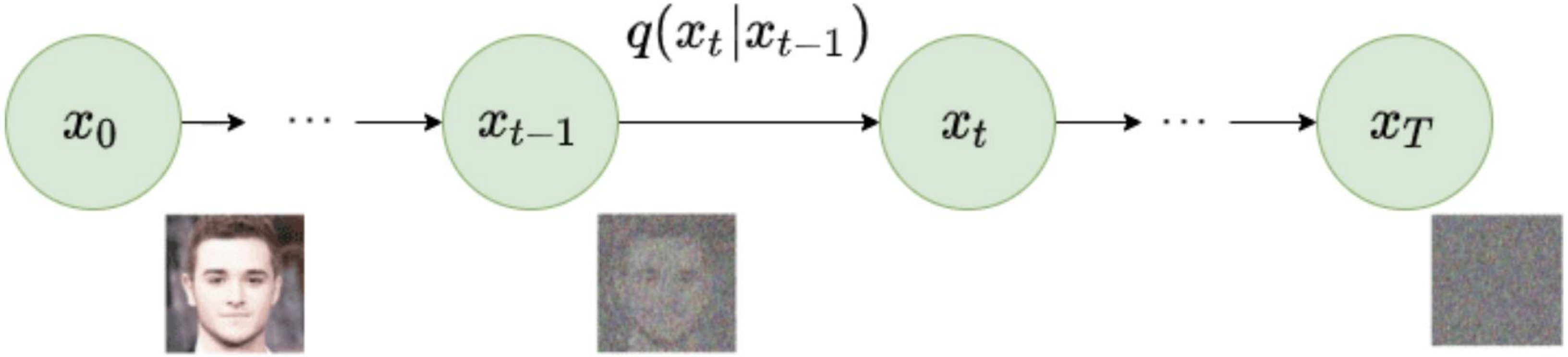


$q(\mathbf{x}_0)$: real data distribution

$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$ ➡ $\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\epsilon, \;\; \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

- Use conditional Gaussian distribution
- β_t: variance schedule
  - Can be learned, or can be fixed
  - DDPM uses scheduled constants ($0 < \beta_1 < \beta_2 < \ldots < \beta_T < 1$)

Forward Process

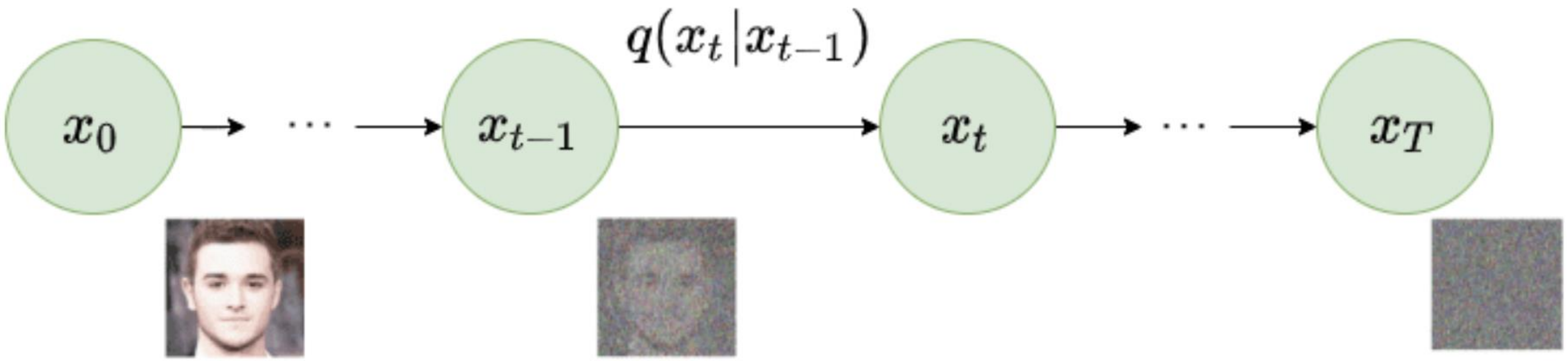


$q(\mathbf{x}_0)$: real data distribution

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \Rightarrow \mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\epsilon, \ \ \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Summary: We can convert any sample to $\mathcal{N}(0, I)$ without learnable parameters by repeatedly adding small Gaussian noise

Forward Process



$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$ ⬅ One-step probability distribution

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$$ ⬅ Joint probability distribution

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$$ ⬅ Creating $\mathbf{x}_t$ straight from $\mathbf{x}_0$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$

As $T \to \infty$: $\bar{\alpha}_t \to 0$ and $q(\mathbf{x}_t|\mathbf{x}_0) \to \mathcal{N}(0, I)$

Forward Process

- Creating $\mathbf{x}_t$ straight from $\mathbf{x}_0$
  - We don't have to repeatedly apply noise to get $\mathbf{x}_t$!
  - We will use this during training

Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$

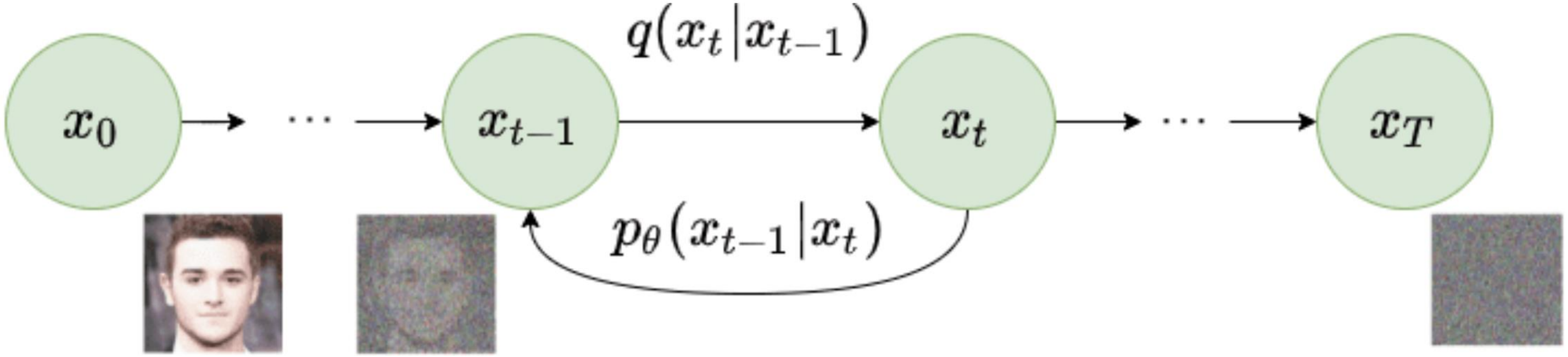
$$
\begin{aligned}
\mathbf{x}_t &= \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} && ;\text{where } \boldsymbol{\epsilon}_{t-1}, \boldsymbol{\epsilon}_{t-2}, \dots \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\
&= \sqrt{\alpha_t}\left(\sqrt{\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon}_{t-2}\right) + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \\
&= \sqrt{\alpha_t}\sqrt{\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{\alpha_t}\sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon}_{t-2} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{\alpha_t(1-\alpha_{t-1})}\boldsymbol{\epsilon}_{t-2} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\boldsymbol{\epsilon}}_{t-2} && ;\text{where } \bar{\boldsymbol{\epsilon}}_{t-2} \text{ merges two Gaussians (*).} \\
&= \dots \\
&= \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}
\end{aligned}
$$

$$\boxed{q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})}$$

(*) Note that $\mathcal{N}(\mathbf{0}, \sigma_1^2\mathbf{I})$ + $\mathcal{N}(\mathbf{0}, \sigma_2^2\mathbf{I})$ = $\mathcal{N}(\mathbf{0}, (\sigma_1^2 + \sigma_2^2)\mathbf{I})$

$$\therefore \sqrt{(1-\alpha_t) + \alpha_t(1-\alpha_{t-1})} = \sqrt{1-\alpha_t\alpha_{t-1}}$$

Reverse Process



- How do we "denoise" a noisy sample i.e., $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$?
  - We don't know the distribution of all images!
- We use a neural network to approximate

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

Note: $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ can be modeled as a Gaussian, because $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is Gaussian when β_t is small enough

Reverse Process

- If we can learn $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$:
  - We can sample from $\mathcal{N}(0, I)$, and "denoise" it to a real sample
- But how do we train $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$?

$$\begin{aligned}
L_{NLL} &= -\mathbb{E}_{q(\mathbf{x}_0)} \log p_\theta(\mathbf{x}_0) \\
&= -\mathbb{E}_{q(\mathbf{x}_0)} \log \Big(\int p_\theta(\mathbf{x}_{0:T}) d\mathbf{x}_{1:T} \Big) \\
&= -\mathbb{E}_{q(\mathbf{x}_0)} \log \Big(\int q(\mathbf{x}_{1:T}|\mathbf{x}_0) \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} d\mathbf{x}_{1:T} \Big) \\
&= -\mathbb{E}_{q(\mathbf{x}_0)} \log \Big(\mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \Big) \\
&\leq -\mathbb{E}_{q(\mathbf{x}_{0:T})} \log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \qquad (\because \text{Jensen's inequality}) \\
&= \mathbb{E}_{q(\mathbf{x}_{0:T})} \Big[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})} \Big] = L_{\text{VLB}}
\end{aligned}$$

Another way to derive variational lower bound

Reverse Process

- If we can learn $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$:
  - We can sample from $\mathcal{N}(0, I)$, and "denoise" it to a real sample
- But how do we train $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$?

$$
\begin{aligned}
L_{\text{NLL}} &= -\mathbb{E}_{q(\mathbf{x}_0)} \log p_\theta(\mathbf{x}_0) \\
&= -\mathbb{E}_{q(\mathbf{x}_0)} \log \left(\int p_\theta(\mathbf{x}_{0:T}) d\mathbf{x}_{1:T} \right) \\
&= -\mathbb{E}_{q(\mathbf{x}_0)} \log \left(\int q(\mathbf{x}_{1:T}|\mathbf{x}_0) \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} d\mathbf{x}_{1:T} \right) \\
&= -\mathbb{E}_{q(\mathbf{x}_0)} \log \left(\mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \right) \\
&\leq -\mathbb{E}_{q(\mathbf{x}_{0:T})} \log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \\
&= \mathbb{E}_{q(\mathbf{x}_{0:T})} \left[\boxed{\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}} \right] = L_{\text{VLB}}
\end{aligned}
$$

$$
\log \frac{\prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}
$$

Further break down of L_{VLB} to reduce variance

$$
\begin{aligned}
L_{\text{VLB}} &= \mathbb{E}_{q(\mathbf{x}_{0:T})}\Big[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\Big] \\
&= \mathbb{E}_q\Big[\log \frac{\prod_{t=1}^T q(\mathbf{x}_t|\mathbf{x}_{t-1})}{\boxed{p_\theta(\mathbf{x}_T)}\prod_{t=1}^T p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=1}^T \log \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \boxed{\log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \Big(\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} \cdot \frac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}\Big) + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \sum_{t=2}^T \log \frac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{q(\mathbf{x}_1|\mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[\log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{p_\theta(\mathbf{x}_T)} + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)\Big]
\end{aligned}
$$

Should be $p(\mathbf{x}_T)$

Separate treatment to avoide edge effect

$$
= \mathbb{E}_{q(\mathbf{x}_0)}\underbrace{\left[D_{\mathrm{KL}}\left(q\left(\mathbf{x}_T \mid \mathbf{x}_0\right) \| p_\theta\left(\mathbf{x}_T\right)\right)\right]}_{L_T} + \sum_{t=2}^T \mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_t)}\underbrace{\left[D_{\mathrm{KL}}\left(q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t,\mathbf{x}_0\right) \| p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)\right)\right]}_{L_{t-1}} + \mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_1)}\underbrace{\left[-\log p_\theta\left(\mathbf{x}_0 \mid \mathbf{x}_1\right)\right]}_{L_0}
$$

Further break down of L_{VLB} to reduce variance

$$
\begin{aligned}
L_{\mathrm{VLB}} &= \mathbb{E}_{q(\mathbf{x}_{0:T})}\Big[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\Big] \\
&= \mathbb{E}_q\Big[\log \frac{\prod_{t=1}^T q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_T)\prod_{t=1}^T p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=1}^T \log \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{\boxed{q(\mathbf{x}_t|\mathbf{x}_{t-1})}}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \boxed{\Big(\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} \cdot \frac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}\Big)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \sum_{t=2}^T \log \frac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{q(\mathbf{x}_1|\mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[\log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{p_\theta(\mathbf{x}_T)} + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)\Big]
\end{aligned}
$$

$$
= \mathbb{E}_{q(\mathbf{x}_0)}\big[\underbrace{D_{\mathrm{KL}}\left(q\left(\mathbf{x}_T \mid \mathbf{x}_0\right) \| p_\theta\left(\mathbf{x}_T\right)\right)}_{L_T}\big] + \sum_{t=2}^T \mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_t)}\big[\underbrace{D_{\mathrm{KL}}\left(q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t,\mathbf{x}_0\right) \| p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)\right)}_{L_{t-1}}\big] + \mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_1)}\big[\underbrace{-\log p_\theta\left(\mathbf{x}_0 \mid \mathbf{x}_1\right)}_{L_0}\big]
$$

This term can be written as a Gaussian distribution.
We will revisit this later

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

Further break down of L_{VLB} to reduce variance

$$
\begin{aligned}
L_{\text{VLB}} &= \mathbb{E}_{q(\mathbf{x}_{0:T})}\Big[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\Big] \\
&= \mathbb{E}_q\Big[\log \frac{\prod_{t=1}^T q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_T)\prod_{t=1}^T p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=1}^T \log \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \Big(\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\cdot\frac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}\Big) + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \boxed{\sum_{t=2}^T \log \frac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \boxed{\log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{q(\mathbf{x}_1|\mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}}\Big] \\
&= \mathbb{E}_q\Big[\log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{p_\theta(\mathbf{x}_T)} + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)\Big]
\end{aligned}
$$

Terms cancel each other out

$$
= \mathbb{E}_{q(\mathbf{x}_0)}[\underbrace{D_{\text{KL}}\left(q\left(\mathbf{x}_T \mid \mathbf{x}_0\right) \| p_\theta\left(\mathbf{x}_T\right)\right)}_{L_T}] + \sum_{t=2}^T \mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_t)}[\underbrace{D_{\text{KL}}\left(q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0\right) \| p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)\right)}_{L_{t-1}}] + \mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_1)}[\underbrace{-\log p_\theta\left(\mathbf{x}_0 \mid \mathbf{x}_1\right)}_{L_0}]
$$

Looking at Each Term

$$L_{\text{VLB}} = L_T + L_{T-1} + \cdots + L_0$$

$$\text{where } L_T = D_{\text{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_T))$$

$$L_{t-1} = D_{\text{KL}}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t\,)\big) \text{ for } 2 \leq t \leq T$$

$$L_0 = -\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)$$

$$L_T = D_{\text{KL}}(\boxed{q(\mathbf{x}_T|\mathbf{x}_0)} \parallel \boxed{p_\theta(\mathbf{x}_T)})$$

Fixed process,
nothing to learn

This is actually just $\mathcal{N}(0, I)$

Therefore we can ignore L_T

Looking at Each Term

$$L_{\mathrm{VLB}} = L_T + L_{T-1} + \cdots + L_0$$
$$\text{where } L_T = D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_T))$$
$$L_{t-1} = D_{\mathrm{KL}}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) || p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t\ )\big) \text{ for } 2 \le t \le T$$
$$L_0 = -\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)$$

$$L_0 = -\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)$$

This is the edge case.
Ho et al. models this with a separate decoder derived from $\mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_\theta(\mathbf{x}_1, 1), \boldsymbol{\Sigma}_\theta(\mathbf{x}_1, 1))$

Looking at Each Term

$$L_{\mathrm{VLB}} = L_T + L_{T-1} + \cdots + L_0$$

$$\text{where } L_T = D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_T))$$

$$L_{t-1} = D_{\mathrm{KL}}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)\big) \text{ for } 2 \leq t \leq T$$

$$L_0 = -\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)$$

$$L_{t-1} = D_{\mathrm{KL}}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)\big) \text{ for } 2 \leq t \leq T$$

This is a KL divergence between two Gaussian distributions

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}) \Rightarrow$$

Why this form of Gaussian distribution?

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

Modeling $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ as Guassian Distrubiton

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

$$\propto \exp\Big(-\frac{1}{2}\big(\frac{(\mathbf{x}_t - \sqrt{\alpha_t}\mathbf{x}_{t-1})^2}{\beta_t} + \frac{(\mathbf{x}_{t-1} - \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0)^2}{1-\bar{\alpha}_{t-1}} - \frac{(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0)^2}{1-\bar{\alpha}_t}\big)\Big)$$

$$= \exp\Big(-\frac{1}{2}\big(\frac{\mathbf{x}_t^2 - 2\sqrt{\alpha_t}\mathbf{x}_t\mathbf{x}_{t-1} + \alpha_t\mathbf{x}_{t-1}^2}{\beta_t} + \frac{\mathbf{x}_{t-1}^2 - 2\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0\mathbf{x}_{t-1} + \bar{\alpha}_{t-1}\mathbf{x}_0^2}{1-\bar{\alpha}_{t-1}} - \frac{(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0)^2}{1-\bar{\alpha}_t}\big)\Big)$$

$$= \exp\Big(-\frac{1}{2}\big((\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}})\mathbf{x}_{t-1}^2 - (\frac{2\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0)\mathbf{x}_{t-1} + \boxed{C(\mathbf{x}_t, \mathbf{x}_0)}\big)\Big)$$

Some function not involving $\mathbf{x}_{t-1}$

Considering that Gaussian pdf: $f(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \det(2\pi\boldsymbol{\Sigma})^{-\frac{1}{2}}\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t\mathbf{I})$$

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) = (\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0)/(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}})$$

$$= (\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0)\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t$$

$$= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0$$

From p.15

$$= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t)$$

$$= \frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\Big)$$

$$\tilde{\beta}_t = 1/(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}) = 1/(\frac{\alpha_t - \bar{\alpha}_t + \beta_t}{\beta_t(1-\bar{\alpha}_{t-1})}) = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t$$

Looking at Each Term

$$L_{t-1} = D_{\mathrm{KL}}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t\ )\big) \text{ for } 2 \le t \le T$$

$$q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0\right) = \mathcal{N}\left(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t, \mathbf{x}_0\right), \tilde{\beta}_t \mathbf{I}\right) = \mathcal{N}\left(\mathbf{x}_{t-1}; \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right), \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \mathbf{I}\right)$$

$$p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta\left(\mathbf{x}_t, t\right), \boldsymbol{\Sigma}_\theta\left(\mathbf{x}_t, t\right)\right)$$

Let us set $\boldsymbol{\mu}_\theta\left(\mathbf{x}_t, t\right) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)$ and $\boldsymbol{\Sigma}_\theta\left(\mathbf{x}_t, t\right) = \sigma_t^2 \mathbf{I}$.

We have two options for σ_t^2: $\sigma_t^2 = \beta_t$ and $\sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$.

According to Ho et al. (2020), both had similar results experimentally.

$$^{*}D_{KL}(p\|q) = \frac{1}{2}\left[\log\frac{|\Sigma_q|}{|\Sigma_p|} - k + \left(\boldsymbol{\mu}_p - \boldsymbol{\mu}_q\right)^T \Sigma_q^{-1}\left(\boldsymbol{\mu}_p - \boldsymbol{\mu}_q\right) + \operatorname{tr}\left\{\Sigma_q^{-1}\Sigma_p\right\}\right]$$

$$\begin{aligned}
L_{t-1} &\propto \frac{1}{2\sigma_t^2}\left\|\tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t, \mathbf{x}_0\right) - \boldsymbol{\mu}_\theta\left(\mathbf{x}_t, t\right)\right\|^2 \\
&= \frac{1}{2\sigma_t^2}\left\|\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right) - \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)\right\|^2 \\
&= \frac{\beta_t^2}{2\sigma_t^2\alpha_t\left(1-\bar{\alpha}_t\right)}\left\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t, t\right)\right\|^2 \\
&= \frac{\beta_t^2}{2\sigma_t^2\alpha_t\left(1-\bar{\alpha}_t\right)}\left\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, t\right)\right\|^2
\end{aligned}$$

Simpler Loss Function

- Ho et al. 2020 empirically found that a simplied loss function works better

$$L_t = \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t)\|\boldsymbol{\Sigma}_\theta\|_2^2}\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, t)\|^2\right]$$

Need to sum L_t over $1 \le t \le T-1$
But now instead uniform sampling

VS

$$\begin{aligned} L_t^{\text{simple}} &= \mathbb{E}_{\boxed{t\sim[1,T]},\mathbf{x}_0,\boldsymbol{\epsilon}_t}\left[\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\|^2\right] \\ &= \mathbb{E}_{t\sim[1,T],\mathbf{x}_0,\boldsymbol{\epsilon}_t}\left[\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, t)\|^2\right] \end{aligned}$$

Training & Sampling Algorithm

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: 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

Algorithm 2 Sampling

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

* Need to train seprate decoder for L_0

Let us set $\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right)$ and $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \sigma_t^2 \mathbf{I}$.

Generation Samples

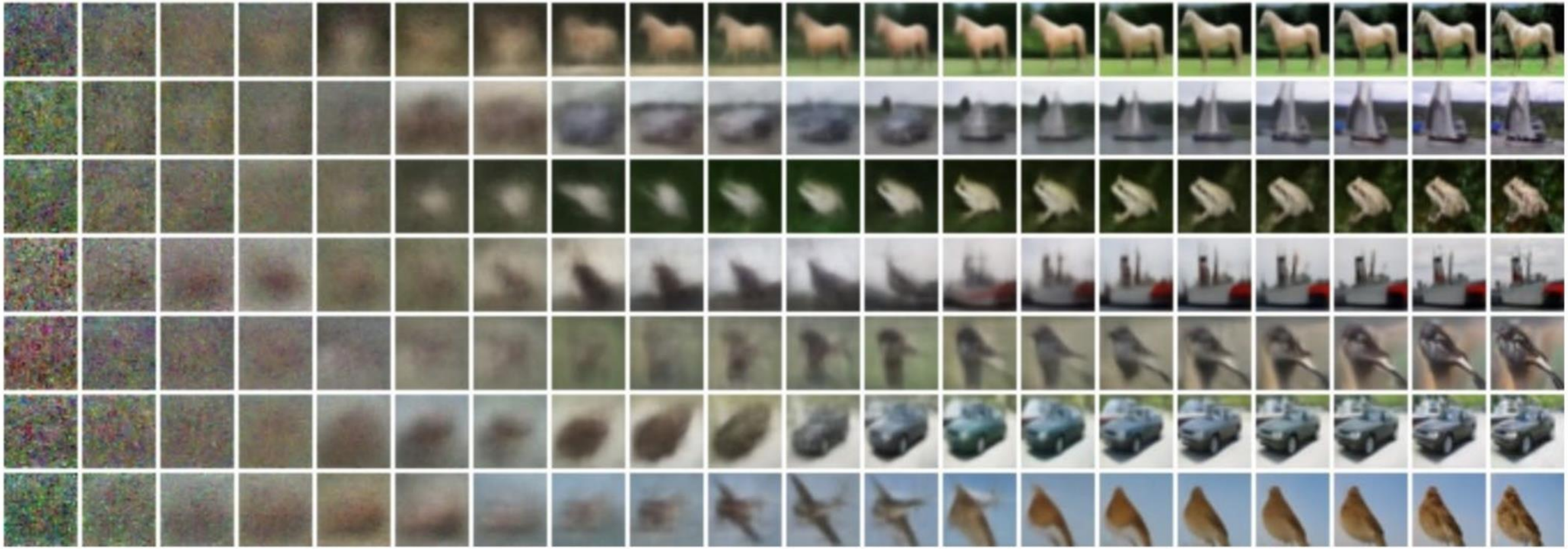


Figure 14: Unconditional CIFAR10 progressive generation

Generation Samples

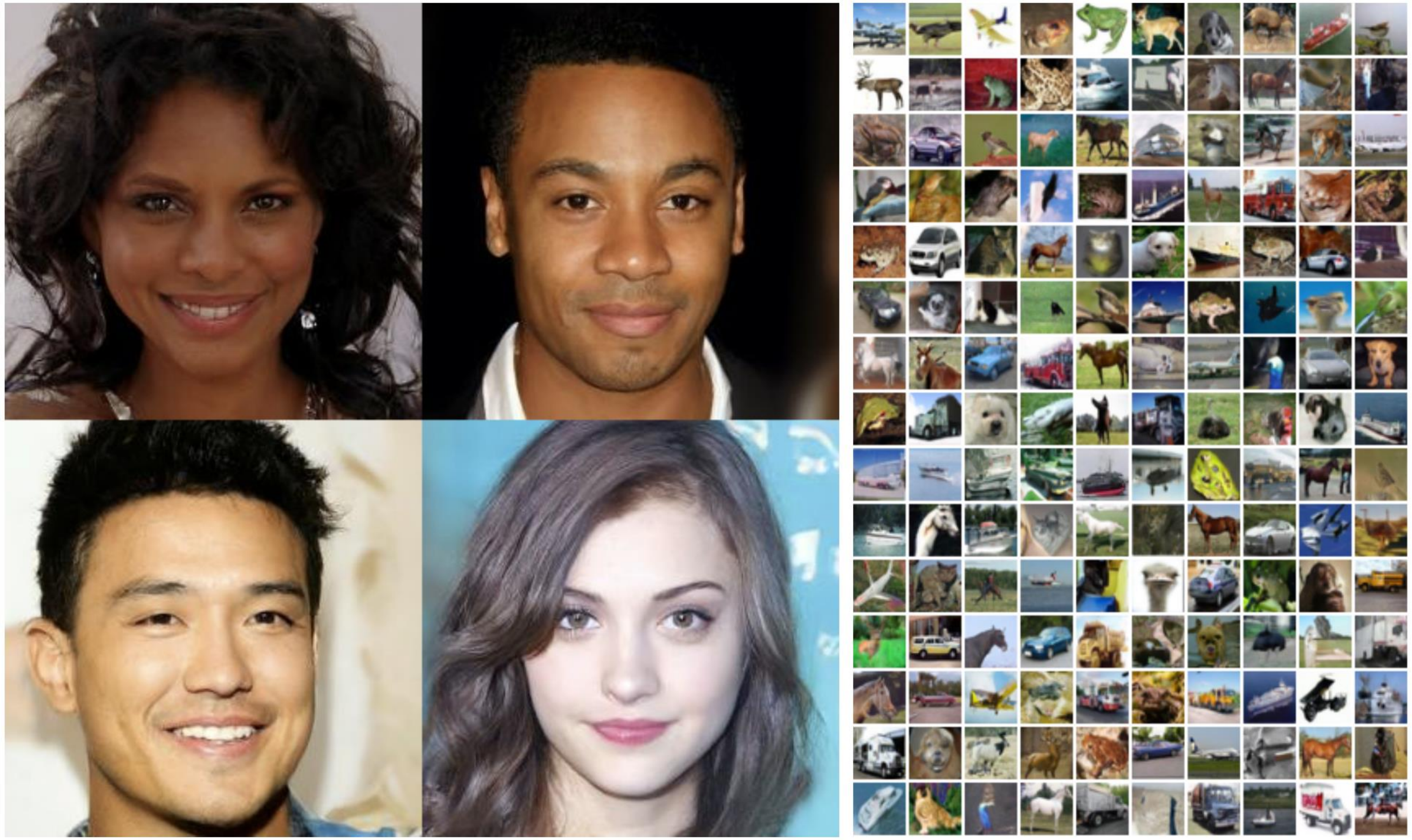


Figure 1: Generated samples on CelebA-HQ 256 × 256 (left) and unconditional CIFAR10 (right)

Quiz 1

- Which of the following is false?
  1. DDPM does not explicitly learn data distribution, unlike autoregressive models (e.g. PixelCNN)
  2. We can model the reverse process with Gaussian due to the Markovian assumption
  3. The input & output sizes of the epsilon function are the same
  4. The fact that we can create x_t from x_0 allows efficient training
  5. In essence, DDPM learns to estimate the noise added between x_{t-1} and x_t

Improved DDPM

- Nichol et al. 2021
- Improved formulation & training than vanilla DDPM

Let us set $\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)$ and $\boxed{\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \sigma_t^2\mathbf{I}.}$
We have two options for σ_t^2: $\sigma_t^2 = \beta_t$ and $\sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$.

Let's learn a diagonal covariance matrix instead!

- Covariance matrix as an interpolation between β_t and $\tilde{\beta}_t$

$$\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \exp(\mathbf{v}\log\beta_t + (1-\mathbf{v})\log\tilde{\beta}_t) \quad \text{where} \quad \tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t$$

$$L_{\text{hybrid}} = L_{\text{simple}} + \lambda L_{\text{VLB}} \text{ where } \lambda = 0.001$$

Model	ImageNet	CIFAR
Glow (Kingma & Dhariwal, 2018)	3.81	3.35
Flow++ (Ho et al., 2019)	3.69	3.08
PixelCNN (van den Oord et al., 2016c)	3.57	3.14
SPN (Menick & Kalchbrenner, 2018)	3.52	-
NVAE (Vahdat & Kautz, 2020)	-	2.91
Very Deep VAE (Child, 2020)	3.52	2.87
PixelSNAIL (Chen et al., 2018)	3.52	2.85
Image Transformer (Parmar et al., 2018)	3.48	2.90
Sparse Transformer (Child et al., 2019)	3.44	**2.80**
Routing Transformer (Roy et al., 2020)	**3.43**	-
DDPM (Ho et al., 2020)	3.77	3.70
DDPM (cont flow) (Song et al., 2020b)	-	2.99
Improved DDPM (ours)	**3.53**	**2.94**

Generalized DDPM (DDIM)

Generalized DDPM

- DDPM suffers from slow generation
  - Must go through thousands of steps to get high-quality samples

    "For example, it takes around 20 hours to sample 50k images of size 32 × 32 from a DDPM, but less than a minute to do so from a GAN on an Nvidia 2080 Ti GPU.", (Song et al. 2020)

- What if we generalized DDPM to non-Markovian?
  - Turns out that we can use pretrained DDPM with two benefits
    1. We can perform deterministic sampling (DDIM)
    2. We can take multiple steps during sampling (accelerated sampling)

Core Idea

- Non-Markovian forward process

$$q_\sigma(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) := q_\sigma(\boldsymbol{x}_T|\boldsymbol{x}_0)\prod_{t=2}^{T} q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$$

Note that from here on, $\alpha_t = \bar{\alpha}_t$

where $q_\sigma(\boldsymbol{x}_T|\boldsymbol{x}_0) = \mathcal{N}(\sqrt{\alpha_T}\boldsymbol{x}_0, (1-\alpha_T)\boldsymbol{I})$ and for all $t > 1$

$$q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}\left(\sqrt{\alpha_{t-1}}\boldsymbol{x}_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\cdot\frac{\boldsymbol{x}_t - \sqrt{\alpha_t}\boldsymbol{x}_0}{\sqrt{1-\alpha_t}}, \sigma_t^2\boldsymbol{I}\right)$$

This guarantees $q_\sigma(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\sqrt{\alpha_t}\boldsymbol{x}_0, (1-\alpha_t)\boldsymbol{I})$ (Same as DDPM in p.15)

- Reverse process

$$p_\theta^{(t)}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \begin{cases}\mathcal{N}(f_\theta^{(1)}(\boldsymbol{x}_1), \sigma_1^2\boldsymbol{I}) & \text{if } t = 1\\ q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, f_\theta^{(t)}(\boldsymbol{x}_t)) & \text{otherwise,}\end{cases}$$

where $f_\theta^{(t)}(\boldsymbol{x}_t) := (\boldsymbol{x}_t - \sqrt{1-\alpha_t}\cdot\epsilon_\theta^{(t)}(\boldsymbol{x}_t))/\sqrt{\alpha_t}$

Properties of Generalized DDPM

- Training objective

$$J_\sigma(\epsilon_\theta) := \mathbb{E}_{\boldsymbol{x}_{0:T} \sim q_\sigma(\boldsymbol{x}_{0:T})}[\log q_\sigma(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) - \log p_\theta(\boldsymbol{x}_{0:T})]$$

$$= \mathbb{E}_{\boldsymbol{x}_{0:T} \sim q_\sigma(\boldsymbol{x}_{0:T})}\left[q_\sigma(\boldsymbol{x}_T|\boldsymbol{x}_0) + \sum_{t=2}^{T} \log q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) - \sum_{t=1}^{T} \log p_\theta^{(t)}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) - \log p_\theta(\boldsymbol{x}_T)\right]$$

Happens to optimize the same loss as DDPM

Theorem 1. *For all $\sigma > \mathbf{0}$, there exists $\gamma \in \mathbb{R}^T_{>0}$ and $C \in \mathbb{R}$, such that $J_\sigma = L_\gamma + C$.*

$$L_\gamma(\epsilon_\theta) := \sum_{t=1}^{T} \gamma_t \mathbb{E}_{\boldsymbol{x}_0 \sim q(\boldsymbol{x}_0), \epsilon_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})}\left[\left\|\epsilon_\theta^{(t)}(\sqrt{\alpha_t}\boldsymbol{x}_0 + \sqrt{1-\alpha_t}\epsilon_t) - \epsilon_t\right\|_2^2\right]$$

Therefore, no need to train new model.
➔ Can use a pre-trained DDPM

Sampling Process

- Sampling formula

$$x_{t-1} = \sqrt{\alpha_{t-1}} \underbrace{\left(\frac{x_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(x_t)}{\sqrt{\alpha_t}} \right)}_{\text{“ predicted } x_0\text{”}} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \epsilon_\theta^{(t)}(x_t)}_{\text{“direction pointing to } x_t\text{”}} + \underbrace{\sigma_t \epsilon_t}_{\text{random noise}}$$

- If we fix $\sigma_t = \sqrt{\frac{(1-\alpha_{t-1})}{(1-\alpha_t)}}\sqrt{\frac{(1-\alpha_t)}{\alpha_{t-1}}}$ for all t, the forward process becomes Markovian. In other words, the generative process becomes DDPM.
- If we fix $\sigma_t = 0$ for all t, the forward process becomes deterministic given x_{t-1} and x_0. Since the resulting model becomes an implicit probabilistic model, this model is called **“denoising diffusion implicit model (DDIM)”.**

Accelerated Sampling

$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}} \underbrace{\left(\frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(\boldsymbol{x}_t)}{\sqrt{\alpha_t}} \right)}_{\text{" predicted } \boldsymbol{x}_0 \text{"}} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \epsilon_\theta^{(t)}(\boldsymbol{x}_t)}_{\text{"direction pointing to } \boldsymbol{x}_t \text{"}} + \underbrace{\sigma_t \epsilon_t}_{\text{random noise}}$$

Sampling process of **Generalized DDPM**

$$\boldsymbol{x}_{\tau_{i-1}}(\eta) = \sqrt{\alpha_{\tau_{i-1}}} \left(\frac{\boldsymbol{x}_{\tau_i} - \sqrt{1-\alpha_{\tau_i}}\epsilon_\theta^{(\tau_i)}(\boldsymbol{x}_{\tau_i})}{\sqrt{\alpha_{\tau_i}}} \right) + \sqrt{1-\alpha_{\tau_{i-1}} - \sigma_{\tau_i}(\eta)^2} \cdot \epsilon_\theta^{(\tau_i)}(\boldsymbol{x}_{\tau_i}) + \sigma_{\tau_i}(\eta)\epsilon$$

$$\text{where} \quad \sigma_{\tau_i}(\eta) = \eta \sqrt{\frac{1-\alpha_{\tau_{i-1}}}{1-\alpha_{\tau_i}}} \sqrt{1 - \frac{\alpha_{\tau_i}}{\alpha_{\tau_{i-1}}}}$$

Accelerated Sampling process of **Generalized DDPM**

- Non-Markovian property allows "skipped" sampling
  - τ_i is the index number at which we perform a reverse process

Comparison

- DDPM vs DDIM

Table 1: CIFAR10 and CelebA image generation measured in FID. $\eta = 1.0$ and $\hat{\sigma}$ are cases of DDPM (although Ho et al. (2020) only considered $T = 1000$ steps, and $S < T$ can be seen as simulating DDPMs trained with S steps), and $\eta = 0.0$ indicates DDIM.

		CIFAR10 (32×32)					CelebA (64×64)				
	S	10	20	50	100	1000	10	20	50	100	1000
η	0.0	**13.36**	**6.84**	**4.67**	**4.16**	4.04	**17.33**	**13.73**	**9.17**	**6.53**	3.51
	0.2	14.04	7.11	4.77	4.25	4.09	17.66	14.11	9.51	6.79	3.64
	0.5	16.66	8.35	5.25	4.46	4.29	19.86	16.06	11.01	8.09	4.28
	1.0	41.07	18.36	8.01	5.78	4.73	33.12	26.03	18.48	13.93	5.98
	$\hat{\sigma}$	367.43	133.37	32.72	9.99	**3.17**	299.71	183.83	71.71	45.20	**3.26**

Guided Sampling

Classifier-guided Sampling

- We want to sample class-specific, high-quality samples

- A new forward process $\hat{q}(\mathbf{x}_{t+1}|\mathbf{x}_t, y)$

- Then, unknown reverse process $\hat{q}(\mathbf{x}_t|\mathbf{x}_{t+1}, y)$ becomes

$$\hat{q}(x_t|x_{t+1}, y) = \frac{q(x_t|x_{t+1})\hat{q}(y|x_t)}{\hat{q}(y|x_{t+1})} = Z \cdot q(x_t|x_{t+1})\hat{q}(y|x_t)$$

(complex derivation omitted)

- So we model the neural network

$$p_{\theta,\phi}(x_t|x_{t+1}, y) = Zp_\theta(x_t|x_{t+1})p_\phi(y|x_t)$$

Conditional Reverse Process

$$p_{\theta,\phi}(x_t|x_{t+1}, y) = Zp_\theta(x_t|x_{t+1})p_\phi(y|x_t)$$

$$p_\theta(x_t|x_{t+1}) = \mathcal{N}(\mu, \Sigma)$$

$$\log p_\theta(x_t|x_{t+1}) = -\frac{1}{2}(x_t - \mu)^T \Sigma^{-1}(x_t - \mu) + C$$

$$\begin{aligned}\log p_\phi(y|x_t) &\approx \log p_\phi(y|x_t)|_{x_t=\mu} + (x_t - \mu)\nabla_{x_t} \log p_\phi(y|x_t)|_{x_t=\mu} \\ &= (x_t - \mu)g + C_1\end{aligned}$$

$$\begin{aligned}\log(p_\theta(x_t|x_{t+1})p_\phi(y|x_t)) &\approx -\frac{1}{2}(x_t - \mu)^T \Sigma^{-1}(x_t - \mu) + (x_t - \mu)g + C_2 \\ &= -\frac{1}{2}(x_t - \mu - \Sigma g)^T \Sigma^{-1}(x_t - \mu - \Sigma g) + \frac{1}{2}g^T \Sigma g + C_2 \\ &= -\frac{1}{2}(x_t - \mu - \Sigma g)^T \Sigma^{-1}(x_t - \mu - \Sigma g) + C_3 \\ &= \log p(z) + C_4, z \sim \mathcal{N}(\mu + \Sigma g, \Sigma)\end{aligned}$$

Classifer-guided Sampling Algorithm

Algorithm 1 Classifier guided diffusion sampling, given a diffusion model $(\mu_\theta(x_t), \Sigma_\theta(x_t))$, classifier $\boxed{p_\phi(y|x_t)}$, and gradient scale s.

Need to train this separately with noisy samples

Input: class label y, gradient scale s
$x_T \leftarrow$ sample from $\mathcal{N}(0, \mathbf{I})$
for all t from T to 1 **do**
  $\mu, \Sigma \leftarrow \mu_\theta(x_t), \Sigma_\theta(x_t)$
  $x_{t-1} \leftarrow$ sample from $\mathcal{N}(\mu + s\Sigma \nabla_{x_t} \log p_\phi(y|x_t), \Sigma)$
end for
return x_0

Figure 3: Samples from an unconditional diffusion model with classifier guidance to condition on the class "Pembroke Welsh corgi". Using classifier scale 1.0 (left; FID: 33.0) does not produce convincing samples in this class, whereas classifier scale 10.0 (right; FID: 12.0) produces much more class-consistent images.

Classifier-guided DDIM

- Connection with Noise-Conditioned Score Matching (NCSM)
  - Sampling via Langevin dynamics
    - Iteratively follow the gradient of the log probability $\nabla_{\mathbf{x}} \log q(\mathbf{x})$
  - A score network $\mathbf{s}_\theta : \mathbb{R}^D \to \mathbb{R}^D$ is trained to estimate it, $\mathbf{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log q(\mathbf{x})$
  - Song & Ermon 2019 improved it to $\mathbf{s}_\theta(\mathbf{x}_t, t) \approx \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t)$
    - Train neural network to learn scores for samples with different levels of noise
    - Increasing noise level <==> DDPM forward process

$$\mathbf{s}_\theta(\mathbf{x}_t, t) \approx \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t) = \mathbb{E}_{q(\mathbf{x}_0)}[\nabla_{\mathbf{x}_t} q(\mathbf{x}_t|\mathbf{x}_0)] = \mathbb{E}_{q(\mathbf{x}_0)}\Big[-\frac{\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}}\Big] = -\frac{\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}}$$

Classifier-guided DDIM

Note that $\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$

Joint distribution of the score function:

$$\begin{aligned}
\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t, y) &= \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log q(y|\mathbf{x}_t) \\
&\approx -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) + \nabla_{\mathbf{x}_t} \log f_\phi(y|\mathbf{x}_t) \\
&= -\frac{1}{\sqrt{1-\bar{\alpha}_t}} (\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) - \sqrt{1-\bar{\alpha}_t} \nabla_{\mathbf{x}_t} \log f_\phi(y|\mathbf{x}_t))
\end{aligned}$$

New classifer-guided noise predictor:

$$\bar{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t) = \boldsymbol{\epsilon}_\theta(x_t, t) - \sqrt{1-\bar{\alpha}_t}\; w \nabla_{\mathbf{x}_t} \log f_\phi(y|\mathbf{x}_t)$$

Algorithm 2 Classifier guided DDIM sampling, given a diffusion model $\epsilon_\theta(x_t)$, classifier $f_\phi(y|x_t)$, and gradient scale s.

Input: class label y, gradient scale s
$x_T \leftarrow$ sample from $\mathcal{N}(0, \mathbf{I})$
for all t from T to 1 **do**
  $\hat{\epsilon} \leftarrow \epsilon_\theta(x_t) - \sqrt{1-\bar{\alpha}_t}\, \nabla_{x_t} \log f_\phi(y|x_t)$
  $x_{t-1} \leftarrow \sqrt{\bar{\alpha}_{t-1}} \left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\hat{\epsilon}}{\sqrt{\bar{\alpha}_t}} \right) + \sqrt{1-\bar{\alpha}_{t-1}}\hat{\epsilon}$
end for
return x_0

Classifier-free Guidance

- Class-conditioned diffusion without pre-trained classifier
- From NCSM-DDPM

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\,\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \qquad \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|y) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\,\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, y)$$

- Assume an implicit classifier

$$\begin{aligned}\nabla_{\mathbf{x}_t} \log p(y|\mathbf{x}_t) &= \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|y) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)\\ &= -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\Big(\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, y) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\Big)\end{aligned}$$

- From Classifier-guided DDIM

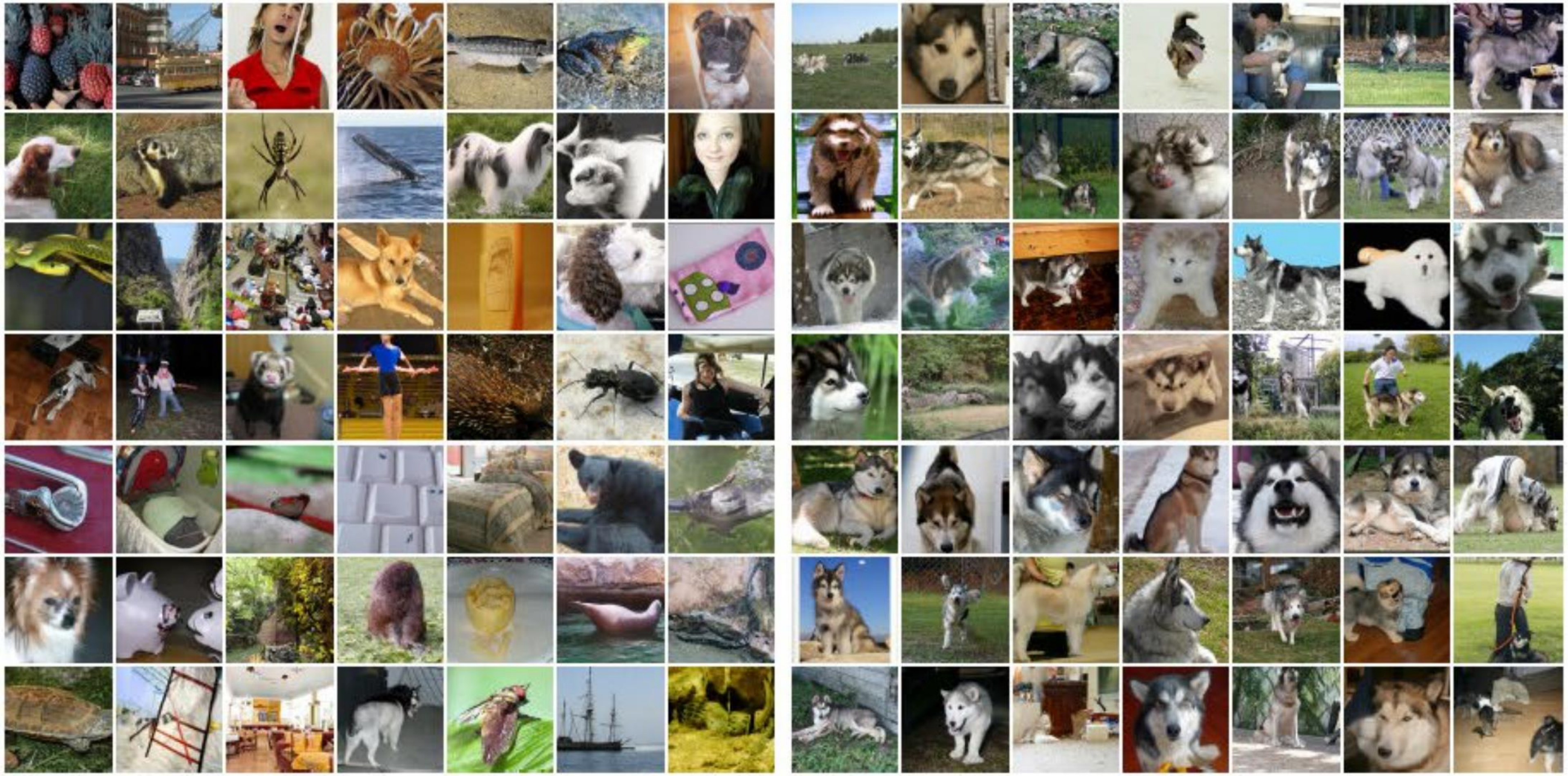
$$\begin{aligned}\bar{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t, y) &= \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, y) - \sqrt{1-\bar{\alpha}_t}\; w\nabla_{\mathbf{x}_t} \log p(y|\mathbf{x}_t)\\ &= \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, y) + w\big(\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, y) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\big)\\ &= (w+1)\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, y) - w\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\end{aligned}$$

Classifier-free Guidance

$$\bar{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t, y) = (w+1)\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, y) - w\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$$

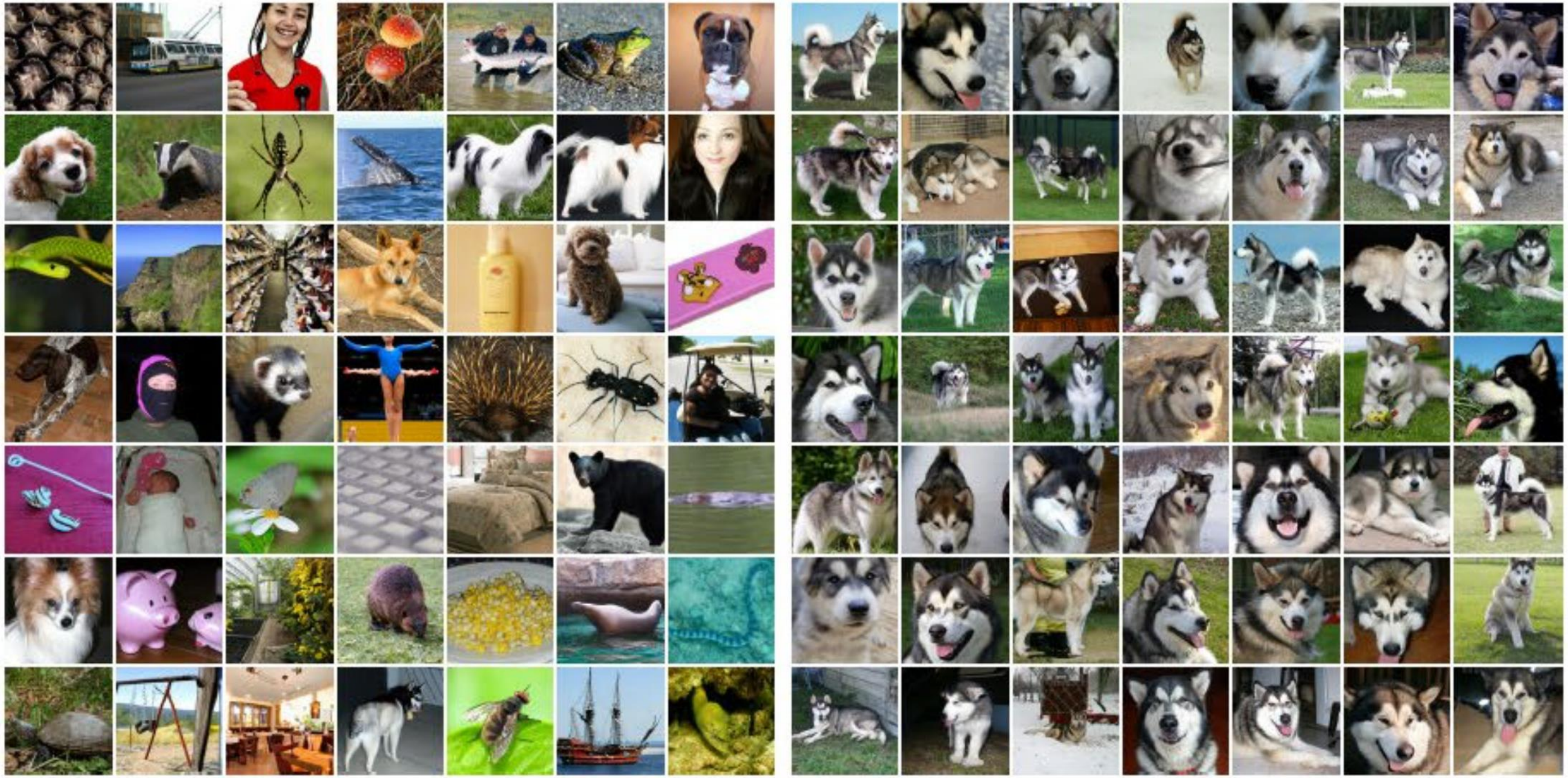
- Guided noise is a linear combination of conditioned/unconditioned noise
- We can train both $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, y)$ and $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$ with a single neural network
- Train with a null class (i.e. random sample) 10% of the training time
  - $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) = \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, y = \varnothing)$

Classifier-free Guidance



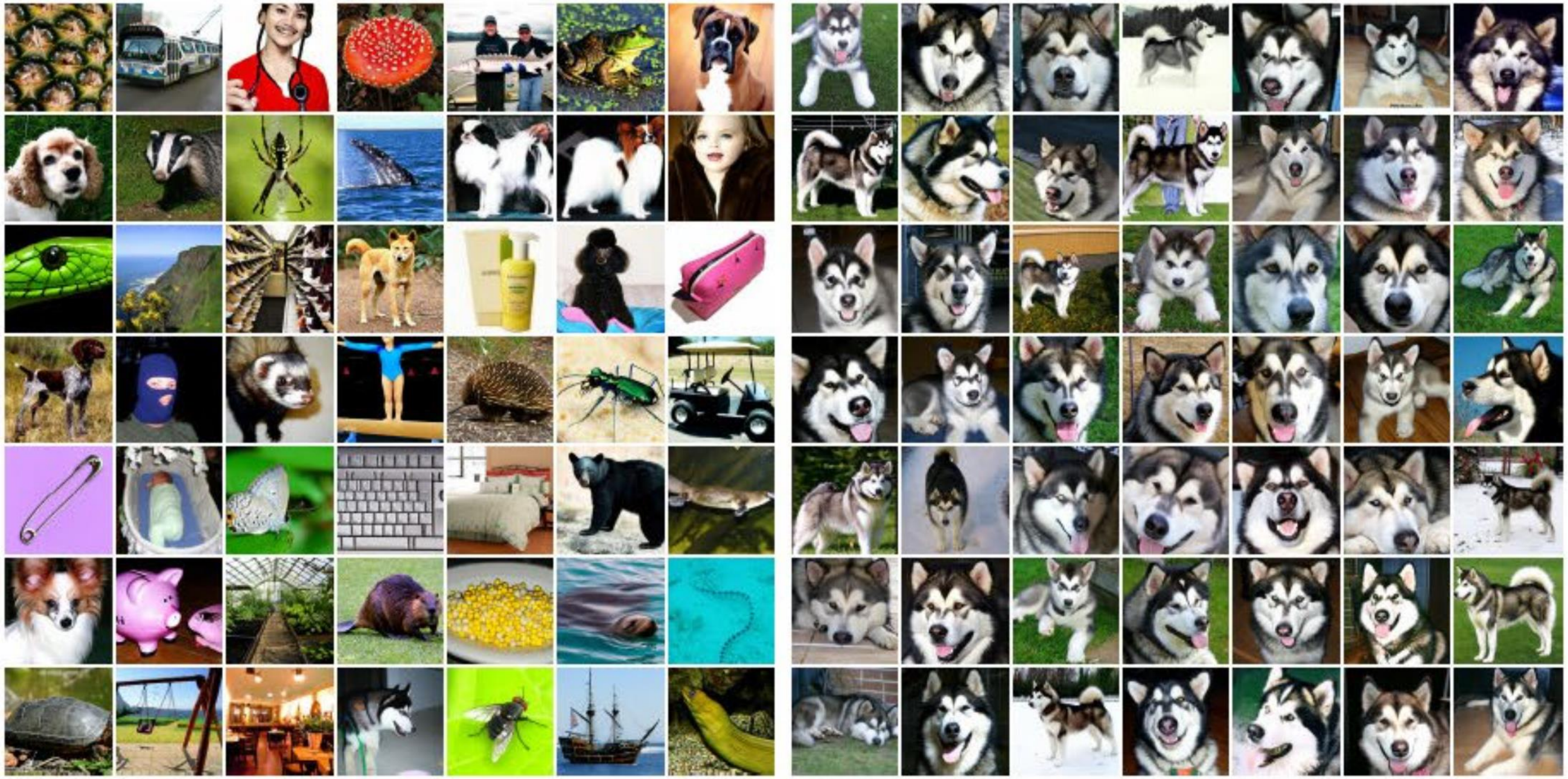
(a) Non-guided conditional sampling: FID=1.80, IS=53.71

Classifier-free Guidance



(b) Classifier-free guidance with $w = 1.0$: FID=12.6, IS=170.1

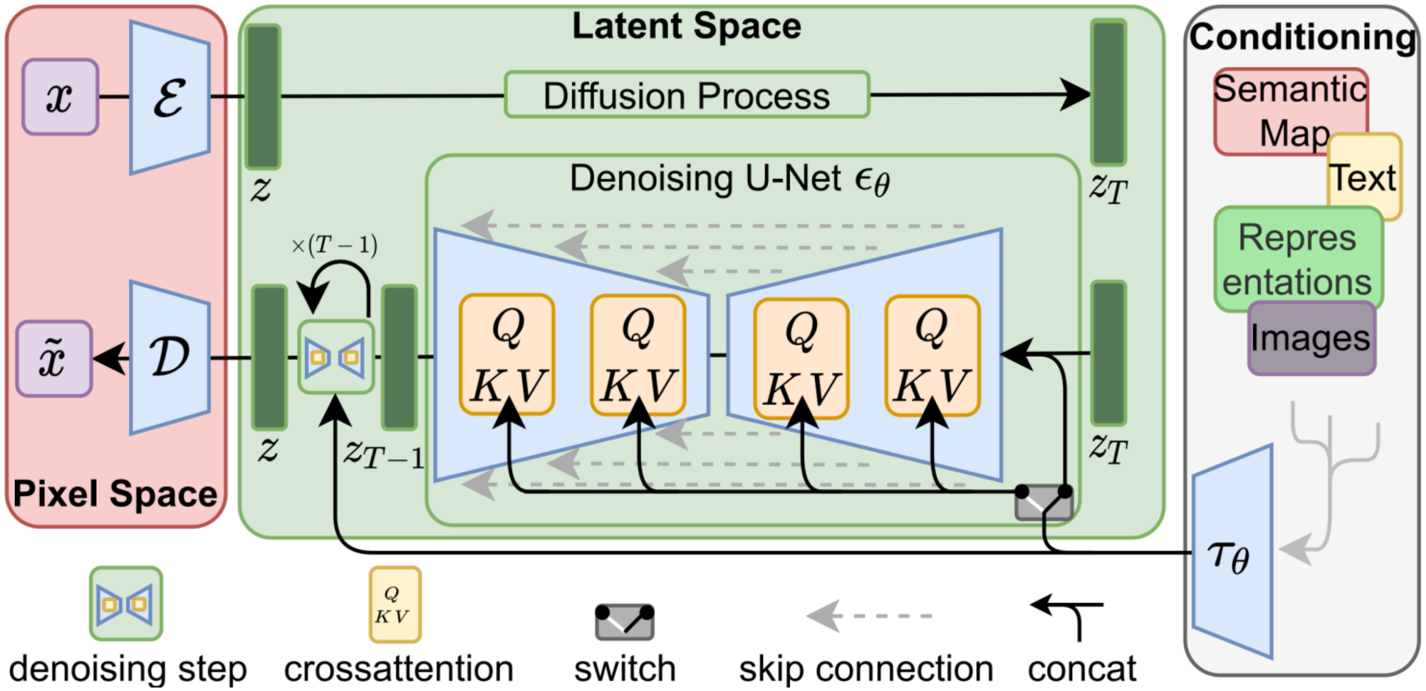
Classifier-free Guidance



(c) Classifier-free guidance with $w = 3.0$: FID=24.83, IS=250.4

Latent DDPM

- DDPM + Autoencoder
  - Compress high-resolution images using an AE
  - Train DDPM in the latent space



Quiz 2

- Which of the following is true?
  - You don’t need a separate classifier to do a guided diffusion
  - The reverse process is in essence moving towards the high-likelihood region
  - Guided diffusion guarantees an output image of the chosen class
  - Latent diffusion allows faster image generation than normal diffusion
  - You can use diffusion to generate data other than images

AI504: Programming for Artificial Intelligence

Week 14: Deep Diffusion Probabilistic Model

Edward Choi

Grad School of AI

edwardchoi@kaist.ac.kr

Variational Lower Bound

Note VAE objective: $\log p_\theta(\mathbf{x}) - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z}|\mathbf{x})) = \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}) - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z}))$

Goal: We want to minimize the negative log-likelihood.

$$
\begin{aligned}
& \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0)}\left[-\log p_\theta\left(\mathbf{x}_0\right)\right] \\
& \leq \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0)}\left[-\log p_\theta\left(\mathbf{x}_0\right)+D_{\mathrm{KL}}\left(q\left(\mathbf{x}_{1:T} \mid \mathbf{x}_0\right) \| p_\theta\left(\mathbf{x}_{1:T} \mid \mathbf{x}_0\right)\right)\right] \\
& =\mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0)}\left[-\log p_\theta\left(\mathbf{x}_0\right)+\mathbb{E}_{\mathbf{x}_{1:T} \sim q(\mathbf{x}_{1:T} | \mathbf{x}_0)}\left[\log \frac{q\left(\mathbf{x}_{1:T} \mid \mathbf{x}_0\right)}{p_\theta\left(\mathbf{x}_{0:T}\right) / p_\theta\left(\mathbf{x}_0\right)}\right]\right] \\
& =\mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0)}\left[-\log p_\theta\left(\mathbf{x}_0\right)+\mathbb{E}_{\mathbf{x}_{1:T} \sim q(\mathbf{x}_{1:T} | \mathbf{x}_0)}\left[\log \frac{q\left(\mathbf{x}_{1:T} \mid \mathbf{x}_0\right)}{p_\theta\left(\mathbf{x}_{0:T}\right)}+\log p_\theta\left(\mathbf{x}_0\right)\right]\right] \\
& =\mathbb{E}_{\mathbf{x}_{0:T} \sim q(\mathbf{x}_{0:T})}\left[\log \frac{q\left(\mathbf{x}_{1:T} \mid \mathbf{x}_0\right)}{p_\theta\left(\mathbf{x}_{0:T}\right)}\right]:=L_{\mathrm{VLB}}
\end{aligned}
$$

In other words, we can achieve the goal by minimizing L_{VLB}!

Return