

Deep Generative Models

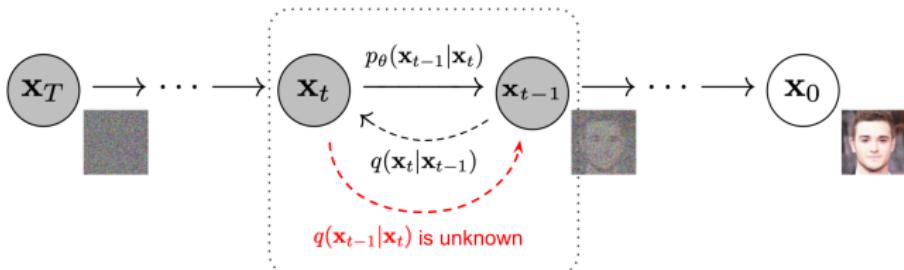
Lecture 10

Roman Isachenko

Moscow Institute of Physics and Technology
Yandex School of Data Analysis

2024, Autumn

Recap of previous lecture



Reverse process

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})q(\mathbf{x}_{t-1})}{q(\mathbf{x}_t)} \approx p(\mathbf{x}_{t-1}|\mathbf{x}_t, \theta) = \mathcal{N}(\boldsymbol{\mu}_{\theta,t}(\mathbf{x}_t), \sigma_{\theta,t}^2(\mathbf{x}_t))$$

Forward process

1. $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x});$
2. $\mathbf{x}_t = \sqrt{1 - \beta_t} \cdot \mathbf{x}_{t-1} + \sqrt{\beta_t} \cdot \epsilon;$
3. $\mathbf{x}_T \sim p_{\infty}(\mathbf{x}) = \mathcal{N}(0, \mathbf{I}).$

Reverse process

1. $\mathbf{x}_T \sim p_{\infty}(\mathbf{x}) = \mathcal{N}(0, \mathbf{I});$
2. $\mathbf{x}_{t-1} = \boldsymbol{\sigma}_{\theta,t}(\mathbf{x}_t) \cdot \epsilon + \boldsymbol{\mu}_{\theta,t}(\mathbf{x}_t);$
3. $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x});$

Note: The forward process does not have any learnable parameters!

Recap of previous lecture

- ▶ $\mathbf{z} = (\mathbf{x}_1, \dots, \mathbf{x}_T)$ is a latent variable.
- ▶ Variational posterior distribution

$$q(\mathbf{z}|\mathbf{x}) = q(\mathbf{x}_1, \dots, \mathbf{x}_T|\mathbf{x}_0) = \prod_{t=1}^T q(\mathbf{x}_t|\mathbf{x}_{t-1}).$$

- ▶ Generative distribution and prior

$$p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}); \quad p(\mathbf{z}|\boldsymbol{\theta}) = \prod_{t=2}^T p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta}) \cdot p(\mathbf{x}_T)$$

ELBO

$$\log p(\mathbf{x}|\boldsymbol{\theta}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x})} = \mathcal{L}_{\phi, \boldsymbol{\theta}}(\mathbf{x}) \rightarrow \max_{q, \boldsymbol{\theta}}$$

$$\begin{aligned} \mathcal{L}_{\phi, \boldsymbol{\theta}}(\mathbf{x}) &= \mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)} \log p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) - KL(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T)) - \\ &\quad - \sum_{t=2}^T \underbrace{\mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} KL(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta}))}_{\mathcal{L}_t} \end{aligned}$$

Recap of previous lecture

ELBO of gaussian diffusion model

$$\begin{aligned}\mathcal{L}_{\phi, \theta}(\mathbf{x}) = & \mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)} \log p(\mathbf{x}_0|\mathbf{x}_1, \theta) - KL(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T)) - \\ & - \sum_{t=2}^T \underbrace{\mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} KL(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p(\mathbf{x}_{t-1}|\mathbf{x}_t, \theta))}_{\mathcal{L}_t}\end{aligned}$$

$$\begin{aligned}q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) &= \mathcal{N}(\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}), \\ p(\mathbf{x}_{t-1}|\mathbf{x}_t, \theta) &= \mathcal{N}(\boldsymbol{\mu}_{\theta, t}(\mathbf{x}_t), \sigma_{\theta, t}^2(\mathbf{x}_t))\end{aligned}$$

Our assumption: $\sigma_{\theta, t}^2(\mathbf{x}_t) = \tilde{\beta}_t \mathbf{I}$.

$$\mathcal{L}_t = \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \left[\frac{1}{2\tilde{\beta}_t} \|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_{\theta, t}(\mathbf{x}_t)\|^2 \right]$$

Recap of previous lecture

$$\mathcal{L}_t = \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \left[\frac{1}{2\tilde{\beta}_t} \|\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \mu_{\theta,t}(\mathbf{x}_t)\|^2 \right]$$

Reparametrization

$$\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{1}{\sqrt{\alpha_t}} \cdot \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{\alpha_t(1 - \bar{\alpha}_t)}} \cdot \epsilon$$

$$\mu_{\theta,t}(\mathbf{x}_t) = \frac{1}{\sqrt{\alpha_t}} \cdot \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{\alpha_t(1 - \bar{\alpha}_t)}} \cdot \epsilon_{\theta,t}(\textcolor{teal}{x}_t)$$

$$\mathcal{L}_t = \mathbb{E}_{\epsilon \sim \mathcal{N}(0, \mathbf{I})} \left[\frac{(1 - \alpha_t)^2}{2\tilde{\beta}_t \alpha_t (1 - \bar{\alpha}_t)} \left\| \epsilon - \epsilon_{\theta,t}(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon) \right\|^2 \right]$$

At each step of reverse diffusion process we try to predict the noise ϵ that we used in the forward diffusion process!

Simplified objective

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{t \sim U\{2, T\}} \mathbb{E}_{\epsilon \sim \mathcal{N}(0, \mathbf{I})} \left\| \epsilon - \epsilon_{\theta,t}(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon) \right\|^2$$

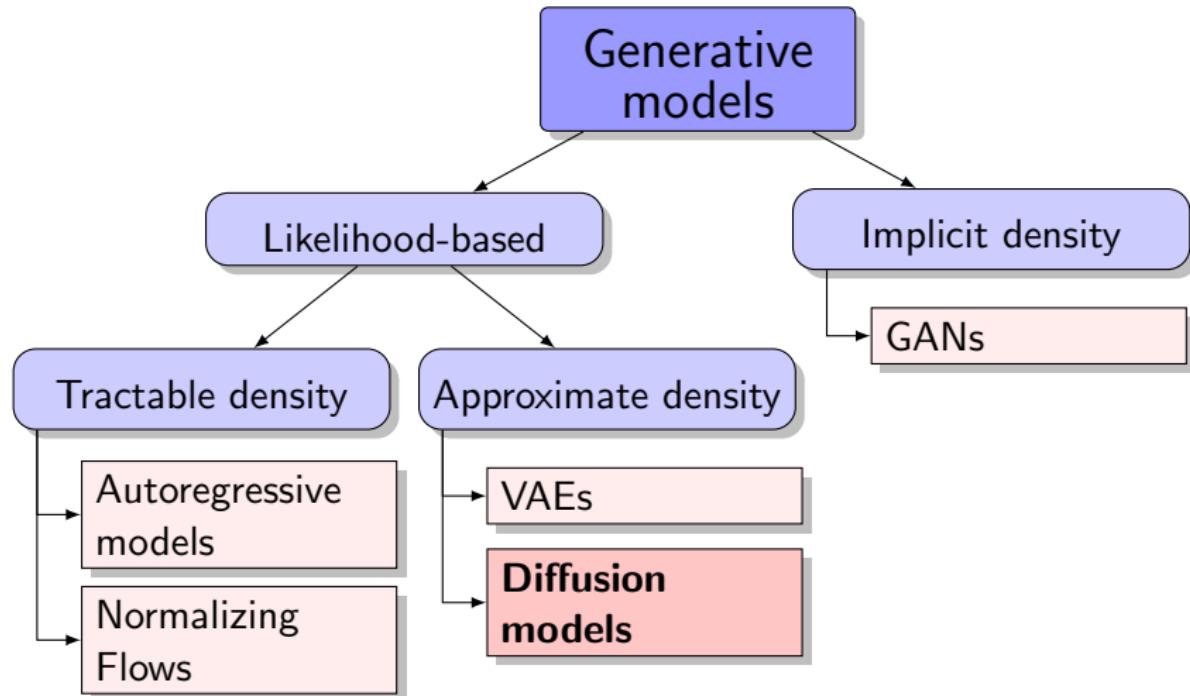
Outline

1. Denoising Diffusion Probabilistic Model (DDPM)
2. Denosing diffusion as score-based generative model
3. Guidance
 - Classifier guidance
 - Classifier-free guidance
4. Continuous-in-time normalizing flows

Outline

1. Denoising Diffusion Probabilistic Model (DDPM)
2. Denosing diffusion as score-based generative model
3. Guidance
 - Classifier guidance
 - Classifier-free guidance
4. Continuous-in-time normalizing flows

Generative models zoo



Denoising diffusion probabilistic model (DDPM)

DDPM is a VAE model

- ▶ Encoder is a fixed Gaussian Markov chain $q(\mathbf{x}_1, \dots, \mathbf{x}_T | \mathbf{x}_0)$.
- ▶ Latent variable is a hierarchical (in each step the dim. of the latent equals to the dim of the input).
- ▶ Decoder is a simple Gaussian model $p(\mathbf{x}_0 | \mathbf{x}_1, \theta)$.
- ▶ Prior distribution is given by parametric Gaussian Makov chain $p(\mathbf{x}_{t-1} | \mathbf{x}_t, \theta)$.

Forward process

1. $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x})$;
2. $\mathbf{x}_t = \sqrt{1 - \beta_t} \cdot \mathbf{x}_{t-1} + \sqrt{\beta_t} \cdot \epsilon$;
3. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$.

Reverse process

1. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$;
2. $\mathbf{x}_{t-1} = \sigma_{\theta, t}(\mathbf{x}_t) \cdot \epsilon + \mu_{\theta, t}(\mathbf{x}_t)$;
3. $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x})$;

Denoising diffusion probabilistic model (DDPM)

Training

1. Get the sample $\mathbf{x}_0 \sim \pi(\mathbf{x})$.
2. Sample timestamp $t \sim U\{1, T\}$ and the noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$.
3. Get noisy image $\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \cdot \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \cdot \epsilon$.
4. Compute loss $\mathcal{L}_{\text{simple}} = \|\epsilon - \epsilon_{\theta, t}(\mathbf{x}_t)\|^2$.

Sampling

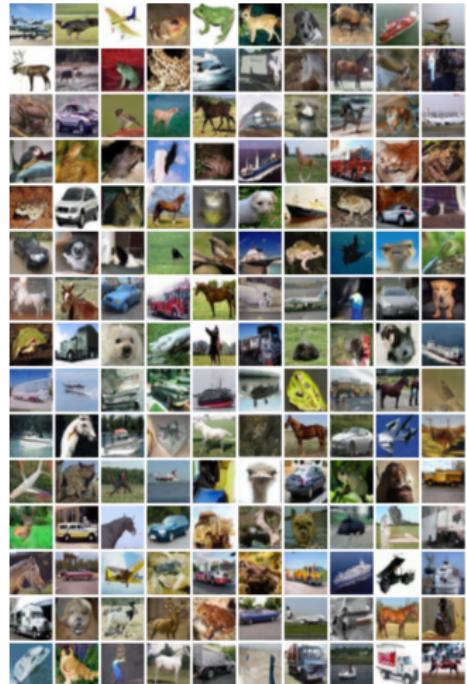
1. Sample $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$.
2. Compute mean of $p(\mathbf{x}_{t-1} | \mathbf{x}_t, \theta) = \mathcal{N}(\mu_{\theta, t}(\mathbf{x}_t), \tilde{\beta}_t \cdot \mathbf{I})$:

$$\mu_{\theta, t}(\mathbf{x}_t) = \frac{1}{\sqrt{\alpha_t}} \cdot \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{\alpha_t(1 - \bar{\alpha}_t)}} \cdot \epsilon_{\theta, t}(\mathbf{x}_t)$$

3. Get denoised image $\mathbf{x}_{t-1} = \mu_{\theta, t}(\mathbf{x}_t) + \sqrt{\tilde{\beta}_t} \cdot \epsilon$, where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$.

Denoising diffusion probabilistic model (DDPM)

Samples



Outline

1. Denoising Diffusion Probabilistic Model (DDPM)
2. Denosing diffusion as score-based generative model
3. Guidance
 - Classifier guidance
 - Classifier-free guidance
4. Continuous-in-time normalizing flows

Denoising diffusion as score-based generative model

DDPM objective

$$\mathcal{L}_t = \mathbb{E}_{\epsilon \sim \mathcal{N}(0, \mathbf{I})} \left[\frac{(1 - \alpha_t)^2}{2\tilde{\beta}_t \alpha_t (1 - \bar{\alpha}_t)} \left\| \epsilon_{\theta, t} (\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon) - \epsilon \right\|_2^2 \right]$$

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

$$\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0) = -\frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t} \cdot \mathbf{x}_0}{1 - \bar{\alpha}_t} = -\frac{\epsilon}{\sqrt{1 - \bar{\alpha}_t}}.$$

Let reparametrize our model:

$$\mathbf{s}_{\theta, t}(\mathbf{x}_t) = -\frac{\epsilon_{\theta, t}(\mathbf{x}_t)}{\sqrt{1 - \bar{\alpha}_t}}.$$

$$\mathcal{L}_t = \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \left[\frac{(1 - \alpha_t)^2}{2\tilde{\beta}_t \alpha_t} \left\| \mathbf{s}_{\theta, t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0) \right\|_2^2 \right]$$

Denoising diffusion as score-based generative model

DDPM objective

$$\mathbb{E}_{\pi(\mathbf{x}_0)} \mathbb{E}_{t \sim U\{1, T\}} \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \left[\frac{(1 - \alpha_t)^2}{2\tilde{\beta}_t \alpha_t} \left\| \mathbf{s}_{\theta, t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0) \right\|_2^2 \right]$$

In practice the coefficient is omitted.

NCSN objective

$$\mathbb{E}_{\pi(\mathbf{x}_0)} \mathbb{E}_{t \sim U\{1, T\}} \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \left\| \mathbf{s}_{\theta, \sigma_t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0) \right\|_2^2$$

Note: The objective of DDPM and NCSN is almost identical. But the difference in sampling scheme:

- ▶ NCSN uses annealed Langevin dynamics;
- ▶ DDPM uses ancestral sampling.

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

Outline

1. Denoising Diffusion Probabilistic Model (DDPM)
2. Denosing diffusion as score-based generative model
3. Guidance
 - Classifier guidance
 - Classifier-free guidance
4. Continuous-in-time normalizing flows

Guidance

- ▶ Throughout the whole course we have discussed unconditional generative models $p(\mathbf{x}|\theta)$.
- ▶ In practice the majority of the generative models are **conditional**: $p(\mathbf{x}|\mathbf{y}, \theta)$.
- ▶ Here \mathbf{y} could be the class label or **text** (for text-to-image models).



Кот ныряет в бассейн, как ребенок на обложке альбома Nevermind, реалистично



рука человека с пятью пальцами, ни четырьмя, ни шестью, а с 5 (пять) пальцами

Taxonomy of conditional tasks

In practice the popular task is to create a conditional model $\pi(x|y)$.

- ▶ y – class label, x – image \Rightarrow image conditional model.
- ▶ y – text prompt, x – image \Rightarrow text-to-image model.
- ▶ y – image, x – image \Rightarrow image-to-image model.
- ▶ y – image, x – text \Rightarrow image-to-text model (image captioning).
- ▶ y – sound, x – text \Rightarrow speech-to-text model (automatic speech recognition).
- ▶ y – English text, x – Russian text \Rightarrow sequence-to-sequence model (machine translation).
- ▶ $y = \emptyset$, x – image \Rightarrow image unconditional model.

Label guidance

Label: Ostrich (10th ImageNet class)



VQ-VAE (Proposed)

BigGAN deep

Text guidance

Prompt: a stained glass window of a panda eating bamboo

Left: $\gamma = 1$, Right: $\gamma = 3$.



Guidance

How to make conditional model $p(\mathbf{x}|\mathbf{y}, \theta)$?

- ▶ If we have **supervised** data $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$ we could treat \mathbf{y} as additional model input:
 - ▶ $p(x_j|\mathbf{x}_{1:j-1}, \mathbf{y}, \theta)$ for AR;
 - ▶ Encoder $q(\mathbf{z}|\mathbf{x}, \mathbf{y}, \phi)$ and decoder $p(\mathbf{x}|\mathbf{z}, \mathbf{y}, \theta)$ for VAE;
 - ▶ $G_\theta(\mathbf{z}, \mathbf{y})$ for NF and GAN;
 - ▶ $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}, \theta)$ for DDPM.
- ▶ If we have **unsupervised** data $\{\mathbf{x}_i\}_{i=1}^n$ we need to create the way to convert unconditional model $p(\mathbf{x}|\theta)$ to the conditional.

Diffusion **unsupervised** guidance

- ▶ Assume that we are given the distribution $q(\mathbf{y}|\mathbf{x}_0)$.
- ▶ **Forward process:** since we have already defined Markov chain, we have $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y}) = q(\mathbf{x}_t|\mathbf{x}_{t-1})$.
- ▶ **Reverse process:** let try to find reverse $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y})$.

Outline

1. Denoising Diffusion Probabilistic Model (DDPM)
2. Denosing diffusion as score-based generative model
3. Guidance
 - Classifier guidance**
 - Classifier-free guidance
4. Continuous-in-time normalizing flows

Classifier guidance

DDPM sampling

1. Sample $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$.
2. Compute mean of $p(\mathbf{x}_{t-1} | \mathbf{x}_t, \theta) = \mathcal{N}(\mu_{\theta,t}(\mathbf{x}_t), \sigma_t^2 \cdot \mathbf{I})$:

$$\mu_{\theta,t}(\mathbf{x}_t) = \frac{1}{\sqrt{\alpha_t}} \cdot \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{\alpha_t(1 - \bar{\alpha}_t)}} \cdot \epsilon_{\theta,t}(\mathbf{x}_t)$$

3. Get denoised image $\mathbf{x}_{t-1} = \mu_{\theta,t}(\mathbf{x}_t) + \sigma_t \cdot \epsilon$, where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$.

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \cdot \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{\alpha_t(1 - \bar{\alpha}_t)}} \cdot \epsilon_{\theta,t}(\mathbf{x}_t) + \sigma_t \cdot \epsilon$$

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

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \cdot \mathbf{x}_t + \frac{1 - \alpha_t}{\sqrt{\alpha_t}} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \theta) + \sigma_t \cdot \epsilon$$

Classifier guidance

Unconditional generation

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \cdot \mathbf{x}_t + \frac{1 - \alpha_t}{\sqrt{\alpha_t}} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \theta) + \sigma_t \cdot \epsilon$$

Conditional generation

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \cdot \mathbf{x}_t + \frac{1 - \alpha_t}{\sqrt{\alpha_t}} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \mathbf{y}, \theta) + \sigma_t \cdot \epsilon$$

Conditional distribution

$$\begin{aligned}\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \mathbf{y}, \theta) &= \nabla_{\mathbf{x}_t} \log \left(\frac{p(\mathbf{y} | \mathbf{x}_t) p(\mathbf{x}_t | \theta)}{p(\mathbf{y})} \right) \\ &= \nabla_{\mathbf{x}_t} \log p(\mathbf{y} | \mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \theta) \\ &= \nabla_{\mathbf{x}_t} \log p(\mathbf{y} | \mathbf{x}_t) - \frac{\epsilon_{\theta, t}(\mathbf{x}_t)}{\sqrt{1 - \bar{\alpha}_t}}\end{aligned}$$

Here $p(\mathbf{y} | \mathbf{x}_t)$ – classifier on noisy samples (we have to learn it separately).

Classifier guidance

Conditional distribution

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

Let parametrize $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \mathbf{y}, \theta) = -\frac{\epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y})}{\sqrt{1 - \bar{\alpha}_t}}$.

Classifier-corrected noise prediction

$$\epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y}) = \epsilon_{\theta,t}(\mathbf{x}_t) - \sqrt{1 - \bar{\alpha}_t} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{y} | \mathbf{x}_t)$$

Guidance scale

$$\epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y}) = \epsilon_{\theta,t}(\mathbf{x}_t) - \gamma \cdot \sqrt{1 - \bar{\alpha}_t} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{y} | \mathbf{x}_t)$$

Here we introduce **guidance scale γ** that controls the magnitude of the classifier guidance.

Classifier guidance

Classifier-corrected noise prediction

$$\epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y}) = \epsilon_{\theta,t}(\mathbf{x}_t) - \gamma \cdot \sqrt{1 - \bar{\alpha}_t} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t)$$

Conditional distribution

$$\frac{\epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y})}{\sqrt{1 - \bar{\alpha}_t}} = \frac{\epsilon_{\theta,t}(\mathbf{x}_t)}{\sqrt{1 - \bar{\alpha}_t}} - \gamma \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t)$$

$$\begin{aligned}\nabla_{\mathbf{x}_t}^{\gamma} \log p(\mathbf{x}_t|\mathbf{y}, \boldsymbol{\theta}) &= \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\boldsymbol{\theta}) + \gamma \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t) \\ &= \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\boldsymbol{\theta}) + \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t)^{\gamma} \\ &= \nabla_{\mathbf{x}_t} \log \left(\frac{p(\mathbf{y}|\mathbf{x}_t)^{\gamma} p(\mathbf{x}_t|\boldsymbol{\theta})}{Z} \right)\end{aligned}$$

Note: Guidance scale γ tries to sharpen the distribution $p(\mathbf{y}|\mathbf{x}_t)$.

Classifier guidance

- ▶ Train DDPM as usual.
- ▶ Train the classifier $p(\mathbf{y}|\mathbf{x}_t)$ on the noisy samples \mathbf{x}_t .

Guided sampling

1. Sample $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$.
2. Compute "corrected" $\epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y})$:

$$\epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y}) = \epsilon_{\theta,t}(\mathbf{x}_t) - \gamma \cdot \sqrt{1 - \bar{\alpha}_t} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t)$$

3. Compute mean of $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}, \theta) = \mathcal{N}(\mu_{\theta,t}(\mathbf{x}_t, \mathbf{y}), \sigma_t^2 \cdot \mathbf{I})$:

$$\mu_{\theta,t}(\mathbf{x}_t, \mathbf{y}) = \frac{1}{\sqrt{\alpha_t}} \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{\alpha_t(1 - \bar{\alpha}_t)}} \cdot \epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y})$$

4. Get denoised image $\mathbf{x}_{t-1} = \mu_{\theta,t}(\mathbf{x}_t, \mathbf{y}) + \sigma_t \cdot \epsilon$, where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$.

Outline

1. Denoising Diffusion Probabilistic Model (DDPM)
2. Denosing diffusion as score-based generative model
3. Guidance
 - Classifier guidance
 - Classifier-free guidance
4. Continuous-in-time normalizing flows

Classifier-free guidance

- ▶ Previous method requires training the additional classifier model $p(\mathbf{y}|\mathbf{x}_t)$ on the noisy data.
- ▶ Let try to avoid this requirement.

$$\nabla_{\mathbf{x}_t}^{\gamma} \log p(\mathbf{x}_t|\mathbf{y}, \theta) = \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\theta) + \gamma \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t)$$

$$\begin{aligned}\nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t) &= \nabla_{\mathbf{x}_t} \log \left(\frac{p(\mathbf{x}_t|\mathbf{y}, \theta)p(\mathbf{y})}{p(\mathbf{x}_t|\theta)} \right) \\ &= \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{y}, \theta) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\theta)\end{aligned}$$

$$\begin{aligned}\nabla_{\mathbf{x}_t}^{\gamma} \log p(\mathbf{x}_t|\mathbf{y}, \theta) &= \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\theta) + \gamma \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t) = \\ &= \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\theta) + \gamma \cdot (\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{y}, \theta) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\theta)) = \\ &= (1 - \gamma) \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\theta) + \gamma \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{y}, \theta)\end{aligned}$$

Note: In the case of $\gamma = 1$ we will get the identity statement.

Classifier-free guidance

$$\nabla_{\mathbf{x}_t}^{\gamma} \log p(\mathbf{x}_t | \mathbf{y}, \theta) = (1 - \gamma) \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \theta) + \gamma \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \mathbf{y}, \theta)$$

$$\frac{\epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y})}{\sqrt{1 - \bar{\alpha}_t}} = (1 - \gamma) \cdot \frac{\epsilon_{\theta,t}(\mathbf{x}_t)}{\sqrt{1 - \bar{\alpha}_t}} + \gamma \cdot \frac{\epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y})}{\sqrt{1 - \bar{\alpha}_t}}$$

Classifier-free-corrected noise prediction

$$\hat{\epsilon}_{\theta,t}(\mathbf{x}_t, \mathbf{y}) = \gamma \cdot \epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y}) + (1 - \gamma) \cdot \epsilon_{\theta,t}(\mathbf{x}_t)$$

- ▶ Train the single model $\epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y})$ on **supervised** data alternating with real conditioning \mathbf{y} and empty conditioning $\mathbf{y} = \emptyset$.
- ▶ Apply the model twice during inference.

Outline

1. Denoising Diffusion Probabilistic Model (DDPM)
2. Denosing diffusion as score-based generative model
3. Guidance
 - Classifier guidance
 - Classifier-free guidance
4. Continuous-in-time normalizing flows

Continuous-in-time normalizing flows

Discrete-in-time NF

Previously we assume that the time axis is discrete:

$$\mathbf{z}_{t+1} = \mathbf{f}_\theta(\mathbf{z}_t); \quad \log p(\mathbf{z}_{t+1}) = \log p(\mathbf{z}_t) - \log \left| \det \frac{\partial \mathbf{f}_\theta(\mathbf{z}_t)}{\partial \mathbf{z}_t} \right|.$$

Let assume the more general case of continuous time. It means that we will have the dynamic function $\mathbf{z}(t)$.

Continuous-in-time dynamics

Consider Ordinary Differential Equation (ODE)

$$\frac{d\mathbf{z}(t)}{dt} = \mathbf{f}_\theta(\mathbf{z}(t), t); \quad \text{with initial condition } \mathbf{z}(t_0) = \mathbf{z}_0.$$

$$\mathbf{z}(t_1) = \int_{t_0}^{t_1} \mathbf{f}_\theta(\mathbf{z}(t), t) dt + \mathbf{z}_0 \approx \text{ODESolve}(\mathbf{z}(t_0), \mathbf{f}_\theta, t_0, t_1).$$

Here we need to define the computational procedure

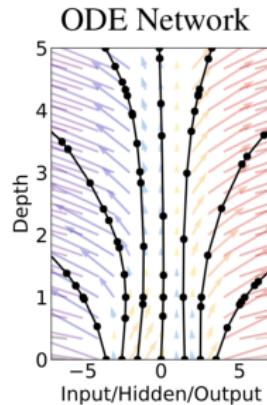
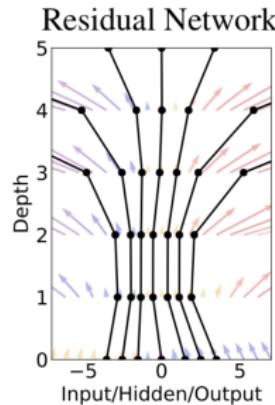
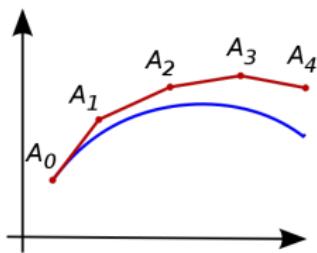
`ODESolve(z(t0), fθ, t0, t1)`.

Continuous-in-time normalizing flows

Euler update step

$$\frac{\mathbf{z}(t + \Delta t) - \mathbf{z}(t)}{\Delta t} = \mathbf{f}_{\theta}(\mathbf{z}(t), t) \Rightarrow \mathbf{z}(t + \Delta t) = \mathbf{z}(t) + \Delta t \cdot \mathbf{f}_{\theta}(\mathbf{z}(t), t)$$

Note: Euler method is the simplest version of ODESolve that is unstable in practice. It is possible to use more sophisticated methods (e.g. Runge-Kutta methods).

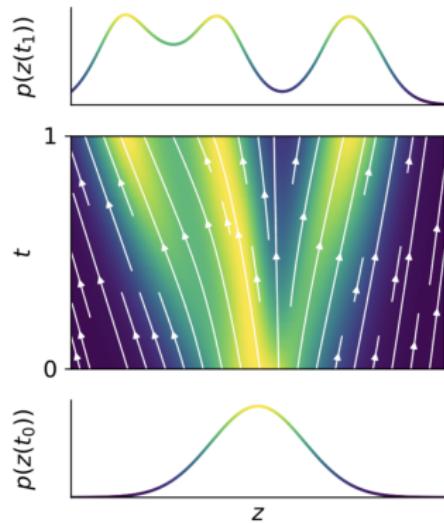


Continuous-in-time Normalizing Flows

Neural ODE

$$\frac{d\mathbf{z}(t)}{dt} = \mathbf{f}_{\theta}(\mathbf{z}(t), t); \quad \text{with initial condition } \mathbf{z}(t_0) = \mathbf{z}_0$$

- ▶ $\mathbf{z}(t_0)$ is a random variable with the density function $p(\mathbf{z}(t_0))$.
- ▶ $\mathbf{z}(t_1)$ is a random variable with the density function $p(\mathbf{z}(t_1))$.
- ▶ $p_t(\mathbf{z}) = p(\mathbf{z}, t)$ is the joint density function (probability path).
What is the difference between $p_t(\mathbf{z}(t))$ and $p_t(\mathbf{z})$?
- ▶ Let consider time interval $[t_0, t_1] = [0, 1]$ without loss of generality.



Continuous-in-time Normalizing Flows

Let say that $p_0(\mathbf{z})$ is the base distribution ($\mathcal{N}(0, \mathbf{I})$) and $p_1(\mathbf{z})$ is the desired model distribution $p(\mathbf{x}|\theta)$.

Theorem (Picard)

If \mathbf{f} is uniformly Lipschitz continuous in \mathbf{z} and continuous in t , then the ODE has a **unique** solution.

It means that we are able **uniquely revert** our ODE.

Forward and inverse transforms

$$\mathbf{x} = \mathbf{z}(1) = \mathbf{z}(0) + \int_0^1 \mathbf{f}_\theta(\mathbf{z}(t), t) dt$$

$$\mathbf{z} = \mathbf{z}(0) = \mathbf{z}(1) + \int_1^0 \mathbf{f}_\theta(\mathbf{z}(t), t) dt$$

Note: Unlike discrete-in-time NF, \mathbf{f} does not need to be bijective (uniqueness guarantees bijectivity).

Summary

- ▶ DDPM is a VAE model that tries to invert forward diffusion process using variational inference. DDPM is really slow, because we have to apply the model T times.
- ▶ Objective of DDPM is closely related to the noise conditioned score network and score matching.
- ▶ Conditional models use labels \mathbf{y} as the additional input. Majority of the modern generative models are conditional.
- ▶ Classifier guidance is the way to turn the unconditional model to the conditional one via the training additional classifier on the noisy data.
- ▶ Classifier-free guidance allows to avoid the training additional classifier to get the conditional model. It is widely used in practice.
- ▶ Continuous-in-time NF uses neural ODE to define continuous dynamic $\mathbf{z}(t)$. It has less functional restrictions.