Deep Generative Models

Lecture 10

Roman Isachenko

Moscow Institute of Physics and Technology
Yandex School of Data Analysis

2025, Autumn

Recap of Previous Lecture

Forward Process: Converts an arbitrary distribution $\pi(\mathbf{x})$ into the standard normal $\mathcal{N}(0, \mathbf{I})$ by incrementally adding noise.

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

Reverse Process: This intractable distribution can be well-approximated by a Gaussian (with unknown parameters) when β_t is sufficiently small.

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

Conditioned Reverse Process: This Gaussian, with known parameters, describes how to denoise a noisy image $\mathbf{x}_t$ when the final clean image $\mathbf{x}_0$ is known.

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

Ho J. Denoising Diffusion Probabilistic Models, 2020

Recap of Previous Lecture

- $\mathbf{z} = (\mathbf{x}_1, \dots, \mathbf{x}_T)$ represents the latent variables.
- 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 model 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

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

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

Das A. An introduction to Diffusion Probabilistic Models, blog post, 2021

Recap of Previous Lecture

ELBO of the Gaussian Diffusion Model

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

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

It is assumed that $\boldsymbol{\sigma}^2_{\boldsymbol{\theta},t}(\mathbf{x}_t) = \tilde{\beta}_t \mathbf{I}$.

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

Ho J. Denoising Diffusion Probabilistic Models, 2020

Recap of Previous Lecture

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

Reparameterization

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

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

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

At each step in the reverse diffusion process, our goal is to predict the noise $\boldsymbol{\epsilon}$ added by the forward process!

Simplified Objective

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

Ho J. Denoising Diffusion Probabilistic Models, 2020

Recap of Previous Lecture

Training DDPM

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

Sampling with DDPM

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

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

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

Ho J. Denoising Diffusion Probabilistic Models, 2020

Outline

1. DDPM as a Score-Based Generative Model
2. Guidance
   - Classifier Guidance
   - Classifier-Free Guidance
3. Continuous-Time Normalizing Flows

Outline

1. DDPM as a Score-Based Generative Model

2. Guidance
   Classifier Guidance
   Classifier-Free Guidance

3. Continuous-Time Normalizing Flows

Denoising Diffusion as a Score-Based Generative Model

DDPM Objective

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

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

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

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

We can reparameterize the model as:

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

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

Ho J. Denoising Diffusion Probabilistic Models, 2020

DDPM vs NCSN: Objectives

DDPM Objective

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

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\cdot\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\cdot\boldsymbol{\epsilon}$$

In practice, this coefficient is often omitted.

NCSN Objective

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

$$\mathbf{x}_t = \mathbf{x}_0 + \sigma_t\cdot\boldsymbol{\epsilon}$$

Maximizing the ELBO leads to the same objective as denoising score matching!

Ho J. Denoising Diffusion Probabilistic Models, 2020

DDPM vs NCSN: Sampling

DDPM Sampling (Ancestral Sampling)

$$
\begin{aligned}
\mathbf{x}_T &\sim \mathcal{N}(0, \mathbf{I}) \\
\mathbf{x}_{t-1} &= \boldsymbol{\mu}_{\boldsymbol{\theta},t}(\mathbf{x}_t) + \sigma_t \cdot \boldsymbol{\epsilon} \\
&= \frac{1}{\sqrt{\alpha_t}} \cdot \mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{\alpha_t(1-\bar{\alpha}_t)}} \cdot \boldsymbol{\epsilon}_{\boldsymbol{\theta},t}(\mathbf{x}_t) + \sigma_t \cdot \boldsymbol{\epsilon} \\
&= \frac{1}{\sqrt{1-\beta_t}} \cdot \mathbf{x}_t + \frac{\beta_t}{\sqrt{1-\beta_t}} \cdot \mathbf{s}_{\boldsymbol{\theta},t}(\mathbf{x}_t) + \sigma_t \cdot \boldsymbol{\epsilon}
\end{aligned}
$$

NCSN Sampling (Annealed Langevin Dynamics)

- Sample $\mathbf{x}_T^0 \sim \mathcal{N}(0, \sigma_T^2 \mathbf{I}) \approx q(\mathbf{x}_T)$.
- Perform L steps of Langevin dynamics:

$$
\mathbf{x}_t^l = \mathbf{x}_t^{l-1} + \frac{\eta_t}{2} \cdot \mathbf{s}_{\boldsymbol{\theta},\sigma_t}(\mathbf{x}_t^{l-1}) + \sqrt{\eta_t} \cdot \boldsymbol{\epsilon}_t^l.
$$

- Set $\mathbf{x}_{t-1}^0 = \mathbf{x}_t^L$ and move to the next σ_t.

Ho J. Denoising Diffusion Probabilistic Models, 2020

DDPM vs NCSN: Summary

Summary

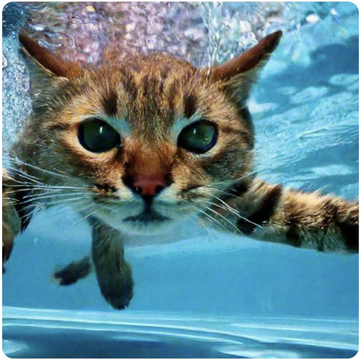
- Different Markov chains:
  - DDPM: $\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \cdot \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \cdot \boldsymbol{\epsilon}$;
  - NCSN: $\mathbf{x}_t = \mathbf{x}_0 + \sigma_t \cdot \boldsymbol{\epsilon}$.
  - One can generalize to $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\alpha_t \cdot \mathbf{x}_0, \sigma_t^2 \mathbf{I})$.
- The objectives coincide: ELBO $\equiv$ score-matching.
- The sampling procedures differ:
  - Ancestral sampling in DDPM;
  - Annealed Langevin dynamics for NCSN;
  - Hybrid approaches that combine both updates are possible.

Kingma D. et al. Variational Diffusion Models, 2021
Song Y. et al. Score-Based Generative Modeling through Stochastic Differential Equations, 2020

Outline

Guidance

- Up to now, we have focused on **unconditional** generative models $p(\mathbf{x}|\boldsymbol{\theta})$.
- In practice, most generative models are **conditional**: $p(\mathbf{x}|\mathbf{y}, \boldsymbol{\theta})$.
- Here, $\mathbf{y}$ might denote a class label or **text** (as in text-to-image tasks).



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

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

Taxonomy of Conditional Tasks

In practice, an important task is to construct a conditional model $\pi(\mathbf{x}|\mathbf{y})$.

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

Label Guidance

Label: Ostrich (10th ImageNet class)

VQ-VAE (Proposed) BigGAN deep

Razavi A., Oord A., et al. Generating Diverse High-Fidelity Images with VQ-VAE-2, 2019

Text Guidance

Prompt: a stained glass window of a panda eating bamboo
Left: $\gamma = 1$, Right: $\gamma = 3$.

Nichol A., et al. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models, 2022

Guidance in Generative Models

- Given **supervised** data $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, we can treat $\mathbf{y}$ as an additional input:
  - $p(x_j|\mathbf{x}_{1:j-1}, \mathbf{y}, \boldsymbol{\theta})$ for AR models;
  - Encoder $q(\mathbf{z}|\mathbf{x}, \mathbf{y}, \phi)$ and decoder $p(\mathbf{x}|\mathbf{z}, \mathbf{y}, \boldsymbol{\theta})$ for VAEs;
  - $G_{\boldsymbol{\theta}}(\mathbf{z}, \mathbf{y})$ for NFs and GANs;
  - $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}, \boldsymbol{\theta})$ for DDPMs.
- For **unsupervised** data $\{\mathbf{x}_i\}_{i=1}^n$, it becomes necessary to devise a way to transform the unconditional model $p(\mathbf{x}|\boldsymbol{\theta})$ into a conditional version.
- Being able to control the strength of guidance is especially valuable.

Types of Guidance

- **Classifier Guidance:**
  - Suitable for unsupervised data;
  - Utilizes an auxiliary classifier (which still requires supervision for classifier training).
- **Classifier-Free Guidance:**
  - Best suited for supervised data;
  - Does not rely on an additional classifier.

Outline

Classifier Guidance

DDPM Sampling

1. Sample $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$.
2. Generate the denoised image (unconditional generation):

$$\begin{aligned}\mathbf{x}_{t-1} &= \frac{1}{\sqrt{1-\beta_t}} \cdot \mathbf{x}_t + \frac{\beta_t}{\sqrt{1-\beta_t}} \cdot \mathbf{s}_{\theta,t}(\mathbf{x}_t) + \sigma_t \cdot \boldsymbol{\epsilon} \\ &= \frac{1}{\sqrt{1-\beta_t}} \cdot \mathbf{x}_t + \frac{\beta_t}{\sqrt{1-\beta_t}} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\boldsymbol{\theta}) + \sigma_t \cdot \boldsymbol{\epsilon}\end{aligned}$$

Conditional Generation

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

- Let us assume $\mathbf{y}$ is a class label.
- Suppose $p(\mathbf{y}|\mathbf{x}_t)$ (a classifier for noisy inputs) is available.

Dhariwal P., Nichol A. Diffusion Models Beat GANs on Image Synthesis, 2021

Classifier Guidance: Conditional Update

Conditional Generation

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

Conditional Distribution

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

Let's reparameterize: $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{y}, \boldsymbol{\theta}) = -\frac{\boldsymbol{\epsilon}_{\boldsymbol{\theta},t}(\mathbf{x}_t, \mathbf{y})}{\sqrt{1-\bar{\alpha}_t}}$.

Classifier-Corrected Noise Prediction

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

Dhariwal P., Nichol A. Diffusion Models Beat GANs on Image Synthesis, 2021

Classifier Guidance: Practical Implementation

Classifier-Corrected Noise Prediction

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

Guidance Scale

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

Here, the guidance scale γ adjusts the strength of classifier guidance.

Training

- Train the DDPM as before.
- Train an additional classifier $p(\mathbf{y}|\mathbf{x}_t)$ on noisy data.

Guided Sampling

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

Dhariwal P., Nichol A. Diffusion Models Beat GANs on Image Synthesis, 2021

Classifier Guidance: Distribution Sharpening

Classifier-Corrected Noise Prediction, with Scaling

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

Guidance-Scaled Conditional Distribution

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

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

Note: Increasing γ sharpens $p(\mathbf{y}|\mathbf{x}_t)$ (with Z independent of $\mathbf{x}_t$).

Dhariwal P., Nichol A. Diffusion Models Beat GANs on Image Synthesis, 2021

Outline

Classifier-Free Guidance

- The previous approach relies on training an additional classifier $p(\mathbf{y}|\mathbf{x}_t)$ for noisy images.
- We now introduce a method to sidestep this requirement.

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

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

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

Note: When $\gamma = 1$, the original identity is restored.

Ho J., Salimans T. Classifier-Free Diffusion Guidance, 2022

Classifier-Free Guidance: Formulation

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

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

Classifier-Free Corrected Noise Prediction

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

- Train a single model $\boldsymbol{\epsilon}_{\boldsymbol{\theta},t}(\mathbf{x}_t, \mathbf{y})$ using **supervised** data, alternating between conditioning and unconditional training (i.e., using $\mathbf{y} = \emptyset$).
- Apply the model for both conditioning modes during inference.

Guided Sampling

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

Ho J., Salimans T. Classifier-Free Diffusion Guidance, 2022

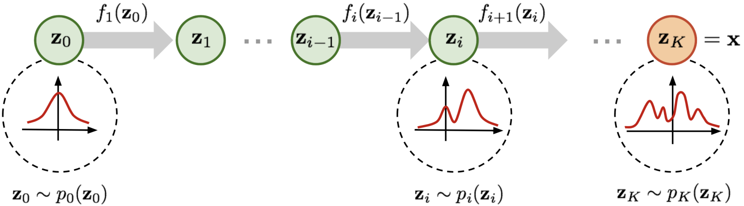
Outline

Discrete-Time Normalizing Flows

Change of Variable Theorem (CoV)

Let $\mathbf{x}$ be a random variable with density $p(\mathbf{x})$, and let $\mathbf{f} : \mathbb{R}^m \to \mathbb{R}^m$ be a differentiable and **invertible** transformation. If $\mathbf{z} = \mathbf{f}(\mathbf{x})$, $\mathbf{x} = \mathbf{f}^{-1}(\mathbf{z}) = \mathbf{g}(\mathbf{z})$, then

$$p(\mathbf{x}) = p(\mathbf{z})|\det(\mathbf{J_f})| = p(\mathbf{z}) \left|\det\left(\frac{\partial \mathbf{z}}{\partial \mathbf{x}}\right)\right| = p(\mathbf{f}(\mathbf{x})) \left|\det\left(\frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}}\right)\right|$$



$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log p(\mathbf{f}_K \circ \cdots \circ \mathbf{f}_1(\mathbf{x})) + \sum_{k=1}^{K} \log \left|\det\left(\frac{\partial \mathbf{f}_k}{\partial \mathbf{f}_{k-1}}\right)\right|.$$

---

https://lilianweng.github.io/lil-log/2018/10/13/flow-based-deep-generative-models.html

Discrete-Time Normalizing Flows: Towards Continuous Time

- Up to this point, we have considered discrete-time normalizing flows:
  $$\mathbf{x}_{t+1} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_t, t); \quad \log p(\mathbf{x}_{t+1}) = \log p(\mathbf{x}_t) - \log \left| \det \frac{\partial \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_t)}{\partial \mathbf{x}_t} \right|.$$
- Let us now move to the general case of continuous time, using a mapping $\mathbf{x}(t) : \mathbb{R} \to \mathbb{R}^m$ to describe continuous dynamics.

Continuous-Time Dynamics

Consider an Ordinary Differential Equation (ODE):

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

$$\mathbf{x}(t_1) = \int_{t_0}^{t_1} \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}(t), t) dt + \mathbf{x}_0$$

Here, $\mathbf{f}_{\boldsymbol{\theta}} : \mathbb{R}^m \times [t_0, t_1] \to \mathbb{R}^m$ is a vector field.

Numerical Solution of ODEs

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

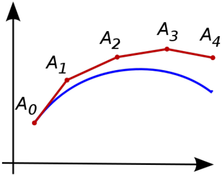
$$\mathbf{x}(t_1) = \int_{t_0}^{t_1} \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}(t), t)dt + \mathbf{x}_0 \approx \texttt{ODESolve}_f(\mathbf{x}_0, \boldsymbol{\theta}, t_0, t_1).$$

Here, we require the numerical routine $\texttt{ODESolve}_f(\mathbf{x}_0, \boldsymbol{\theta}, t_0, t_1)$.

Euler Update Step

$$\frac{\mathbf{x}(t + \Delta t) - \mathbf{x}(t)}{\Delta t} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}(t), t)$$

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \Delta t \cdot \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}(t), t)$$



Note: The Euler method is the simplest ODE solver but can be unstable in practice. More advanced schemes, such as Runge-Kutta, are typically employed.

Continuous-Time Normalizing Flows: Neural ODE

Neural ODE

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

Euler `ODESolve`

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \Delta t \cdot \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}(t), t)$$

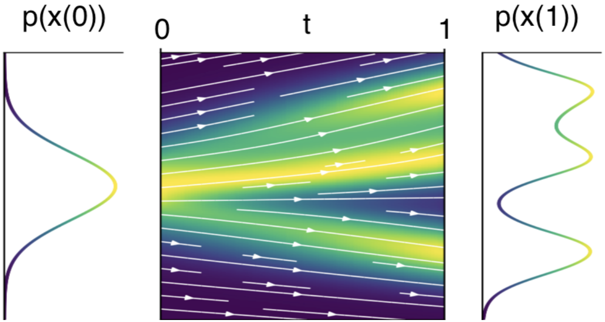
- Consider $[t_0, t_1] = [0, 1]$ for simplicity.
- If $\mathbf{x}(0)$ is a random variable with density $p_0(\mathbf{x})$,
- Then, for any t, $\mathbf{x}(t)$ is a random variable with density $p_t(\mathbf{x})$.

Chen R. T. Q. et al. Neural Ordinary Differential Equations, 2018

Continuous-Time Normalizing Flows: Intuition

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

- $p_t(\mathbf{x}) = p(\mathbf{x}, t)$ describes the **probability path** interpolating between $p_0(\mathbf{x})$ and $p_1(\mathbf{x})$.
- What is the difference between $p_t(\mathbf{x}(t))$ and $p_t(\mathbf{x})$?



Grathwohl W. et al. FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models, 2018

Continuous-Time Normalizing Flows: Reversibility

Theorem (Picard)

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

This guarantees the ODE is **uniquely reversible**.

$$\mathbf{x}(1) = \mathbf{x}(0) + \int_0^1 \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}(t), t)dt$$
$$\mathbf{x}(0) = \mathbf{x}(1) + \int_1^0 \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}(t), t)dt$$

Note: Unlike discrete-time flows, $\mathbf{f}$ need not be invertible (uniqueness ensures bijection).
How can we compute $p_t(\mathbf{x})$ at arbitrary t?

Chen R. T. Q. et al. Neural Ordinary Differential Equations, 2018

Summary

- DDPM and NCSN are intimately connected at the objective level.
- Classifier guidance provides a technique to turn an unconditional model into a conditional one by training an auxiliary classifier on noisy data.
- Classifier-free guidance removes the need for such a classifier, yielding a practical recipe now widely used.
- Continuous-time normalizing flows leverage neural ODEs to define continuous-time trajectories $\mathbf{x}(t)$, relaxing many constraints of discrete-time flows.
- If $\mathbf{x}_0$ is a random variable, this yields a **probability path** $p_t(\mathbf{x})$ as time evolves.