Deep Generative Models

Lecture 13

Roman Isachenko

Moscow Institute of Physics and Technology
Yandex School of Data Analysis

2025, Autumn

Recap of Previous Lecture

Flow Matching (FM)

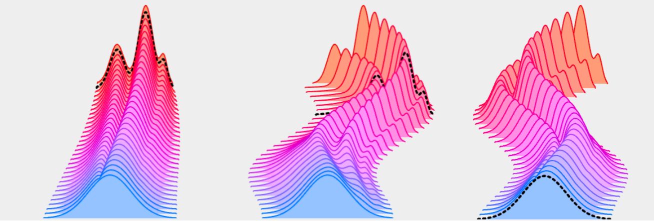
$$\mathbb{E}_{t \sim U[0,1]} \mathbb{E}_{\mathbf{x} \sim p_t(\mathbf{x})} \left\| \mathbf{f}(\mathbf{x}, t) - \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}, t) \right\|^2 \to \min_{\boldsymbol{\theta}}$$

Conditional Flow Matching (CFM)

$$\mathbb{E}_{t \sim U[0,1]} \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})} \mathbb{E}_{\mathbf{x} \sim p_t(\mathbf{x}|\mathbf{z})} \left\| \mathbf{f}(\mathbf{x}, \mathbf{z}, t) - \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}, t) \right\|^2 \to \min_{\boldsymbol{\theta}}$$

Theorem

If $\text{supp}(p_t(\mathbf{x})) = \mathbb{R}^m$, then the optimal value of the FM objective equals the optimum for CFM.



Tong A., et al. Improving and Generalizing Flow-Based Generative Models with Minibatch Optimal Transport, 2023

Recap of Previous Lecture

$$\begin{tikzcd}
\overbrace{\forall z \in Z}^{\textbf{choice 1}} \quad \overbrace{p(x|t,z)}^{\textbf{choice 2}} \arrow[r, leftrightarrow, "\text{determines explicitly}", "\text{(by continuity eq.)}"'] \arrow[d, "\text{determines (by expectation over } z\text{)}"'] & u^{\text{cond}}(x,t,z) \arrow[d, "\text{allows computing (Thm. 1)}"] \\
p(x|t) \arrow[r, leftrightarrow, "\text{determines implicitly}", "\text{(by continuity eq.)}"'] & u(x,t)
\end{tikzcd}$$

Constraints

$$p(\mathbf{x}) = \mathcal{N}(0, \mathbf{I}) = \mathbb{E}_{p(\mathbf{z})} p_0(\mathbf{x}|\mathbf{z}); \quad p_{\text{data}}(\mathbf{x}) = \mathbb{E}_{p(\mathbf{z})} p_1(\mathbf{x}|\mathbf{z}).$$

- How should we choose the conditioning latent variable $\mathbf{z}$?
- How can we define $p_t(\mathbf{x}|\mathbf{z})$ so that it meets the constraints?

Gaussian Conditional Probability Path

$$p_t(\mathbf{x}|\mathbf{z}) = \mathcal{N}\left(\boldsymbol{\mu}_t(\mathbf{z}), \boldsymbol{\sigma}_t^2(\mathbf{z})\right)$$

$$\mathbf{x}_t = \boldsymbol{\mu}_t(\mathbf{z}) + \boldsymbol{\sigma}_t(\mathbf{z}) \odot \mathbf{x}_0, \quad \mathbf{x}_0 \sim p_0(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$$

image credit: A Visual Dive into Conditional Flow Matching

Recap of Previous Lecture

Gaussian Conditional Probability Path

$$p_t(\mathbf{x}|\mathbf{z}) = \mathcal{N}\left(\boldsymbol{\mu}_t(\mathbf{z}), \boldsymbol{\sigma}_t^2(\mathbf{z})\right); \quad \mathbf{x}_t = \boldsymbol{\mu}_t(\mathbf{z}) + \boldsymbol{\sigma}_t(\mathbf{z}) \odot \mathbf{x}_0$$

$$\mathbf{f}(\mathbf{x}, \mathbf{z}, t) = \boldsymbol{\mu}_t'(\mathbf{z}) + \frac{\boldsymbol{\sigma}_t'(\mathbf{z})}{\boldsymbol{\sigma}_t(\mathbf{z})} \odot (\mathbf{x} - \boldsymbol{\mu}_t(\mathbf{z}))$$

Conditioning Latent Variable

Let's choose $\mathbf{z} = \mathbf{x}_1$. Then $p(\mathbf{z}) = p_1(\mathbf{x}_1)$.

$$p_t(\mathbf{x}) = \int p_t(\mathbf{x}|\mathbf{x}_1) p_1(\mathbf{x}_1) d\mathbf{x}_1$$

We must ensure the boundary constraints:

$$\begin{cases} p(\mathbf{x}) = \mathbb{E}_{p(\mathbf{z})} p_0(\mathbf{x}|\mathbf{z}); (= \mathcal{N}(0, \mathbf{I})) \\ p_{\text{data}}(\mathbf{x}) = \mathbb{E}_{p(\mathbf{z})} p_1(\mathbf{x}|\mathbf{z}). \end{cases} \quad \Rightarrow \quad \begin{cases} p_0(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}(0, \mathbf{I}); \\ p_1(\mathbf{x}|\mathbf{x}_1) = \delta(\mathbf{x} - \mathbf{x}_1). \end{cases}$$

Lipman Y., et al. Flow Matching for Generative Modeling, 2022

Recap of Previous Lecture

$$p_0(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}(0, \mathbf{I}); \quad p_1(\mathbf{x}|\mathbf{x}_1) = \delta(\mathbf{x} - \mathbf{x}_1).$$

Gaussian Conditional Probability Path

$$p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(\boldsymbol{\mu}_t(\mathbf{x}_1), \boldsymbol{\sigma}_t^2(\mathbf{x}_1)\right); \quad \mathbf{x}_t = \boldsymbol{\mu}_t(\mathbf{x}_1) + \boldsymbol{\sigma}_t(\mathbf{x}_1) \odot \mathbf{x}_0.$$

Let's consider straight conditional paths:

$$\begin{cases} \boldsymbol{\mu}_t(\mathbf{x}_1) = t\mathbf{x}_1; \\ \boldsymbol{\sigma}_t(\mathbf{x}_1) = 1 - t. \end{cases} \quad \Rightarrow \quad \begin{cases} p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(t\mathbf{x}_1, (1-t)^2\mathbf{I}\right); \\ \mathbf{x}_t = t\mathbf{x}_1 + (1-t)\mathbf{x}_0. \end{cases}$$

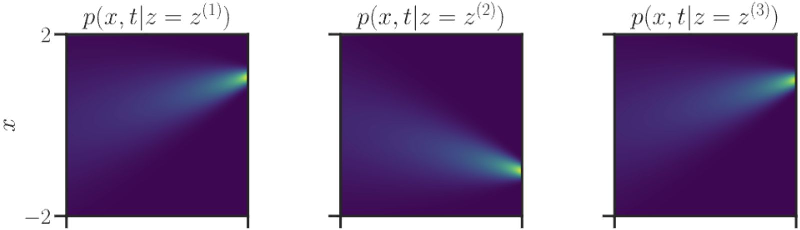


image credit: A Visual Dive into Conditional Flow Matching

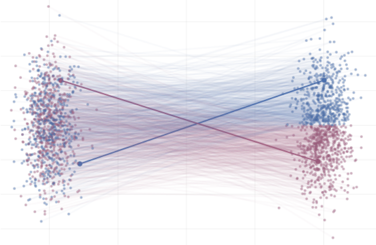
Recap of Previous Lecture

$$p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(t\mathbf{x}_1, (1-t)^2\mathbf{I}\right); \quad \mathbf{x}_t = t\mathbf{x}_1 + (1-t)\mathbf{x}_0$$

$$\mathbf{f}(\mathbf{x}_t, \mathbf{x}_1, t) = \frac{\mathbf{x}_1 - \mathbf{x}_t}{1-t} = \mathbf{x}_1 - \mathbf{x}_0$$

$$\mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\mathbb{E}_{\mathbf{x}\sim p_t(\mathbf{x}|\mathbf{z})} \left\|\mathbf{f}(\mathbf{x}, \mathbf{z}, t) - \mathbf{f}_\theta(\mathbf{x}, t)\right\|^2 =$$
$$= \mathbb{E}_{\mathbf{x}_1\sim p_{\text{data}}(\mathbf{x})}\mathbb{E}_{\mathbf{x}_0\sim\mathcal{N}(0,\mathbf{I})} \left\|(\mathbf{x}_1 - \mathbf{x}_0) - \mathbf{f}_\theta\left(t\mathbf{x}_1 + (1-t)\mathbf{x}_0, t\right)\right\|^2$$

- $\mathbf{f}(\mathbf{x}_t, \mathbf{x}_1, t)$ defines straight lines between $p_{\text{data}}(\mathbf{x})$ and $\mathcal{N}(0, \mathbf{I})$.
- The **marginal** path $p_t(\mathbf{x})$ does not give straight lines.



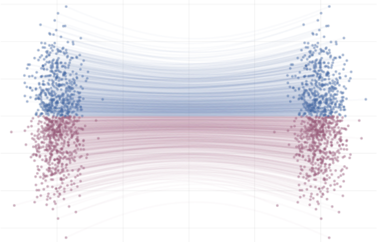


image credit: https://mlg.eng.cam.ac.uk/blog/2024/01/20/flow-matching.html

Recap of Previous Lecture

$$\mathbb{E}_{t\sim U[0,1]}\mathbb{E}_{\mathbf{x}_1\sim p_{\text{data}}(\mathbf{x})}\mathbb{E}_{\mathbf{x}_0\sim\mathcal{N}(0,\mathbf{I})}\left\|(\mathbf{x}_1-\mathbf{x}_0)-\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_t,t)\right\|^2 \to \min_{\boldsymbol{\theta}}$$

Training

1. Sample $\mathbf{x}_1 \sim p_{\text{data}}(\mathbf{x})$.
2. Sample time $t \sim U[0,1]$ and $\mathbf{x}_0 \sim \mathcal{N}(0,\mathbf{I})$.
3. Obtain the noisy image $\mathbf{x}_t = t\mathbf{x}_1 + (1-t)\mathbf{x}_0$.
4. Compute the loss $\mathcal{L} = \|(\mathbf{x}_1 - \mathbf{x}_0) - \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}, t)\|^2$.

Sampling

1. Sample $\mathbf{x}_0 \sim \mathcal{N}(0,\mathbf{I})$.
2. Solve the ODE to obtain $\mathbf{x}_1$:

$$\mathbf{x}_1 = \texttt{ODESolve}_f(\mathbf{x}_0, \boldsymbol{\theta}, t_0 = 0, t_1 = 1)$$

Lipman Y., et al. Flow Matching for Generative Modeling, 2022

Recap of Previous Lecture

Let us choose $\mathbf{z} = (\mathbf{x}_0, \mathbf{x}_1)$. Then $p(\mathbf{z}) = p(\mathbf{x}_0, \mathbf{x}_1) = p_0(\mathbf{x}_0)p_1(\mathbf{x}_1)$.

$$p_0(\mathbf{x}|\mathbf{x}_0, \mathbf{x}_1) = \delta(\mathbf{x} - \mathbf{x}_0); \quad p_1(\mathbf{x}|\mathbf{x}_0, \mathbf{x}_1) = \delta(\mathbf{x} - \mathbf{x}_1)$$

Gaussian Conditional Probability Path

$$p_t(\mathbf{x}|\mathbf{x}_0, \mathbf{x}_1) = \mathcal{N}\left(\boldsymbol{\mu}_t(\mathbf{x}_0, \mathbf{x}_1), \boldsymbol{\sigma}_t^2(\mathbf{x}_0, \mathbf{x}_1)\right); \quad \mathbf{x}_t = \boldsymbol{\mu}_t(\mathbf{x}_0, \mathbf{x}_1) + \boldsymbol{\sigma}_t(\mathbf{x}_0, \mathbf{x}_1) \odot \boldsymbol{\epsilon}$$

Let's consider straight conditional paths:

$$\boldsymbol{\mu}_t(\mathbf{x}_1) = t\mathbf{x}_1 + (1-t)\mathbf{x}_0 \quad \boldsymbol{\sigma}_t(\mathbf{x}_1) = \epsilon$$

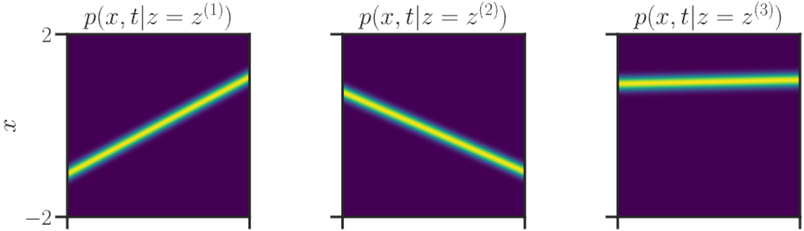


image credit: A Visual Dive into Conditional Flow Matching

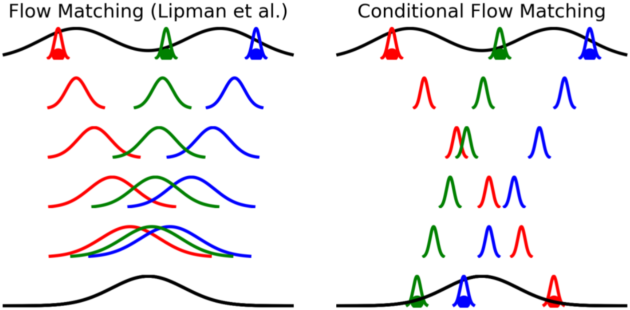
Recap of Previous Lecture

Endpoint conditioning

$$
\begin{aligned}
\mathbf{z} &= \mathbf{x}_1 \\
p_t(\mathbf{x}|\mathbf{x}_1) &= \mathcal{N}\left(t\mathbf{x}_1, (1-t)^2\mathbf{I}\right) \\
\mathbf{x}_t &= t\mathbf{x}_1 + (1-t)\mathbf{x}_0
\end{aligned}
$$

Pair conditioning

$$
\begin{aligned}
\mathbf{z} &= (\mathbf{x}_0, \mathbf{x}_1) \\
p_t(\mathbf{x}|\mathbf{x}_0, \mathbf{x}_1) &= \mathcal{N}\left(t\mathbf{x}_1 + (1-t)\mathbf{x}_0, \epsilon^2\mathbf{I}\right) \\
\mathbf{x}_t &= t\mathbf{x}_1 + (1-t)\mathbf{x}_0
\end{aligned}
$$



Tong A., et al. Improving and Generalizing Flow-Based Generative Models with Minibatch Optimal Transport, 2023

Recap of Previous Lecture

- This conditioning allows us to transport any distribution $p_0(\mathbf{x})$ to any distribution $p_1(\mathbf{x})$.
- It's possible to apply this approach to paired tasks, e.g., style transfer.

Training Procedure

1. Sample $(\mathbf{x}_0, \mathbf{x}_1) \sim p(\mathbf{x}_0, \mathbf{x}_1)$.
2. Sample time $t \sim U[0, 1]$.
3. Compute the noisy image $\mathbf{x}_t = t\mathbf{x}_1 + (1 - t)\mathbf{x}_0$.
4. Compute the loss $\mathcal{L} = \|(\mathbf{x}_1 - \mathbf{x}_0) - \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}, t)\|^2$.

Sampling

1. Sample $\mathbf{x}_0 \sim p_0(\mathbf{x})$.
2. Solve the ODE to obtain $\mathbf{x}_1$:

$$\mathbf{x}_1 = \texttt{ODESolve}_f(\mathbf{x}_0, \boldsymbol{\theta}, t_0 = 0, t_1 = 1)$$

Lipman Y., et al. Flow Matching for Generative Modeling, 2022

Outline

1. Link between Flow Matching and Score-Based Models

2. Discrete Diffusion Models
   Forward Discrete Process
   Reverse Discrete Diffusion

Outline

1. Link between Flow Matching and Score-Based Models

2. Discrete Diffusion Models
   - Forward Discrete Process
   - Reverse Discrete Diffusion

Score-Based Generative Models through SDEs

Training

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x}(0))}\mathbb{E}_{t\sim U[0,1]}\mathbb{E}_{q(\mathbf{x}(t)|\mathbf{x}(0))}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}(t),t)-\nabla_{\mathbf{x}(t)}\log q(\mathbf{x}(t)|\mathbf{x}(0))\right\|_2^2$$

Lipman Y., et al. Flow Matching for Generative Modeling, 2022

Score-Based Generative Models through SDEs

Training

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x}(0))}\mathbb{E}_{t\sim U[0,1]}\mathbb{E}_{q(\mathbf{x}(t)|\mathbf{x}(0))}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}(t), t) - \nabla_{\mathbf{x}(t)} \log q(\mathbf{x}(t)|\mathbf{x}(0))\right\|_2^2$$

Variance Exploding SDE (NCSN)

$$q(\mathbf{x}(t)|\mathbf{x}(0)) = \mathcal{N}\left(\mathbf{x}(0), \left[\sigma^2(t) - \sigma^2(0)\right] \cdot \mathbf{I}\right), \quad \sigma(0) = 0$$

Variance Preserving SDE (DDPM)

$$q(\mathbf{x}(t)|\mathbf{x}(0)) = \mathcal{N}\left(\mathbf{x}(0)\alpha(t), \left(1 - \alpha(t)^2\right) \cdot \mathbf{I}\right); \quad \alpha(t) = e^{-\frac{1}{2}\int_0^t \beta(s)ds}$$

Lipman Y., et al. Flow Matching for Generative Modeling, 2022

Score-Based Generative Models through SDEs

Training

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x}(0))}\mathbb{E}_{t\sim U[0,1]}\mathbb{E}_{q(\mathbf{x}(t)|\mathbf{x}(0))}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}(t), t) - \nabla_{\mathbf{x}(t)} \log q(\mathbf{x}(t)|\mathbf{x}(0))\right\|_2^2$$

Variance Exploding SDE (NCSN)

$$q(\mathbf{x}(t)|\mathbf{x}(0)) = \mathcal{N}\left(\mathbf{x}(0), \left[\sigma^2(t) - \sigma^2(0)\right]\cdot \mathbf{I}\right), \quad \sigma(0) = 0$$

Variance Preserving SDE (DDPM)

$$q(\mathbf{x}(t)|\mathbf{x}(0)) = \mathcal{N}\left(\mathbf{x}(0)\alpha(t), \left(1 - \alpha(t)^2\right)\cdot \mathbf{I}\right); \quad \alpha(t) = e^{-\frac{1}{2}\int_0^t \beta(s)ds}$$

Flow matching uses reverse time direction:

$$p_t(\mathbf{x}|\mathbf{x}_1) = q_{1-t}(\mathbf{x}|\mathbf{x}_0 = \mathbf{x}_1)$$

Lipman Y., et al. Flow Matching for Generative Modeling, 2022

Score-Based Generative Models through SDEs

$$p_t(\mathbf{x}_t|\mathbf{x}_1) = q_{1-t}(\mathbf{x}_{1-t}|\mathbf{x}_0 = \mathbf{x}_1)$$

VE (NCSN): $p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(\mathbf{x}_1, \sigma_{1-t}^2 \cdot \mathbf{I}\right)$

VP (DDPM): $p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(\alpha_{1-t}\mathbf{x}_1, \left(1 - \alpha_{1-t}^2\right) \cdot \mathbf{I}\right)$

Score-Based Generative Models through SDEs

$$p_t(\mathbf{x}_t|\mathbf{x}_1) = q_{1-t}(\mathbf{x}_{1-t}|\mathbf{x}_0 = \mathbf{x}_1)$$

$$\textbf{VE (NCSN):}\ p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(\mathbf{x}_1, \sigma_{1-t}^2 \cdot \mathbf{I}\right)$$

$$\textbf{VP (DDPM):}\ p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(\alpha_{1-t}\mathbf{x}_1, \left(1 - \alpha_{1-t}^2\right) \cdot \mathbf{I}\right)$$

Flow Matching Probability Path

$$p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(t\mathbf{x}_1, (1-t)^2\mathbf{I}\right); \quad \mathbf{f}(\mathbf{x}_t, \mathbf{x}_1, t) = \frac{\mathbf{x}_1 - \mathbf{x}_t}{1-t}$$

$$\frac{d\mathbf{x}_t}{dt} = \mathbf{f}(\mathbf{x}_t, \mathbf{x}_1, t) = \boldsymbol{\mu}_t'(\mathbf{x}_1) + \frac{\boldsymbol{\sigma}_t'(\mathbf{x}_1)}{\boldsymbol{\sigma}_t(\mathbf{x}_1)} \odot (\mathbf{x}_t - \boldsymbol{\mu}_t(\mathbf{x}_1))$$

Score-Based Generative Models through SDEs

$$p_t(\mathbf{x}_t|\mathbf{x}_1) = q_{1-t}(\mathbf{x}_{1-t}|\mathbf{x}_0 = \mathbf{x}_1)$$

$$\textbf{VE (NCSN):}\ p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(\mathbf{x}_1, \sigma_{1-t}^2 \cdot \mathbf{I}\right)$$

$$\textbf{VP (DDPM):}\ p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(\alpha_{1-t}\mathbf{x}_1, \left(1 - \alpha_{1-t}^2\right) \cdot \mathbf{I}\right)$$

Flow Matching Probability Path

$$p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(t\mathbf{x}_1, (1-t)^2\mathbf{I}\right); \quad \mathbf{f}(\mathbf{x}_t, \mathbf{x}_1, t) = \frac{\mathbf{x}_1 - \mathbf{x}_t}{1-t}$$

$$\frac{d\mathbf{x}_t}{dt} = \mathbf{f}(\mathbf{x}_t, \mathbf{x}_1, t) = \boldsymbol{\mu}_t'(\mathbf{x}_1) + \frac{\boldsymbol{\sigma}_t'(\mathbf{x}_1)}{\boldsymbol{\sigma}_t(\mathbf{x}_1)} \odot (\mathbf{x}_t - \boldsymbol{\mu}_t(\mathbf{x}_1))$$

Let's derive the conditional vector fields for VE (NCSN) and VP (DDPM).

Flow Matching vs. Score-Based SDE Models

$$\frac{d\mathbf{x}_t}{dt} = \mathbf{f}(\mathbf{x}_t, \mathbf{x}_1, t) = \boldsymbol{\mu}'_t(\mathbf{x}_1) + \frac{\boldsymbol{\sigma}'_t(\mathbf{x}_1)}{\boldsymbol{\sigma}_t(\mathbf{x}_1)} \odot (\mathbf{x}_t - \boldsymbol{\mu}_t(\mathbf{x}_1))$$

Lipman Y., et al. Flow Matching for Generative Modeling, 2022

Flow Matching vs. Score-Based SDE Models

$$\frac{d\mathbf{x}_t}{dt} = \mathbf{f}(\mathbf{x}_t, \mathbf{x}_1, t) = \boldsymbol{\mu}'_t(\mathbf{x}_1) + \frac{\boldsymbol{\sigma}'_t(\mathbf{x}_1)}{\boldsymbol{\sigma}_t(\mathbf{x}_1)} \odot (\mathbf{x}_t - \boldsymbol{\mu}_t(\mathbf{x}_1))$$

Variance Exploding SDE Probability Path

$$p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(\mathbf{x}_1, \sigma^2_{1-t}\mathbf{I}\right) \quad \Rightarrow \quad \mathbf{f}(\mathbf{x}_t, \mathbf{x}_1, t) = -\frac{\sigma'_{1-t}}{\sigma_{1-t}}(\mathbf{x}_t - \mathbf{x}_1)$$

Lipman Y., et al. Flow Matching for Generative Modeling, 2022

Flow Matching vs. Score-Based SDE Models

$$\frac{d\mathbf{x}_t}{dt} = \mathbf{f}(\mathbf{x}_t, \mathbf{x}_1, t) = \boldsymbol{\mu}'_t(\mathbf{x}_1) + \frac{\boldsymbol{\sigma}'_t(\mathbf{x}_1)}{\boldsymbol{\sigma}_t(\mathbf{x}_1)} \odot (\mathbf{x}_t - \boldsymbol{\mu}_t(\mathbf{x}_1))$$

Variance Exploding SDE Probability Path

$$p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(\mathbf{x}_1, \sigma^2_{1-t}\mathbf{I}\right) \quad \Rightarrow \quad \mathbf{f}(\mathbf{x}_t, \mathbf{x}_1, t) = -\frac{\sigma'_{1-t}}{\sigma_{1-t}}(\mathbf{x}_t - \mathbf{x}_1)$$

Variance Preserving SDE Probability Path

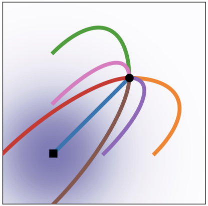
$$p_t(\mathbf{x}|\mathbf{x}_1) = \mathcal{N}\left(\alpha_{1-t}\mathbf{x}_1, (1-\alpha^2_{1-t})\mathbf{I}\right) \Rightarrow \mathbf{f}(\mathbf{x}_t, \mathbf{x}_1, t) = \frac{\alpha'_{1-t}}{1-\alpha^2_{1-t}} \cdot (\alpha_{1-t}\mathbf{x}_t - \mathbf{x}_1)$$

Thus, VE/VP SDE models correspond to particular choices of the Gaussian probability path within the flow matching framework.

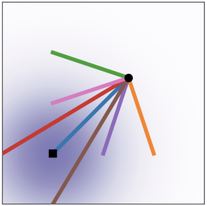
Lipman Y., et al. Flow Matching for Generative Modeling, 2022

Flow Matching vs. Score-Based SDE Models

Trajectories



Diffusion

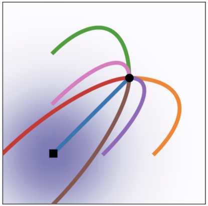


OT

Lipman Y., et al. Flow Matching for Generative Modeling, 2022

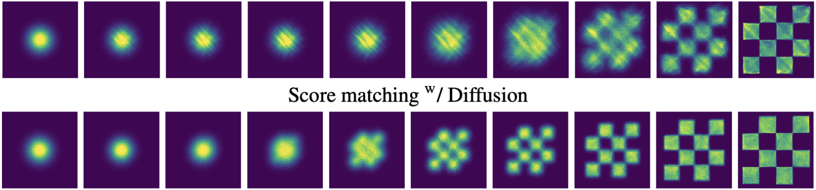
Flow Matching vs. Score-Based SDE Models

Trajectories



Diffusion

OT



Score matching w/ Diffusion

Flow Matching w/ OT

Lipman Y., et al. Flow Matching for Generative Modeling, 2022

Outline

1. Link between Flow Matching and Score-Based Models

2. Discrete Diffusion Models
    - Forward Discrete Process
    - Reverse Discrete Diffusion

Discrete or Continuous Diffusion Models?

Reminder: Diffusion models define a forward corruption process and a reverse denoising process. Previously, we studied diffusion models with continuous states $\mathbf{x}(t) \in \mathbb{R}^m$.

Continuous state space

- **Discrete time** $t \in \{0, 1, \ldots, T\}$ $\Rightarrow$ **DDPM / NCSN**.
- **Continuous time** $t \in [0, 1]$ $\Rightarrow$ **Score-based SDE models**.

Discrete or Continuous Diffusion Models?

Reminder: Diffusion models define a forward corruption process and a reverse denoising process. Previously, we studied diffusion models with continuous states $\mathbf{x}(t) \in \mathbb{R}^m$.

Continuous state space

- **Discrete time** $t \in \{0, 1, \ldots, T\}$ $\Rightarrow$ **DDPM / NCSN**.
- **Continuous time** $t \in [0, 1]$ $\Rightarrow$ **Score-based SDE models**.

Now we turn to diffusion over discrete-value states $\mathbf{x}(t) \in \{1, \ldots, K\}^m$.

Discrete state space

- **Discrete time** $t \in \{0, 1, \ldots, T\}$.
- **Continuous time** $t \in [0, 1]$.

Let's discuss why we need discrete diffusion models.

Why Discrete Diffusion Models?

While autoregressive (AR) models dominate discrete-data domains (e.g., text or sequences), they have fundamental limitations.

https://aaronlou.com/blog/2024/discrete-diffusion/

Why Discrete Diffusion Models?

While autoregressive (AR) models dominate discrete-data domains (e.g., text or sequences), they have fundamental limitations.

Key advantages of discrete diffusion

- **Parallel generation:** diffusion enables sampling all tokens simultaneously, unlike AR's strictly left-to-right process.

https://aaronlou.com/blog/2024/discrete-diffusion/

Why Discrete Diffusion Models?

While autoregressive (AR) models dominate discrete-data domains (e.g., text or sequences), they have fundamental limitations.

Key advantages of discrete diffusion

- **Parallel generation:** diffusion enables sampling all tokens simultaneously, unlike AR's strictly left-to-right process.
- **Flexible infilling:**. diffusion can mask arbitrary parts of a sequence and reconstruct them, rather than generating only from prefix to suffix.

https://aaronlou.com/blog/2024/discrete-diffusion/

Why Discrete Diffusion Models?

While autoregressive (AR) models dominate discrete-data domains (e.g., text or sequences), they have fundamental limitations.

Key advantages of discrete diffusion

- **Parallel generation:** diffusion enables sampling all tokens simultaneously, unlike AR's strictly left-to-right process.
- **Flexible infilling:**. diffusion can mask arbitrary parts of a sequence and reconstruct them, rather than generating only from prefix to suffix.
- **Robustness:** diffusion avoids the "exposure bias" caused by teacher forcing in AR training.

https://aaronlou.com/blog/2024/discrete-diffusion/

Why Discrete Diffusion Models?

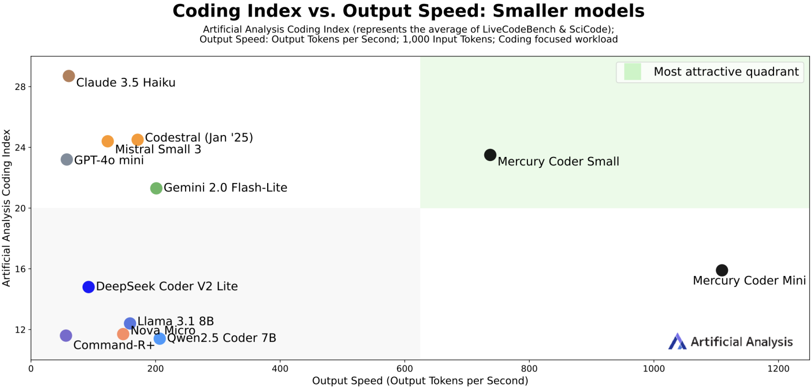
While autoregressive (AR) models dominate discrete-data domains (e.g., text or sequences), they have fundamental limitations.

Key advantages of discrete diffusion

- **Parallel generation:** diffusion enables sampling all tokens simultaneously, unlike AR's strictly left-to-right process.
- **Flexible infilling:**. diffusion can mask arbitrary parts of a sequence and reconstruct them, rather than generating only from prefix to suffix.
- **Robustness:** diffusion avoids the "exposure bias" caused by teacher forcing in AR training.
- **Unified framework:** diffusion generalizes naturally to discrete domains that do not suit continuous Gaussian noise.

https://aaronlou.com/blog/2024/discrete-diffusion/

2025 – Big Bang of Discrete Diffusion Models



Khanna S. et al. Mercury: Ultra-fast language models based on diffusion, 2025.

Outline

1. Link between Flow Matching and Score-Based Models

2. Discrete Diffusion Models

   Forward Discrete Process

   Reverse Discrete Diffusion

Forward Discrete Process

Continuous Diffusion Markov Chain

In continuous diffusion, the forward Markov chain is defined by progressively corrupting data with Gaussian noise:

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

Austin J. et al. Structured denoising diffusion models in discrete state-spaces, 2021.

Forward Discrete Process

Continuous Diffusion Markov Chain

In continuous diffusion, the forward Markov chain is defined by progressively corrupting data with Gaussian noise:

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

Discrete Diffusion Markov Chain

For discrete data, we instead define a Markov chain over categorical states:

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

Austin J. et al. Structured denoising diffusion models in discrete state-spaces, 2021.

Forward Discrete Process

Continuous Diffusion Markov Chain

In continuous diffusion, the forward Markov chain is defined by progressively corrupting data with Gaussian noise:

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

Discrete Diffusion Markov Chain

For discrete data, we instead define a Markov chain over categorical states:

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

- Each $\mathbf{x}_t \in \{0,1\}^K$ is a **one-hot vector** encoding the categorical state (it is just one token).
- What is the transition matrix $\mathbf{Q}_t$?

Austin J. et al. Structured denoising diffusion models in discrete state-spaces, 2021.

Forward Process over Time

Transition Matrix

$\mathbf{Q}_t \in [0,1]^{K\times K}$ is a **transition matrix** where each column gives transition probabilities from one state to all others, and columns sum to 1:

$$[\mathbf{Q}_t]_{ij} = q(x_t = i | x_{t-1} = j), \qquad \sum_{i=1}^{K} [\mathbf{Q}_t]_{ij} = 1.$$

Austin J. et al. Structured denoising diffusion models in discrete state-spaces, 2021.

Forward Process over Time

Transition Matrix

$\mathbf{Q}_t \in [0,1]^{K\times K}$ is a **transition matrix** where each column gives transition probabilities from one state to all others, and columns sum to 1:

$$[\mathbf{Q}_t]_{ij} = q(x_t = i | x_{t-1} = j), \qquad \sum_{i=1}^{K} [\mathbf{Q}_t]_{ij} = 1.$$

- The forward diffusion gradually destroys information through repeated random transitions.

Austin J. et al. Structured denoising diffusion models in discrete state-spaces, 2021.

Forward Process over Time

Transition Matrix

$\mathbf{Q}_t \in [0,1]^{K\times K}$ is a **transition matrix** where each column gives transition probabilities from one state to all others, and columns sum to 1:

$$[\mathbf{Q}_t]_{ij} = q(x_t = i | x_{t-1} = j), \qquad \sum_{i=1}^{K} [\mathbf{Q}_t]_{ij} = 1.$$

- The forward diffusion gradually destroys information through repeated random transitions.
- Applying the transition t times yields the marginal distribution:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathrm{Cat}(\mathbf{Q}_{1:t}\mathbf{x}_0), \qquad \mathbf{Q}_{1:t} = \mathbf{Q}_t \mathbf{Q}_{t-1} \cdots \mathbf{Q}_1.$$

Austin J. et al. Structured denoising diffusion models in discrete state-spaces, 2021.

Forward Process over Time

Transition Matrix

$\mathbf{Q}_t \in [0,1]^{K\times K}$ is a **transition matrix** where each column gives transition probabilities from one state to all others, and columns sum to 1:

$$[\mathbf{Q}_t]_{ij} = q(x_t = i | x_{t-1} = j), \qquad \sum_{i=1}^{K} [\mathbf{Q}_t]_{ij} = 1.$$

- The forward diffusion gradually destroys information through repeated random transitions.
- Applying the transition t times yields the marginal distribution:

$$q(\mathbf{x}_t | \mathbf{x}_0) = \mathrm{Cat}(\mathbf{Q}_{1:t}\mathbf{x}_0), \qquad \mathbf{Q}_{1:t} = \mathbf{Q}_t \mathbf{Q}_{t-1} \cdots \mathbf{Q}_1.$$

- As $t \to T$, the process drives the data toward a stationary distribution.

Austin J. et al. Structured denoising diffusion models in discrete state-spaces, 2021.

Forward Process over Time

Transition Matrix

$\mathbf{Q}_t \in [0,1]^{K \times K}$ is a **transition matrix** where each column gives transition probabilities from one state to all others, and columns sum to 1:

$$[\mathbf{Q}_t]_{ij} = q(x_t = i | x_{t-1} = j), \qquad \sum_{i=1}^{K} [\mathbf{Q}_t]_{ij} = 1.$$

- The forward diffusion gradually destroys information through repeated random transitions.
- Applying the transition t times yields the marginal distribution:

$$q(\mathbf{x}_t | \mathbf{x}_0) = \mathrm{Cat}(\mathbf{Q}_{1:t}\mathbf{x}_0), \qquad \mathbf{Q}_{1:t} = \mathbf{Q}_t \mathbf{Q}_{t-1} \cdots \mathbf{Q}_1.$$

- As $t \to T$, the process drives the data toward a stationary distribution.
- We design the transition matrices $\mathbf{Q}_t$ to achieve this behavior.

Austin J. et al. Structured denoising diffusion models in discrete state-spaces, 2021.

Transition Matrix

- The choice of $\mathbf{Q}_t$ determines how information is erased and what the stationary distribution becomes.

Austin J. et al. Structured denoising diffusion models in discrete state-spaces, 2021.

Transition Matrix

- The choice of $\mathbf{Q}_t$ determines how information is erased and what the stationary distribution becomes.
- $\mathbf{Q}_t$ and $\mathbf{Q}_{1:t}$ should be easy to compute for each t.

Austin J. et al. Structured denoising diffusion models in discrete state-spaces, 2021.

Transition Matrix

- The choice of $\mathbf{Q}_t$ determines how information is erased and what the stationary distribution becomes.
- $\mathbf{Q}_t$ and $\mathbf{Q}_{1:t}$ should be easy to compute for each t.

Common choices

- **Uniform diffusion**

$$\mathbf{Q}_t = (1-\beta_t)\mathbf{I} + \beta_t \mathbf{U}, \qquad \mathbf{U}_{ij} = \tfrac{1}{K}.$$

Each token is replaced by a uniformly random symbol with probability β_t. The stationary distribution is uniform noise.

Austin J. et al. Structured denoising diffusion models in discrete state-spaces, 2021.

Transition Matrix

- The choice of $\mathbf{Q}_t$ determines how information is erased and what the stationary distribution becomes.
- $\mathbf{Q}_t$ and $\mathbf{Q}_{1:t}$ should be easy to compute for each t.

Common choices

- **Uniform diffusion**

  $$\mathbf{Q}_t = (1 - \beta_t)\mathbf{I} + \beta_t \mathbf{U}, \qquad \mathbf{U}_{ij} = \frac{1}{K}.$$

  Each token is replaced by a uniformly random symbol with probability β_t. The stationary distribution is uniform noise.
- **Absorbing diffusion**

  $$\mathbf{Q}_t = (1 - \beta_t)\mathbf{I} + \beta_t \, \mathbf{e}_m \mathbf{1}^\top.$$

  Tokens are gradually replaced by a special mask m; the stationary distribution is fully masked.

Austin J. et al. Structured denoising diffusion models in discrete state-spaces, 2021.

Transition Matrix

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathrm{Cat}(\mathbf{Q}_{1:t}\mathbf{x}_0), \qquad \mathbf{Q}_{1:t} = \mathbf{Q}_t\mathbf{Q}_{t-1}\cdots\mathbf{Q}_1.$$

Transition Matrix

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathrm{Cat}(\mathbf{Q}_{1:t}\mathbf{x}_0), \qquad \mathbf{Q}_{1:t} = \mathbf{Q}_t\mathbf{Q}_{t-1}\cdots\mathbf{Q}_1.$$

Uniform Diffusion

$$\mathbf{Q}_t = (1-\beta_t)\mathbf{I} + \beta_t\mathbf{U}, \qquad \mathbf{U}_{ij} = \tfrac{1}{K}.$$

$$\mathbf{Q}_{1:t} = \bar{\alpha}_t\mathbf{I} + (1-\bar{\alpha}_t)\mathbf{U}, \quad \bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s).$$

Transition Matrix

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathrm{Cat}(\mathbf{Q}_{1:t}\mathbf{x}_0), \qquad \mathbf{Q}_{1:t} = \mathbf{Q}_t\mathbf{Q}_{t-1}\cdots\mathbf{Q}_1.$$

Uniform Diffusion

$$\mathbf{Q}_t = (1-\beta_t)\mathbf{I} + \beta_t\mathbf{U}, \qquad \mathbf{U}_{ij} = \tfrac{1}{K}.$$

$$\mathbf{Q}_{1:t} = \bar{\alpha}_t\mathbf{I} + (1-\bar{\alpha}_t)\mathbf{U}, \quad \bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s).$$

- Each token retains its original value with prob. $\bar{\alpha}_t$.
- It becomes uniformly random with prob. $(1-\bar{\alpha}_t)$.
- As $t \to T$, the process converges to the stationary uniform distribution.

Transition Matrix

Absorbing Diffusion

$$\mathbf{Q}_t = (1 - \beta_t)\mathbf{I} + \beta_t \, \mathbf{e}_m \mathbf{1}^\top,$$

$$\mathbf{Q}_{1:t} = \bar{\alpha}_t \, \mathbf{I} + (1 - \bar{\alpha}_t) \, \mathbf{e}_m \mathbf{1}^\top, \qquad \bar{\alpha}_t = \prod_{s=1}^{t} (1 - \beta_s).$$

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Transition Matrix

Absorbing Diffusion

$$\mathbf{Q}_t = (1-\beta_t)\mathbf{I} + \beta_t\,\mathbf{e}_m\mathbf{1}^\top,$$

$$\mathbf{Q}_{1:t} = \bar{\alpha}_t\,\mathbf{I} + (1-\bar{\alpha}_t)\,\mathbf{e}_m\mathbf{1}^\top, \qquad \bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s).$$

- Each token retains its original value with prob. $\bar{\alpha}_t$.
- It becomes $\mathbf{e}_m$ with prob. $(1-\bar{\alpha}_t)$.
- As $t \to T$, all tokens converge to the mask state: $q(\mathbf{x}_T) \approx \mathrm{Cat}(\mathbf{e}_m)$.
- This makes the process analogous to **masked language modeling**.

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Uniform vs. Absorbing Transition Matrix

Aspect	**Uniform Diffusion**	**Absorbing Diffusion**
$\mathbf{Q}_t$	$(1-\beta_t)\mathbf{I}+\beta_t\mathbf{U}$	$(1-\beta_t)\mathbf{I}+\beta_t\mathbf{e}_m\mathbf{1}^\top$
$\mathbf{Q}_{1:t}$	$\bar{\alpha}_t\mathbf{I}+(1-\bar{\alpha}_t)\mathbf{U}$	$\bar{\alpha}_t\mathbf{I}+(1-\bar{\alpha}_t)\mathbf{e}_m\mathbf{1}^\top$
$\mathbf{Q}_{1:\infty}$	$\mathbf{U}$	$\mathrm{Cat}(\mathbf{e}_m)$
Interpretation	Random replacement	Gradual masking of tokens
Application	Image / symbol diffusion	Text diffusion $\approx$ Masked LM

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Uniform vs. Absorbing Transition Matrix

Aspect	Uniform Diffusion	Absorbing Diffusion
$\mathbf{Q}_t$	$(1-\beta_t)\mathbf{I}+\beta_t\mathbf{U}$	$(1-\beta_t)\mathbf{I}+\beta_t\mathbf{e}_m\mathbf{1}^\top$
$\mathbf{Q}_{1:t}$	$\bar{\alpha}_t\mathbf{I}+(1-\bar{\alpha}_t)\mathbf{U}$	$\bar{\alpha}_t\mathbf{I}+(1-\bar{\alpha}_t)\mathbf{e}_m\mathbf{1}^\top$
$\mathbf{Q}_{1:\infty}$	$\mathbf{U}$	$\mathrm{Cat}(\mathbf{e}_m)$
Interpretation	Random replacement	Gradual masking of tokens
Application	Image / symbol diffusion	Text diffusion $\approx$ Masked LM

Observation

Both schemes gradually destroy information, but differ in their stationary limit. Absorbing diffusion bridges diffusion and masked-language-model objectives.

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Outline

1. Link between Flow Matching and Score-Based Models

2. Discrete Diffusion Models

   Forward Discrete Process

   Reverse Discrete Diffusion

Posterior of the Forward Process

ELBO

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

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Posterior of the Forward Process

ELBO

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

- Conditioned reverse distribution $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ played crucial role in the continuous-state diffusion model.
- It shows the probability of a previous state given the noisy state $\mathbf{x}_t$ and the original clean data $\mathbf{x}_0$.

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Posterior of the Forward Process

ELBO

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

- Conditioned reverse distribution $q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)$ played crucial role in the continuous-state diffusion model.
- It shows the probability of a previous state given the noisy state $\mathbf{x}_t$ and the original clean data $\mathbf{x}_0$.

Discrete conditioned reverse distribution

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{x}_0)\, q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} \\ = \frac{\mathrm{Cat}(\mathbf{Q}_t)\cdot\mathrm{Cat}(\mathbf{Q}_{1:t-1})}{\mathrm{Cat}(\mathbf{Q}_{1:t})}.$$

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Posterior of the Forward Process

Discrete conditioned reverse distribution

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathrm{Cat}\left(\frac{\mathbf{Q}_t \mathbf{x}_t \odot \mathbf{Q}_{1:t-1}\mathbf{x}_0}{\mathbf{x}_t^\top \mathbf{Q}_{1:t}\mathbf{x}_0}\right).$$

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Posterior of the Forward Process

Discrete conditioned reverse distribution

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathrm{Cat}\left(\frac{\mathbf{Q}_t\mathbf{x}_t \odot \mathbf{Q}_{1:t-1}\mathbf{x}_0}{\mathbf{x}_t^\top \mathbf{Q}_{1:t}\mathbf{x}_0}\right).$$

Recall the ELBO term

$$\mathcal{L}_t = \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}\mathrm{KL}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \,\|\, p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)\big),$$

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Posterior of the Forward Process

Discrete conditioned reverse distribution

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathrm{Cat}\left(\frac{\mathbf{Q}_t \mathbf{x}_t \odot \mathbf{Q}_{1:t-1}\mathbf{x}_0}{\mathbf{x}_t^\top \mathbf{Q}_{1:t}\mathbf{x}_0}\right).$$

Recall the ELBO term

$$\mathcal{L}_t = \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \mathrm{KL}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \,\|\, p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)\big),$$

- Both $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ and $q(\mathbf{x}_t|\mathbf{x}_0)$ are known analytically from the forward process.
- The reverse process $p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is a learned categorical distribution:

  $$p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathrm{Cat}\big(\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\big),$$

  where $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ is a neural network.

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Discrete-time ELBO for Discrete Diffusion

ELBO term

$$\mathcal{L}_t = \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \mathrm{KL}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \,\|\, p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)\big).$$

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Discrete-time ELBO for Discrete Diffusion

ELBO term

$$\mathcal{L}_t = \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \mathrm{KL}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \,\|\, p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)\big).$$

Categorical KL

$$\mathrm{KL}\big(\mathrm{Cat}(\mathbf{q}) \,\|\, \mathrm{Cat}(\mathbf{p})\big) = \sum_{k=1}^{K} q_k \log \frac{q_k}{p_k} = \mathrm{H}(\mathbf{q}, \mathbf{p}) - \mathrm{H}(\mathbf{q}),$$

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Discrete-time ELBO for Discrete Diffusion

ELBO term

$$\mathcal{L}_t = \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \mathrm{KL}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \,\|\, p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)\big).$$

Categorical KL

$$\mathrm{KL}\big(\mathrm{Cat}(\mathbf{q}) \,\|\, \mathrm{Cat}(\mathbf{p})\big) = \sum_{k=1}^{K} q_k \log \frac{q_k}{p_k} = \mathrm{H}(\mathbf{q}, \mathbf{p}) - \mathrm{H}(\mathbf{q}),$$

- $\mathrm{H}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)\big)$ is a constant w.r.t. $\boldsymbol{\theta}$.
- $\mathrm{H}(\mathbf{q}, \mathbf{p}) = -\sum_k q_k \log p_k$ is a **cross-entropy loss**.

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Discrete-time ELBO for Discrete Diffusion

ELBO term

$$\mathcal{L}_t = \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \mathrm{KL}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \,\|\, p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)\big).$$

Categorical KL

$$\mathrm{KL}\big(\mathrm{Cat}(\mathbf{q}) \,\|\, \mathrm{Cat}(\mathbf{p})\big) = \sum_{k=1}^{K} q_k \log \frac{q_k}{p_k} = \mathrm{H}(\mathbf{q}, \mathbf{p}) - \mathrm{H}(\mathbf{q}),$$

- $\mathrm{H}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)\big)$ is a constant w.r.t. $\boldsymbol{\theta}$.
- $\mathrm{H}(\mathbf{q}, \mathbf{p}) = -\sum_k q_k \log p_k$ is a **cross-entropy loss**.

Therefore, minimizing $\mathcal{L}_t$ w.r.t. $\boldsymbol{\theta}$ is equivalent to minimizing

$$\mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \mathrm{H}\Big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0),\, p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)\Big).$$

Austin J. et al., Structured denoising diffusion models in discrete state-spaces, 2021.

Summary

- Diffusion and score-based models are special cases of the flow matching approach, but use curved trajectories.
- Diffusion approach has several key advantages over autoregressive approach.
- Forward discrete diffusion process defines Markov chain with discrete states.
- There are several ways to make it tractable (uniform / absorbing transitions).
- Reverse discrete diffusion process uses the variational approach to invert forward process.
- Discrete-time ELBO for discrete diffusion is a cross-entropy loss.