Deep Generative Models

Lecture 14

Roman Isachenko

AI Masters

Autumn, 2022

Recap of previous lecture

Continuous dynamic

$$\frac{d\mathbf{z}(t)}{dt} = f(\mathbf{z}(t), \boldsymbol{\theta}).$$

Forward pass

$$\mathbf{z}(t_1) = \int_{t_0}^{t_1} f(\mathbf{z}(t), \boldsymbol{\theta}) dt + \mathbf{z}_0 \quad \Rightarrow \quad \text{ODE Solver}$$

Backward pass

$$\left.\begin{aligned}
\frac{\partial L}{\partial \boldsymbol{\theta}(t_0)} &= \mathbf{a}_{\boldsymbol{\theta}}(t_0) = -\int_{t_1}^{t_0} \mathbf{a}_{\mathbf{z}}(t)^T \frac{\partial f(\mathbf{z}(t), \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(t)} dt + 0 \\
\frac{\partial L}{\partial \mathbf{z}(t_0)} &= \mathbf{a}_{\mathbf{z}}(t_0) = -\int_{t_1}^{t_0} \mathbf{a}_{\mathbf{z}}(t)^T \frac{\partial f(\mathbf{z}(t), \boldsymbol{\theta})}{\partial \mathbf{z}(t)} dt + \frac{\partial L}{\partial \mathbf{z}(t_1)} \\
\mathbf{z}(t_0) &= -\int_{t_1}^{t_0} f(\mathbf{z}(t), \boldsymbol{\theta}) dt + \mathbf{z}_1.
\end{aligned}\right\} \Rightarrow \text{ODE Solver}$$

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

Recap of previous lecture

Continuous normalizing flows

$$\frac{d \log p(\mathbf{z}(t), t)}{dt} = -\text{tr} \left(\frac{\partial f(\mathbf{z}(t), \boldsymbol{\theta})}{\partial \mathbf{z}(t)} \right).$$

Forward transform + log-density

$$\begin{bmatrix} \mathbf{x} \\ \log p(\mathbf{x}|\boldsymbol{\theta}) \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \log p(\mathbf{z}) \end{bmatrix} + \int_{t_0}^{t_1} \begin{bmatrix} f(\mathbf{z}(t), \boldsymbol{\theta}) \\ -\text{tr} \left(\frac{\partial f(\mathbf{z}(t), \boldsymbol{\theta})}{\partial \mathbf{z}(t)} \right) \end{bmatrix} dt.$$

Hutchinson's trace estimator

$$\log p(\mathbf{z}(t_1)) = \log p(\mathbf{z}(t_0)) - \int_{t_0}^{t_1} \text{tr} \left(\frac{\partial f(\mathbf{z}(t), \boldsymbol{\theta})}{\partial \mathbf{z}(t)} \right) dt =$$

$$= \log p(\mathbf{z}(t_0)) - \mathbb{E}_{p(\boldsymbol{\epsilon})} \int_{t_0}^{t_1} \left[\boldsymbol{\epsilon}^T \frac{\partial f}{\partial \mathbf{z}} \boldsymbol{\epsilon} \right] dt.$$

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

Recap of previous lecture

SDE basics

Let define stochastic process $\mathbf{x}(t)$ with initial condition $\mathbf{x}(0) \sim p_0(\mathbf{x})$:

$$d\mathbf{x} = \mathbf{f}(\mathbf{x}, t)dt + g(t)d\mathbf{w},$$

where $\mathbf{w}(t)$ is the standard Wiener process (Brownian motion)

$$\mathbf{w}(t) - \mathbf{w}(s) \sim \mathcal{N}(0, (t-s)\mathbf{I}), \quad d\mathbf{w} = \boldsymbol{\epsilon} \cdot \sqrt{dt}, \text{ where } \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}).$$

Langevin dynamics

Let $\mathbf{x}_0$ be a random vector. Then under mild regularity conditions for small enough η samples from the following dynamics

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \eta \frac{1}{2} \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\boldsymbol{\theta}) + \sqrt{\eta} \cdot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}).$$

will comes from $p(\mathbf{x}|\boldsymbol{\theta})$.

The density $p(\mathbf{x}|\boldsymbol{\theta})$ is a **stationary** distribution for the Langevin SDE.

Welling M. Bayesian Learning via Stochastic Gradient Langevin Dynamics, 2011

Outline

1. Score matching
2. Noise conditioned score network
3. Diffusion models

Outline

1. Score matching
2. Noise conditioned score network
3. Diffusion models

Score matching

We could sample from the model if we have $\nabla_{\mathbf{x}} \log p(\mathbf{x}|\boldsymbol{\theta})$.

Fisher divergence

$$D_F(\pi, p) = \frac{1}{2}\mathbb{E}_{\pi}\bigl\|\nabla_{\mathbf{x}} \log p(\mathbf{x}|\boldsymbol{\theta}) - \nabla_{\mathbf{x}} \log \pi(\mathbf{x})\bigr\|_2^2 \rightarrow \min_{\boldsymbol{\theta}}$$

Score function

$$\mathbf{s}(\mathbf{x}, \boldsymbol{\theta}) = \nabla_{\mathbf{x}} \log p(\mathbf{x}|\boldsymbol{\theta})$$

Problem: we do not know $\nabla_{\mathbf{x}} \log \pi(\mathbf{x})$.

Theorem

Under some regularity conditions, it holds

$$\frac{1}{2}\mathbb{E}_{\pi}\bigl\|\mathbf{s}(\mathbf{x}, \boldsymbol{\theta}) - \nabla_{\mathbf{x}} \log \pi(\mathbf{x})\bigr\|_2^2 = \mathbb{E}_{\pi}\Bigl[\frac{1}{2}\|\mathbf{s}(\mathbf{x}, \boldsymbol{\theta})\|_2^2 + \mathrm{tr}\bigl(\nabla_{\mathbf{x}}\mathbf{s}(\mathbf{x}, \boldsymbol{\theta})\bigr)\Bigr] + \text{const}$$

Here $\nabla_{\mathbf{x}}\mathbf{s}(\mathbf{x}, \boldsymbol{\theta}) = \nabla^2_{\mathbf{x}} \log p(\mathbf{x}|\boldsymbol{\theta})$ is a Hessian matrix.

Hyvarinen A. Estimation of non-normalized statistical models by score matching, 2005

Score matching

Theorem

$$\frac{1}{2}\mathbb{E}_\pi\Big\|\mathbf{s}(\mathbf{x},\boldsymbol{\theta})-\nabla_\mathbf{x}\log\pi(\mathbf{x})\Big\|_2^2=\mathbb{E}_\pi\left[\frac{1}{2}\|\mathbf{s}(\mathbf{x},\boldsymbol{\theta})\|_2^2+\mathrm{tr}\Big(\nabla_\mathbf{x}\mathbf{s}(\mathbf{x},\boldsymbol{\theta})\Big)\right]+\text{const}$$

Proof (only for 1D)

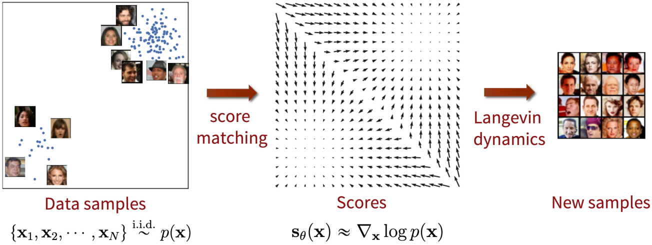
$$\mathbb{E}_\pi\Big\|s(x)-\nabla_x\log\pi(x)\Big\|_2^2=\mathbb{E}_\pi\left[s(x)^2+(\nabla_x\log\pi(x))^2-2[s(x)\nabla_x\log\pi(x)]\right]$$

$$\begin{aligned}
\mathbb{E}_\pi[s(x)\nabla_x\log\pi(x)] &= \int \pi(x)\nabla_x\log p(x)\nabla_x\log\pi(x)dx \\
&= \int \nabla_x\log p(x)\nabla_x\pi(x)dx = \pi(x)\nabla_x\log p(x)\Big|_{-\infty}^{+\infty} \\
&- \int \nabla_x^2\log p(x)\pi(x)dx = -\mathbb{E}_\pi\nabla_x^2\log p(x)
\end{aligned}$$

$$\frac{1}{2}\mathbb{E}_\pi\Big\|s(x)-\nabla_x\log\pi(x)\Big\|_2^2=\frac{1}{2}\mathbb{E}_\pi\Big[s(x)^2+\nabla_x s(x)\Big]+\text{const}.$$

Hyvarinen A. Estimation of non-normalized statistical models by score matching, 2005

Score matching



Theorem (implicit score matching)

$$\frac{1}{2}\mathbb{E}_\pi \left\|\mathbf{s}(\mathbf{x},\boldsymbol{\theta}) - \nabla_\mathbf{x} \log \pi(\mathbf{x})\right\|_2^2 = \mathbb{E}_\pi \left[\frac{1}{2}\|\mathbf{s}(\mathbf{x},\boldsymbol{\theta})\|_2^2 + \mathrm{tr}\left(\nabla_\mathbf{x}\mathbf{s}(\mathbf{x},\boldsymbol{\theta})\right)\right] + \mathrm{const}$$

1. The left hand side is intractable due to unknown $\pi(\mathbf{x})$ – **denoising score matching**.
2. The right hand side is complex due to Hessian matrix – **sliced score matching**.

Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021

Score matching

Sliced score matching (Hutchinson's trace estimation)

$$\mathrm{tr}\Big(\nabla_{\mathbf{x}} \mathbf{s}(\mathbf{x}, \boldsymbol{\theta})\Big) = \mathbb{E}_{p(\boldsymbol{\epsilon})} \left[\boldsymbol{\epsilon}^T \nabla_{\mathbf{x}} \mathbf{s}(\mathbf{x}, \boldsymbol{\theta}) \boldsymbol{\epsilon} \right],$$

where $\mathbb{E}[\boldsymbol{\epsilon}] = 0$ and $\mathrm{Cov}(\boldsymbol{\epsilon}) = \mathbf{I}$.

Denoising score mathing

Let perturb original data by normal noise $p(\mathbf{x}|\mathbf{x}', \sigma) = \mathcal{N}(\mathbf{x}|\mathbf{x}', \sigma^2 \mathbf{I})$

$$\pi(\mathbf{x}|\sigma) = \int \pi(\mathbf{x}') p(\mathbf{x}|\mathbf{x}', \sigma) d\mathbf{x}'.$$

Then the solution of

$$\frac{1}{2} \mathbb{E}_{\pi(\mathbf{x}|\sigma)} \Big\| \mathbf{s}(\mathbf{x}, \boldsymbol{\theta}, \sigma) - \nabla_{\mathbf{x}} \log \pi(\mathbf{x}|\sigma) \Big\|_2^2 \to \min_{\boldsymbol{\theta}}$$

satisfies $\mathbf{s}(\mathbf{x}, \boldsymbol{\theta}, \sigma) \approx \mathbf{s}(\mathbf{x}, \boldsymbol{\theta}, 0) = \mathbf{s}(\mathbf{x}, \boldsymbol{\theta})$ using small enough noise scale σ.

Song Y. Sliced Score Matching: A Scalable Approach to Density and Score Estimation, 2019

Vincent P. A connection between score matching and denoising autoencoders. Neural computation, 2011

Denoising score matching

Theorem

$$\mathbb{E}_{\pi(\mathbf{x}|\sigma)}\Big\|\mathbf{s}(\mathbf{x}, \boldsymbol{\theta}, \sigma) - \nabla_{\mathbf{x}} \log \pi(\mathbf{x}|\sigma)\Big\|_2^2 =$$
$$= \mathbb{E}_{\pi(\mathbf{x}')}\mathbb{E}_{p(\mathbf{x}|\mathbf{x}',\sigma)}\Big\|\mathbf{s}(\mathbf{x}, \boldsymbol{\theta}, \sigma) - \nabla_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{x}', \sigma)\Big\|_2^2$$

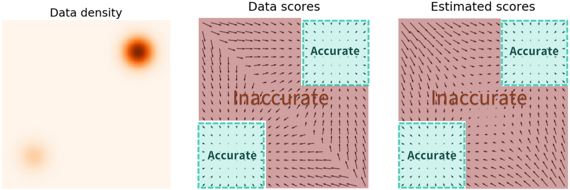
Here $\nabla_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{x}', \sigma) = -\frac{\mathbf{x}-\mathbf{x}'}{\sigma^2}$.

- The RHS does not need to compute $\nabla_{\mathbf{x}} \log \pi(\mathbf{x}|\sigma)$ and even more $\nabla_{\mathbf{x}} \log \pi(\mathbf{x})$.
- $\mathbf{s}(\mathbf{x}, \boldsymbol{\theta}, \sigma)$ tries to **denoise** a corrupted sample.
- Score function $\mathbf{s}(\mathbf{x}, \boldsymbol{\theta}, \sigma)$ parametrized by σ. How to make it?

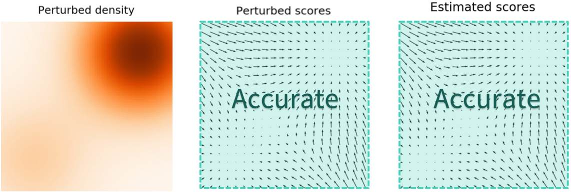
Vincent P. A Connection Between Score Matching and Denoising Autoencoders, 2010

Denoising score matching

- If σ is **small**, the score function is not accurate and Langevin dynamics will probably fail to jump between modes.



- If σ is **large**, it is good for low-density regions and multimodal distributions, but we will learn too corrupted distribution.



Song Y. Generative Modeling by Estimating Gradients of the Data Distribution, blog post, 2021

Outline

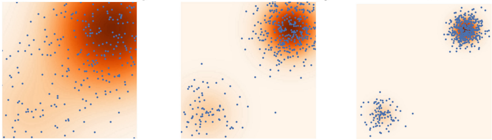
Noise conditioned score network

- Define the sequence of noise levels: $\sigma_1 > \sigma_2 > \dots > \sigma_L$.
- Perturb the original data with the different noise level to get $\pi(\mathbf{x}|\sigma_1), \dots, \pi(\mathbf{x}|\sigma_L)$.
- Train denoised score function $\mathbf{s}(\mathbf{x}, \boldsymbol{\theta}, \sigma)$ for each noise level:

$$\sum_{l=1}^{L} \sigma_l^2 \mathbb{E}_{\pi(\mathbf{x}')} \mathbb{E}_{p(\mathbf{x}|\mathbf{x}', \sigma_l)} \big\| \mathbf{s}(\mathbf{x}, \boldsymbol{\theta}, \sigma_l) - \nabla_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{x}', \sigma_l) \big\|_2^2 \to \min_{\boldsymbol{\theta}}$$

- Sample from **annealed** Langevin dynamics (for $l = 1, \dots, L$).

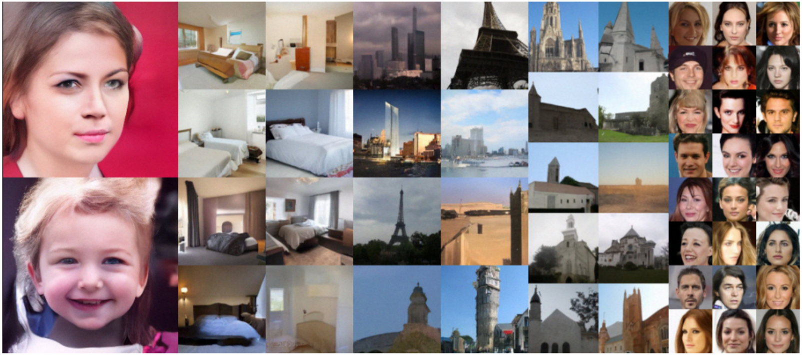
$\sigma_1 > \sigma_2 > \sigma_3$



Song Y. et al. Generative Modeling by Estimating Gradients of the Data Distribution, 2019

Noise conditioned score network

Samples



Song Y. et al. *Improved Techniques for Training Score-Based Generative Models, 2020*

Outline

1. Score matching
2. Noise conditioned score network
3. Diffusion models

Forward diffusion process

Let $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x})$, $\beta \in (0, 1)$. Define the Markov chain

$$\mathbf{x}_t = \sqrt{1-\beta}\mathbf{x}_{t-1} + \sqrt{\beta}\boldsymbol{\epsilon}, \quad \text{where } \boldsymbol{\epsilon} \sim \mathcal{N}(0, 1);$$

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

Statement

Applying the Markov chain to samples from any $\pi(\mathbf{x})$ we will get $\mathbf{x}_\infty \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, 1)$. Here $p_\infty(\mathbf{x})$ is a **stationary** distribution:

$$p_\infty(\mathbf{x}) = \int q(\mathbf{x}|\mathbf{x}')p_\infty(\mathbf{x}')d\mathbf{x}'.$$

Statement

Denote $\alpha_t = 1 - \beta_t$, $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_t$. Then

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \quad \text{where } \boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$$

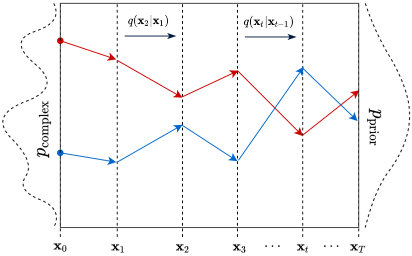
$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t|\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}).$$

We could sample from any timestamp using only $\mathbf{x}_0$.

Sohl-Dickstein J. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, 2015

Forward diffusion process

Diffusion refers to the flow of particles from high-density regions towards low-density regions.

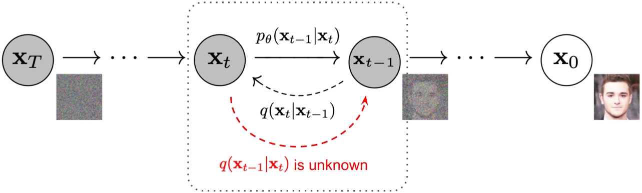


1. $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x})$;
2. $\mathbf{x}_t = \sqrt{1-\beta}\mathbf{x}_{t-1} + \sqrt{\beta}\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0,1),\ t \geq 1$;
3. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0,1)$.

Now our goal is to revert this process.

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

Reverse diffusion process



Let define the reverse process

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

Forward process

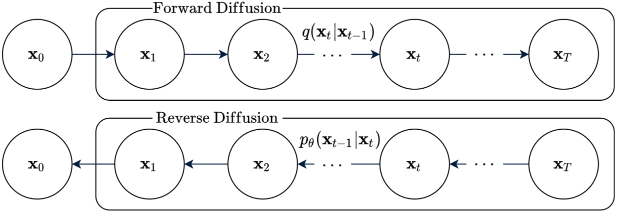
1. $\mathbf{x}_0 = \mathbf{x} \sim \pi(\mathbf{x})$;
2. $\mathbf{x}_t = \sqrt{1-\beta}\mathbf{x}_{t-1} + \sqrt{\beta}\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$, $t \geq 1$;
3. $\mathbf{x}_T \sim p_\infty(\mathbf{x}) = \mathcal{N}(0, 1)$.

Reverse process

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

Weng L. What are Diffusion Models?, blog post, 2021

Diffusion model



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

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

- Probabilistic model

$$p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) = p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})p(\mathbf{z}|\boldsymbol{\theta})$$

- Generative distribution and prior

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

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

Diffusion model

ELBO

$$\log p(\mathbf{x}|\boldsymbol{\theta}) \geq \int q(\mathbf{z}|\mathbf{x}) \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z}|\mathbf{x})} d\mathbf{z} = \mathcal{L}(q, \boldsymbol{\theta}) \to \max_{q,\theta}$$

Statement

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{x}_1, \dots, \mathbf{x}_T|\mathbf{x}_0)} \frac{p(\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_T|\boldsymbol{\theta})}{q(\mathbf{x}_1, \dots, \mathbf{x}_T|\mathbf{x}_0)} =$$

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

$$- \log p(\mathbf{x}_0|\mathbf{x}_1, \boldsymbol{\theta}) \Big]$$

- First term is constant (KL between two standard normals).
- Third term is a decoder distribution (could be AR model or discretized distribution (like mixture of logistics)).

Ho J. Denoising Diffusion Probabilistic Models, 2020

Diffusion model

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

Here

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

$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0)$ and $\tilde{\beta}_t$ have analytical formulas (we omit it) and both dependent on β_t.

- Assume $\boldsymbol{\sigma}^2(\mathbf{x}_t, \boldsymbol{\theta}, t) = \tilde{\beta}_t \mathbf{I}$ (reminder: $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}_{t-1}|\boldsymbol{\mu}(\mathbf{x}_t, \boldsymbol{\theta}, t), \boldsymbol{\sigma}^2(\mathbf{x}_t, \boldsymbol{\theta}, t))$).
- Use KL formula for normal distributions.
- Use the fact $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}$

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

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

Ho J. Denoising Diffusion Probabilistic Models, 2020

Diffusion model

Reparametrization

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

KL term

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

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

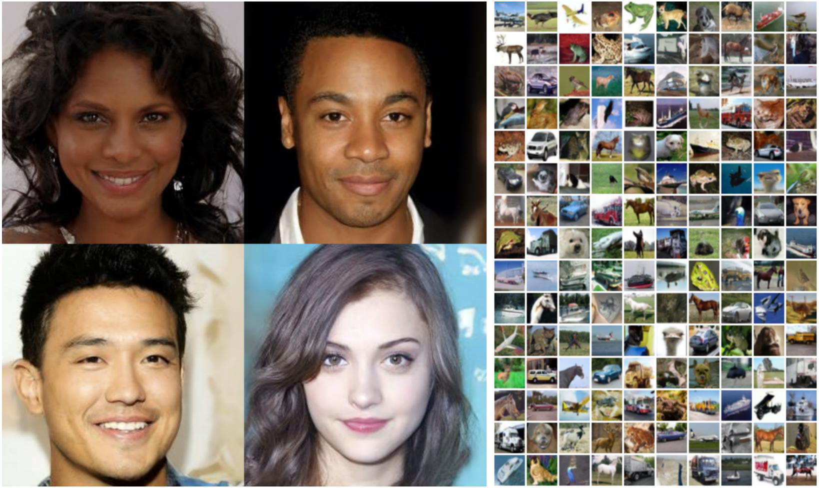
Noise conditioned score network

$$\mathbb{E}_{p(\mathbf{x}|\mathbf{x}', \sigma_l)} \left\| \mathbf{s}(\mathbf{x}, \boldsymbol{\theta}, \sigma_l) - \nabla_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{x}', \sigma_l) \right\|_2^2 \to \min_{\boldsymbol{\theta}}$$

Ho J. Denoising Diffusion Probabilistic Models, 2020

Denoising diffusion probabilistic model

Samples



Ho J. *Denoising Diffusion Probabilistic Models*, 2020

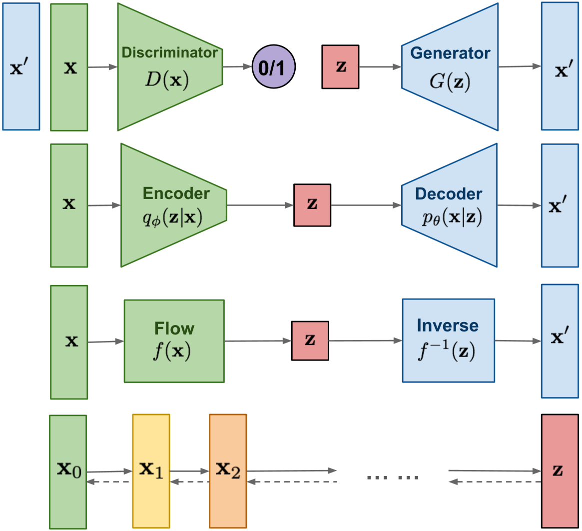
The poorest course overview :)

GAN: Adversarial training

VAE: maximize variational lower bound

Flow-based models: Invertible transform of distributions

Diffusion models: Gradually add Gaussian noise and then reverse



Weng L. What are Diffusion Models?, blog post, 2021

Summary

- Score matching proposes to minimize Fisher divergence to get score function.
- Sliced score matching and denoising score matching are two techniques to get scalable algorithm for fitting Fisher divergence.
- Noise conditioned score nework uses multiple noise levels and annealed Langevin dynamics to fit score function.
- Gaussian diffusion process is a Markov chain that inject Gaussian noise.
- Diffusion model is a VAE model which revert gaussian diffusion process using variational inference.
- Objective of diffusion model is closely related to the noise conditioned score network.