Score-based Generative Modeling in Latent Space

Pramook Khungurn

June 14, 2023

This note is written as I read the paper "Score-based Generative Modeling in Latent Space" by Vahdat et al. [VKK21].

1 Introduction

- This paper presents an interesting approach to generative modeling. It is a combination of a variational autoencoder (VAE) and a denoising diffusion generative model (DDPM) or a score-based model (SGM).
- The paper starts with a VAE that is already quite good, the NVAE [VK21], and try to make it better by combining it with a diffusion model.
- For a VAE, the latent code $\mathbf{z}$ is ideally supposed to be distributed according to a prior distribution $p(\mathbf{z})$, which we typically take to be the Gaussian distribution $\mathcal{N}(\mathbf{0}, I)$.
- Nevertheless, the distribution of latent codes that the VAE decoder can decode well is the distribution $\{\mathbf{z} : \mathbf{x} \sim p_{\mathrm{data}}, \mathbf{z} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})\}$ where $q_{\boldsymbol{\phi}}$ is the VAE encoder. This distribution might not agree with $\mathcal{N}(\mathbf{0}, I)$ even though we have optimized the VAE as best as we can.
- The idea of the paper is to transform $\mathcal{N}(\mathbf{0}, I)$ into $\{\mathbf{z} : \mathbf{x} \sim p_{\mathrm{data}}, \mathbf{z} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})\}$ with a diffusion model.
- The approach is called "latent score-based generative model" (LSGM) because, to generate a sample, we use a diffusion model (aka a score-based model) to sample a VAE latent code. Then, we use the VAE decoder to generate a real data item.
- In this write-up, I will not follow the paper that closely. This is because I don't like the way it formulates diffusion model, which uses the SDE in Song et al.'s 2021 paper [SSDK+21]. I'd rather use the VDM formulation [KSPH21], which I'm more familiar with.

2 Background

- VAE and DDPM are similar.
  - To generate a data sample, they both sample a **latent code** from a prior distribution.
  - Then, they transform the latent code to a real data sample through some kind of process.
- We shall call the process that goes from a latent code to a real data sample the "backward process." The other way around is the "forward process".
- The backward process is denoted by the probability function p,
  and the forward process the probabiliy function q.
- We denote a real data sample with $\mathbf{x}$. We assume that it is a member of $\mathbb{R}^D$, which we call the **ambient space**. A data item is distributed according to the data distribution p_{data}.
- We denote a latent code with $\mathbf{z}$. It is a member of the **latent space** $\mathbb{R}^d$.

2.1 Variational Autoencoder

- For a VAE, we have that $d < D$ in general.
- The backward process.
  - Sample $\mathbf{z}$ according to the prior distribution $p(\mathbf{z})$, which we generally take to be $\mathcal{N}(\mathbf{0}, I)$.
  - Use the VAE decoder to sample $\mathbf{x}$ given $\mathbf{z}$.
  - So,

$$
\begin{aligned}
p(\mathbf{x}, \mathbf{z}) &= p(\mathbf{x}|\mathbf{z})p(\mathbf{z}) \\
p(\mathbf{x}) &= \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})\,\mathrm{d}\mathbf{z}.
\end{aligned} \tag{1}
$$

- The forward process.
  - Sample $\mathbf{x}$ according to p_{data}.
    * For convenience, we shall denote p_{data} with q, so $p_{\text{data}}(\mathbf{x}) = q(\mathbf{x})$.
  - Use the encoder to sample $\mathbf{z}$ given $\mathbf{x}$.
  - Symbolically,

$$q(\mathbf{x}, \mathbf{z}) = q(\mathbf{z}|\mathbf{x})q(\mathbf{x}).$$

- We model the encoder with a neural network with parameters $\boldsymbol{\phi}$. Typically, this network has two functions $\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x})$ and $\boldsymbol{\sigma}_{\boldsymbol{\phi}}(\mathbf{x})$, and the distribution it models is given by

$$q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}), \operatorname{diag}(\boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\mathbf{x}))).$$

- We model the decoder with a neural network with parameters $\boldsymbol{\psi}$. The network only has one function $\boldsymbol{\mu}_{\boldsymbol{\psi}}(\mathbf{z})$. The distribution it models is given by

$$p_{\boldsymbol{\psi}}(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{\boldsymbol{\psi}}(\mathbf{z}), \sigma_E^2 I)$$

  where σ_E is a hyperparameter.

- To optimize the network, we seek to minimize the negative log-likelihood

$$
\begin{aligned}
&E_{\mathbf{x}\sim q(\mathbf{x})}[-\log p(\mathbf{x})] \\
&= -E_{\mathbf{x}\sim q(\mathbf{x})}\left[\log \int p_{\boldsymbol{\psi}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})\,\mathrm{d}\mathbf{z}\right] && \text{(Equation 1)} \\
&= -E_{\mathbf{x}\sim q(\mathbf{x})}\left[\log \int q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})\frac{p_{\boldsymbol{\psi}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\,\mathrm{d}\mathbf{z}\right] && \text{(importance sampling)} \\
&= -E_{\mathbf{x}\sim q(\mathbf{x})}\left[\log E_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\frac{p_{\boldsymbol{\psi}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\right]\right] && \text{(integral to expectation)} \\
&\geq -E_{\mathbf{x}\sim q(\mathbf{x})}\left[E_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_{\boldsymbol{\psi}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\right]\right] && \text{(Jensen's inquality)} \\
&= -E_{\mathbf{x}\sim q(\mathbf{x}), \mathbf{z}\sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\big[\log p_{\boldsymbol{\psi}}(\mathbf{x}|\mathbf{z}) - \log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) + \log p(\mathbf{z})\big].
\end{aligned}
$$

2.2 Variational Diffusion Model

- We follow the treatment of Kingma et al. [KSPH21], which is summarized in one of my notes [Khu22].
- For a diffusion model, $D = d$.
- The forward process starts with a data item $\mathbf{x}$ sampled from $p_{\mathrm{data}}(\mathbf{x}) = q(\mathbf{x})$.
- We then scale $\mathbf{x}$ down and add noise to it. This process is a continous process that runs from time $t = 0$ to $t = 1$. At time t, the noised data item is denoted by $\mathbf{z}_t$. The distribution of $\mathbf{z}_t$ is given by:

  $$q(\mathbf{z}_t|\mathbf{x}) = \mathcal{N}(\mathbf{z}_t; \alpha_t \mathbf{x}, \sigma_t^2 I).$$

  The functions α_t and σ_t are together called the **noise schedule** of the diffusion process. The process is governed by the following stochastic differential equation:

  $$\mathrm{d}\mathbf{z}_t = \frac{\alpha_t'}{\alpha_t}\mathbf{z}_t\,\mathrm{d}t + \sqrt{2\sigma_t\bigg(\sigma_t' - \frac{\alpha_t'}{\alpha_t}\bigg)}\,\mathrm{d}\mathbf{W}.$$

- We require that $\alpha_t^2 + \sigma_t^2 = 1$ for all $t \in [0, 1]$. So, we are using the variance preserving formulation.
- The **signal-to-noise ratio** at time t, denoted by $\mathrm{SNR}(t)$, is defined as:

  $$\mathrm{SNR}(t) = \frac{\alpha_t^2}{\sigma_t^2}.$$

3 Method

- The paper refers to the method it proposes as the **latent score-based generative model** (LSGM).
- The method contains the following components.
  - An encoder $q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})$.
  - A diffusion model $p_{\boldsymbol{\theta}}(\mathbf{z}_0)$, which acts as the prior distribution.
  - A decoder $p_{\boldsymbol{\psi}}(\mathbf{x}|\mathbf{z}_0)$
- To train the whole system, we would like to minimize the log-likelihood:

  $$\begin{aligned} &E_{\mathbf{x}\sim p_{\mathrm{data}}}[-\log p(\mathbf{x})] \\ &\leq E_{\mathbf{x}\sim p_{\mathrm{data}}}\Big[E_{\mathbf{z}_0\sim q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})}[-\log p_{\boldsymbol{\psi}}(\mathbf{x}|\mathbf{z}_0)] + \mathrm{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})\|p_{\boldsymbol{\theta}}(\mathbf{z}_0))\Big] \\ &= E_{\mathbf{x}\sim p_{\mathrm{data}},\mathbf{z}_0\sim q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})}\Big[-\log p_{\boldsymbol{\psi}}(\mathbf{x}|\mathbf{z}_0) + \log q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x}) - \log p_{\boldsymbol{\theta}}(\mathbf{z}_0)\Big]. \end{aligned}$$

  The second line comes from the standard evidence lower-bound derivation found in VAE papers [Khu20]:

  $$-\log p(\mathbf{x}) \leq E_{\mathbf{z}_0\sim q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})}[-\log p_{\boldsymbol{\psi}}(\mathbf{x}|\mathbf{z}_0)] + \mathrm{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})\|p_{\boldsymbol{\theta}}(\mathbf{z}_0)).$$

  The third line just expanding the KL divergence into its constituent parts:

  $$\mathrm{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})\|p_{\boldsymbol{\theta}}(\mathbf{z}_0)) = E_{\mathbf{z}_0\sim q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})}\bigg[\log\frac{q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z}_0)}\bigg] = E_{\mathbf{z}_0\sim q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})}\Big[\log q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x}) - \log p_{\boldsymbol{\theta}}(\mathbf{z}_0)\Big].$$

- Now, the paper went on at a great length to find an expression for the cross entropy term

$$E_{\mathbf{z}_0 \sim q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})}\Big[-\log p_{\boldsymbol{\theta}}(\mathbf{z}_0) \Big].$$

  However, we will refer to the derivation in the "Variational Diffusion Models" paper by Kingma et al. [KSPH21, Khu22] as a shortcut.

- To apply the shortcut, we need to reinterpret the notations.
  - We will use q to denote the "forward" process. The forward process now has three steps.
    - Sample $\mathbf{x} \sim p_{\mathrm{data}}$.
    - Sample the latent code $\mathbf{z}_0$ corresponding to $\mathbf{x}$: $\mathbf{z}_0 \sim q(\mathbf{z}_0|\mathbf{x})$.
      - This process is modeled with the encoder $q_{\boldsymbol{\phi}}$.
    - Run the diffusion process forward in time to obtain $\mathbf{z}_1$ from $\mathbf{z}_0$: $\mathbf{z}_1 \sim q(\mathbf{z}_1|\mathbf{z}_0)$.
      - The paramaters of the forward process is the noise schedule α_t and σ_t.
      - We assume that $\alpha_t^2 + \sigma_t^2 = 1$; i.e., the variance preserving formuation.
      - We do this so that we can use the derivation in the Kingma et al. paper.
      - Note, however, that the Vahdat et al. paper is not limited to the variance preserving formulation.
  - The backward process is denoted by p. It also has three steps.
    - Sample $\mathbf{z}_1$ according to $p(\mathbf{z}_1)$, which is an isotropic Gaussian distribution.
    - Run the diffusion process backward to obtain $\mathbf{z}_0$ from $\mathbf{z}_1$: $\mathbf{z}_0 \sim p(\mathbf{z}_0|\mathbf{z}_1)$.
      - This part is modeled by the diffusion model $p_{\boldsymbol{\theta}}$.
    - Sample a data item in the ambient space $\mathbf{x}$ from the latent code $\mathbf{z}_0$: $\mathbf{x} \sim p(\mathbf{x}|\mathbf{z}_0)$.
      - This is modeled by the decoder $p_{\boldsymbol{\psi}}$.

- In practice, however, we will not run the diffusion process from exactly $t = 0$ to $t = 1$. Instead, we will run it from $t_{\min} \gtrapprox 0$ to $t_{\max} \lessapprox 1$. The reason for this is so that, for any time in the interval $[t_{\min}, t_{\max}]$, the **signal-to-noise ratio** (SNR) $\frac{\alpha_t^2}{\sigma_t^2}$ is finite.

- So, in practice, the backward process is as follows.
  - Sample $\mathbf{z}_{t_{\max}}$ according to $p(\mathbf{z}_{t_{\max}})$, which we shall approximate with $\mathcal{N}(\mathbf{0}, \sigma^2_{t_{\max}} I)$.
  - Run the diffusion process backward to obtain $\mathbf{z}_{t_{\min}}$ from $\mathbf{z}_{t_{\max}}$.
  - Use $\mathbf{z}_{t_{\min}}$ as an approximation for $\mathbf{z}_0$, and then sample the data item in the ambient space $\mathbf{x} \sim p(\mathbf{x}|\mathbf{z}_{t_{\min}})$.

- Using the variational lower bound in [Khu22], we have that

$$-\log p(\mathbf{z}_0) \le \mathrm{KL}(q(\mathbf{z}_1|\mathbf{z}_0) \| p(\mathbf{z}_{t_{\max}})) + E_{\mathbf{z}_{t_{\min}} \sim q(\mathbf{z}_{t_{\min}}|\mathbf{z}_0)}\big[-\log p(\mathbf{z}_0|\mathbf{z}_{p_{t_{\min}}}) \big] + \mathcal{L}_\infty(\mathbf{z}_0).$$

  If we choose $t_{\min}$ to be low enough and $t_{\max}$ high enough, the first two terms chould be negligible. So,

$$\begin{aligned} -\log p(\mathbf{z}_0) &\lessapprox \mathcal{L}_\infty(\mathbf{z}_0) \\ &= \frac{1}{2} E_{\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, I)}\bigg[\int_{\mathrm{SNR}_{\min}}^{\mathrm{SNR}_{\max}} \|\mathbf{z}_0 - \hat{\mathbf{z}}_{\boldsymbol{\theta}}(\underbrace{\alpha_{t(v)}\mathbf{z}_0 + \sigma_{t(v)}\boldsymbol{\xi}}_{\mathbf{z}_{t(v)}}, t(v))\big\|^2 \,\mathrm{d}v \bigg] \\ &= \frac{1}{2} E_{\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, I), v \sim \mathcal{U}([\mathrm{SNR}_{\min}, \mathrm{SNR}_{\max}])}\Big[\big\|\mathbf{z}_0 - \hat{\mathbf{z}}_{\boldsymbol{\theta}}(\alpha_{t(v)}\mathbf{z}_0 + \sigma_{t(v)}\boldsymbol{\xi}, t(v))\big\|^2 \Big] \end{aligned}$$

  where $\hat{\mathbf{z}}_{\boldsymbol{\theta}}$ is a neural network that predicts the denoised latent code $\mathbf{z}_0$ from the noised latent code $\mathbf{z}_t$, and $t(v)$ is the function that computes the time t from the SNR v.

- So, the overall loss that we want to minimize is:

$$E_{\mathbf{x} \sim p_{\mathrm{data}}, \mathbf{z}_0 \sim q_{\boldsymbol{\phi}(\mathbf{z}_0|\mathbf{x})}}$$

References

[Khu20] Pramook Khungurn. Variational autoencoder. `https://pkhungurn.github.io/notes/notes/ml/vae/index.html`, 2020. Accessed: 2023-06-13.

[Khu22] Pramook Khungurn. Variational diffusion models. `https://pkhungurn.github.io/notes/notes/ml/variational-diffusion-models/variational-diffusion-models.pdf`, 2022. Accessed: 2023-06-13.

[KSPH21] Diederik P. Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models, 2021.

[SSDK+21] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[VK21] Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder, 2021.

[VKK21] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space, 2021.