Noise Conditional Score Networks and Stochastic Differential Equations

Trevor Norton

November 2, 2023

Paper 1: Generative Modeling by Estimating Gradients of the Data Distribution

In [1], a separate approach to generative modeling is proposed:

Problem: Generate samples from some (possibly high-dimensional/complex) distribution $p_{\text{data}}(\mathbf{x})$

Idea: Learn the (Stein) score of the distribution: $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$

Challenge: There may be low-density areas in the ambient space for which the score is hard to approximate/undefined.

Solution: Perturb the distribution with different levels of Gaussian noise and estimate those scores.

Framework

Distribution to learn: $p_{\text{data}}(\mathbf{x}_0)$

Noise levels:* $\sigma_1 < \sigma_2 < \cdots < \sigma_L$

Perturbed distributions: $q_\sigma(\mathbf{x}) = \int \mathcal{N}(\mathbf{x}; \mathbf{x}', \sigma^2 \mathbf{I}) \, \mathrm{d}\mathbf{x}'$

$q_{\sigma_1}(\mathbf{x}) \approx p_{\text{data}}(\mathbf{x})$

$q_{\sigma_L}(\mathbf{x}) \approx \mathcal{N}(\mathbf{x}; \mathbf{0}, \sigma_L \mathbf{I})$

Score network: $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, \sigma) \approx \nabla_{\mathbf{x}} q_\sigma(\mathbf{x})$

Training

There are many techniques for learning the score of a distribution. We use *denoising score matching* for training.

1. Sample a point from distribution.
2. Perturb the point with Gaussian noise.
3. Score should try to reverse the perturbation.

$$\ell(\boldsymbol{\theta}; \sigma) := \frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{\tilde{\mathbf{x}}\sim\mathcal{N}(\mathbf{x},\sigma^2\mathbf{I})}\left[\left\|\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}, \sigma) - \frac{\tilde{\mathbf{x}} - \mathbf{x}}{\sigma^2}\right\|^2\right]$$

$$\mathcal{L}(\boldsymbol{\theta}; \{\sigma_k\}_{k=1}^{L}) := \frac{1}{L}\sum_{i=1}^{L}\lambda(\sigma_i)\ell(\boldsymbol{\theta}; \sigma_i)$$

Sampling

We know that if the σ's are close then the distributions should be approximately the same. So to sample, we start at the highest noise level (which is approximately normal) and refine the sample at each noise level.

Langevin MCMC:

$$\mathbf{x}_t \leftarrow \mathbf{x}_{t-1} + \frac{\epsilon}{2}\nabla_{\mathbf{x}} \log q_{\sigma_i}(\mathbf{x}_{t-1})$$

Put simply: to sample q_{σ_i} we do gradient ascent to approach areas of high likelihood...

Sampling

We know that if the σ's are close then the distributions should be approximately the same. So to sample, we start at the highest noise level (which is approximately normal) and refine the sample at each noise level.

Langevin MCMC:

$$\mathbf{x}_t \leftarrow \mathbf{x}_{t-1} + \frac{\epsilon}{2}\nabla_{\mathbf{x}} \log q_{\sigma_i}(\mathbf{x}_{t-1}) + \sqrt{\epsilon}\mathbf{z}_t$$

Put simply: to sample q_{σ_i} we do gradient ascent to approach areas of high likelihood...and add noise to fill out the region. The stationary distribution is $q_{\sigma_i}(\mathbf{x})$.

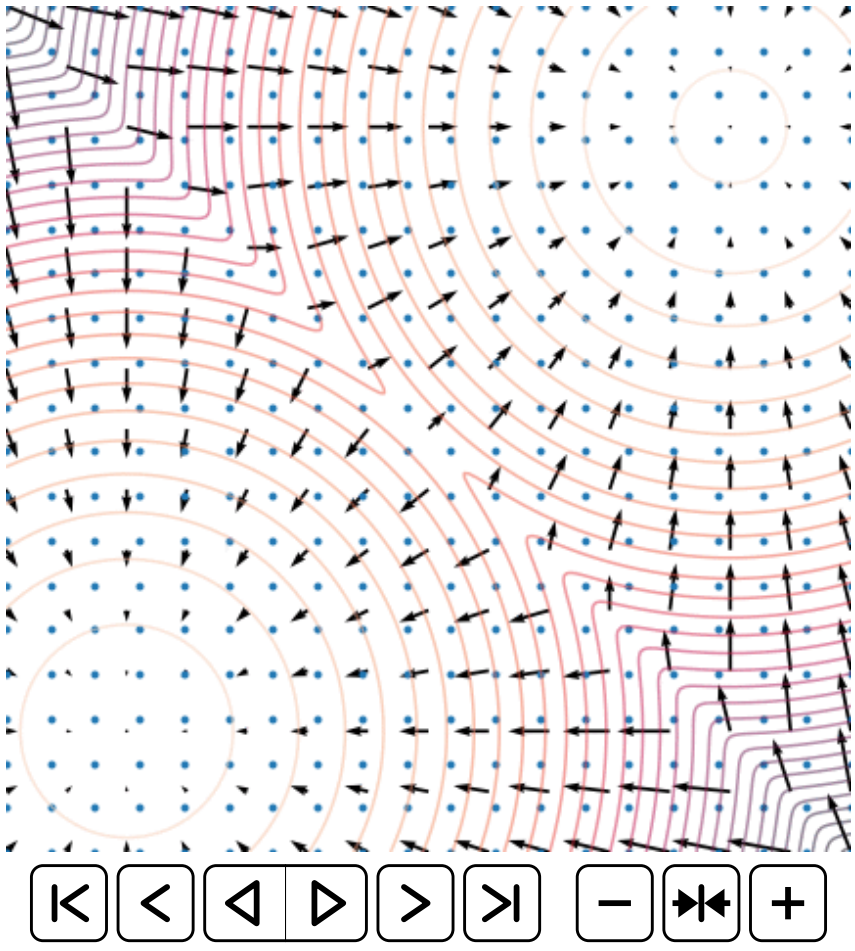

Figure: Example of Langevin MCMC method converging to underlying distribution [2].

Paper 2: Score-based Generative Modeling through Stochastic Differential Equations

In [3], previous diffusion models are viewed as specific discretizations of continuous time models.

The diffusion process is now viewed as a stochastic differential equation (SDE) that continuously adds Gaussian noise until reaching a stationary distribution.

The framework provides new benefits such as a deterministic way to sample the distribution and flexibility in designing new diffusion models

Diffusion with SDE

Starting with $\mathbf{x}(0) \sim p_0$

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

The term $\mathrm{d}\mathbf{w}$ can be thought of as Gaussian "white noise". After a sufficiently long time we have $\mathbf{x}(T) \sim p_T$ where p_T is (typically) approximately normal.

The process can be reversed with the following SDE:

$$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] \mathrm{d}t + g(t)\mathrm{d}\overline{\mathbf{w}}$$

If the score $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ is known, then we can recover p_0 starting from p_T.

Training and Sampling

Training the score network $\mathbf{s}_\theta(\mathbf{x}, t)$ typically is done with denoising score matching:

$$\mathbb{E}_t\Big\{\lambda(t)\mathbb{E}_{p_0(\mathbf{x}')}\mathbb{E}_{p_{0,t}(\mathbf{x}|\mathbf{x}')}\left[\|s_\theta(\mathbf{x}, t) - \nabla_{\mathbf{x}} \log p_{0,t}(\mathbf{x} \mid \mathbf{x}')\|^2\right]\Big\}.$$

For affine/linear SDEs, it is possible to get a closed form for the transition kernel $p_{0t}(\mathbf{x}(t) \mid \mathbf{x}(0))$. Otherwise, one can use sliced score matching or another technique.

Sampling is done by simulating the reverse SDE.

Deterministic Sampling

Deterministic ODE with same marginal probabilities as the reverse SDE:

$$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] \mathrm{d}t$$

1. Exact likelihood computation: one can compute $\log p_0(\mathbf{x}_0)$ for a sampled $\mathbf{x}_0$)
2. Unique encodings: can encode $\mathbf{x}(0)$ into latent space $\mathbf{x}(T)$ and (given sufficient data and model capacity) this encoding is uniquely determined by the data distribution
3. Efficient sampling: can simulate ODE to generate samples more quickly than the SDE (although sample quality suffers)


Yang Song and Stefano Ermon.
Generative Modeling by Estimating Gradients of the Data Distribution.
In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.


Generative Modeling by Estimating Gradients of the Data Distribution.
`https://yang-song.net/blog/2021/score/`.
Accessed: 2023-11-1.


Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole.
Score-Based Generative Modeling through Stochastic Differential Equations, November 2020.