2025_ML_assignment7_written: Score Matching and Its Role in Diffusion Models

Kai-Hung Cheng

2025/10/21

1. Score Matching and Its Role in Diffusion Models

Score matching is a technique for training generative models without directly estimating the full probability density of the data. Normally, learning a data distribution $p(x)$ is hard because we need to know the normalization constant (the partition function), which is often intractable. Instead of learning $p(x)$ itself, score matching focuses on learning its *score function*:

$$S(x) = \nabla_x \log p(x),$$

which represents the gradient of the log-density with respect to the input x. Intuitively, $S(x)$ points toward the direction where the probability density increases the fastest — in other words, toward regions where data is more likely to appear.

The original (explicit) score matching loss compares a model's score $S(x;\theta)$ to the true score $\nabla_x \log p(x)$:

$$L_{\mathrm{ESM}} = \mathbb{E}_{p(x)} \| S(x;\theta) - \nabla_x \log p(x) \|^2.$$

However, we can't compute $\nabla_x \log p(x)$ directly. To avoid this, the loss can be rewritten (using integration by parts) into an equivalent form that depends only on $S(x;\theta)$ and its divergence — this is called *implicit score matching (ISM)*.

In practice, a more efficient version called *denoising score matching (DSM)* is widely used. We add Gaussian noise to data x_0 to get a noisy sample $x = x_0 + \sigma\epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$. The model then learns to predict the conditional score

$$\nabla_x \log p(x|x_0) = -\frac{x - x_0}{\sigma^2}.$$

The DSM loss becomes

$$L_{\text{DSM}} = \mathbb{E}_{x_0,\epsilon}\frac{1}{\sigma^2}\|\sigma S_\sigma(x_0+\sigma\epsilon;\theta)+\epsilon\|^2,$$

which means the model tries to predict the noise ϵ that was added to the data. This is exactly the idea behind diffusion models.

In score-based (or diffusion) generative models, we train a neural network $S(x, t; \theta)$ to estimate the score of progressively noisier data distributions $p_t(x)$. Once the model learns to approximate $\nabla_x \log p_t(x)$ for all noise levels t, we can generate new samples by starting from random Gaussian noise and gradually "denoising" it — either through Langevin dynamics or by solving the reverse-time SDE or ODE defined by the score function.

In short, score matching teaches the model how to move data points back toward regions of higher density, and diffusion models turn this idea into a powerful generative process by chaining together many denoising steps across noise levels.

2. Unanswered Questions

1. In real training, how much does the performance of a diffusion model depend on the noise schedule we pick? If the schedule isn't well designed, could that make the training unstable or lead to blurry generations?

2. Since the DSM loss changes with the noise level σ, what's a good way to pick or balance different σ values so the model can learn both the big picture and the fine details of the data?