Generative models based on ODE and SDE

Denis Rakitin
rakitindenis32@gmail.com

1 Introduction

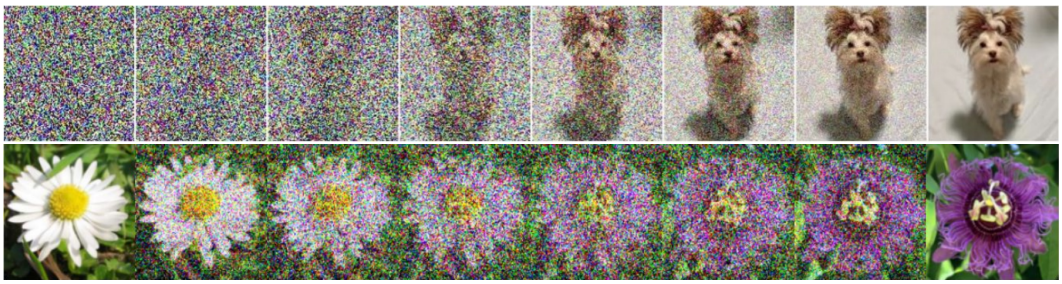

Figure 1: Top: denoising process, defined by a diffusion model [16]. Bottom: stochastic interpolation between two pictures [1].

This tutorial aims to introduce the reader to the mathematical methods which are widely used in contemporary generative modeling. Big practical success of diffusion models [7, 3, 16], which generate a sequence of pictures instead of just the target one, led to development of a novel family of models that describe some dynamic processes. Almost all processes occuring in the real world can be described by differential equations, which will be the basis of the generative models, reviewed here. We will cover methods based on either ordinary (ODE) or stochastic (SDE) differential equations, applicable for generation [16, 10, 17, 2, 1], paired [17, 2, 1, 11] and unpaired [12, 14, 9, 6] domain translation, formalized as an instance of the optimal transport problem.

2 Diffusion models

Generally speaking, diffusion models define a process of step-by-step noising of a picture and try to learn the reverse process, which allows to generate new images starting with a pure noise. There are different approaches for formalizing this concept. The earlier score-based models as NCSN [15] learn score functions of probability distributions at all noise levels and sample with a consequent Langevin dynamics [18] going from larger to smaller noise levels. In [7] the backward denoising process is trained as a latent variable model with a variational distribution on noisy images, corresponding to a forward noising process. Finally, the authors of [16] consider a continuous-time noising SDE and construct the reverse, which can be seen as a unifying framework for previous approaches. On the one hand, the backward SDE explicitly requires knowing score functions at all noise levels, which is similar to the earlier score-based models. On the other hand, discrete-time sampling schemes from the backward SDE result in a discrete-time process very similar to the one in [7]. In this section, we will cover the original score-based approach and models, based on SDE.

2.1 Score-based models

The general problem of generative modeling consists of constructing the probabilistic model p_θ given a data set $X_1, \ldots, X_n \sim p_{data}$ in such a way that $p_\theta \approx p_{data}$.

There are mainly 3 families of non-diffusion generative models: Generative Adversarial Networks (GANs) [5], Variational Autoencoders (VAEs) [8] and Normalizing Flows (NFs) [4, 13]. Despite different training procedures, all of them share the same simple generation scheme: picture x is obtained by transforming noise z by a trained network G_θ: $x = G_\theta(z)$. In contrast, score-based models do not learn the generator explicitly, but try to approximate the score function of the distribution: $s_\theta(x) \approx \nabla_x \log p_{data}(x)$. But why is this statistic valuable?

First, it can be used for finding mode of the distribution by gradient ascent: if

$$X_{t+1} = X_t + \gamma \nabla \log p_{data}(X_t),$$

then

$$X_t \xrightarrow[t\to\infty]{} x* = \arg\max_x \log p_{data}(x) = \arg\max_x p_{data}(x).$$

Of course, in general multi-modal case method converges to one of the stationary points of $\log p_{data}(x)$.

Furthermore, it can be used to obtain samples from the distribution using Langevin dynamics [18]:

$$X_{t+1} = X_t + \gamma \nabla \log p_{data}(X_t) + \sqrt{2\gamma}\, \varepsilon_t,$$

where $\varepsilon_t \sim \mathcal{N}(0, I)$ is independent with X_t. In regular cases, one can prove that

$$p_{X_t} \xrightarrow[t\to\infty]{} p^\gamma,$$

where p^γ is a distribution close to p_{data} that depends on a discretizarion step and converges to p_{data}, when γ converges to zero:

$$p^\gamma \xrightarrow[\gamma\to 0]{} p_{data}.$$

Given this, score-based models obtain sample from the model by running Langevin dynamics with trained approximation of the score function s_θ instead of the true one:

$$X_{t+1} = X_t + \gamma s_\theta(X_t) + \sqrt{2\gamma}\, \varepsilon_t.$$

The only question remained is how to train it. Ideally, we would like to match our approximation with the true score function on the samples from data set and solve the Score Matching objective

$$\mathbb{E}\|s_\theta(X) - \nabla \log p_{data}(X)\|^2 \to \min_\theta, \tag{1}$$

where $X \sim p_{data}$. Unfortunately, this objective has 2 issues:

1. Trivially, we do not know the true score function and cannot compute it to perform regression;
2. It is common to assume that such high-dimensional structured data like images lie in a much less-dimensional manifold. In this case, data distribution p_{data} does not have density in a regular sence. Even if the data distribution is not strictly concentrated on a manifold, but is very close, there will be very rapid transitions from zero to high-density regions, which will result in large unstable values of $\nabla \log p_{data}$.

2.2 Denoising score matching

To address the second problem, one can make the data distribution more smooth by, for example, adding gaussian noise:

$$\hat{X} = X + \sigma\varepsilon,\ \varepsilon \sim \mathcal{N}(0, I),$$

or, more generally, deleting part of the object and replacing it with gaussian noise:

$$\hat{X} = \alpha X + \sigma\varepsilon,\ \varepsilon \sim \mathcal{N}(0, I),\ \alpha < 1.$$

In terms of distributions, we defined a conditional distribution

$$p_{\hat{X}|X}(\hat{x}|x) = \mathcal{N}(\hat{x} \mid \alpha x, \sigma^2 I).$$

The perturbed variable $\hat{X}$ always has density, which can be represented as

$$p_{\hat{X}}(\hat{x}) = \int p_{\hat{X}|X}(\hat{x}|x) p_X(x) \mathrm{d}x. \tag{2}$$

Consequently, it has a score function $\nabla \log p_{\hat{X}}(\hat{x})$, which can be theoretically learned by optimizing the score matching objective

$$\mathbb{E}\|s_\theta(\hat{X}) - \nabla \log p_{\hat{X}}(\hat{X})\|^2 \to \min_\theta,$$

which is, however, still intractable.

The representation of density in the Equation 2 is important, because it rewrites a complicated density $p_{\hat{X}}$ as an integral of a very easy conditional density $p_{\hat{X}|X}$ with respect to the distribution p_X, from which we have a data set of samples. I turns out that the score function of $p_{\hat{X}}$ has a similar representation. To derive it, we start with differentiating logarithm (so-called log-derivative trick):

$$\nabla_{\hat{x}} \log p_{\hat{X}}(\hat{x}) = \frac{\nabla_{\hat{x}} p_{\hat{X}}(\hat{x})}{p_{\hat{X}}(\hat{x})}.$$

Next, we use the representation of density $p_{\hat{X}}(\hat{x})$ in Eq. 2 and obtain

$$\frac{\nabla_{\hat{x}} \int p_{\hat{X}|X}(\hat{x}|x) p_X(x) \mathrm{d}x}{p_{\hat{X}}(\hat{x})} = \frac{\int \nabla_{\hat{x}}\, p_{\hat{X}|X}(\hat{x}|x) p_X(x) \mathrm{d}x}{p_{\hat{X}}(\hat{x})}.$$

Finally, we apply the reversed log-derivative trick to $\nabla_{\hat{x}} p_{\hat{X}|X}(\hat{x}|x)$ and obtain

$$\frac{\int \nabla_{\hat{x}} \log p_{\hat{X}|X}(\hat{x}|x) \cdot p_{\hat{X}|X}(\hat{x}|x) p_X(x) \mathrm{d}x}{p_{\hat{X}}(\hat{x})} = \int \nabla_{\hat{x}} \log p_{\hat{X}|X}(\hat{x}|x) \cdot p_{X|\hat{X}}(x|\hat{x}) \mathrm{d}x,$$

which is just the conditional expectation of the conditional score function. Thereby, we obtained:

$$\nabla_{\hat{x}} \log p_{\hat{X}}(\hat{x}) = \mathbb{E}\left[\nabla_{\hat{x}} \log p_{\hat{X}|X}(\hat{X}|X) \mid \hat{X} = \hat{x}\right]. \tag{3}$$

Given this, the score matching objective can be rewritten as

$$\mathbb{E}\left\| s_\theta(\hat{X}) - \mathbb{E}\left[\nabla_{\hat{x}} \log p_{\hat{X}|X}(\hat{X}|X) \mid \hat{X}\right] \right\|^2 \to \min_\theta. \tag{4}$$

Conditional expectation is very convenient to work with since it is the best predictor in the L_2 sense: for all pairs of r.v. (X, Y)

$$g^*(x) = \mathbb{E}[Y \mid X = x] = \arg\min_g \mathbb{E}\|g(X) - \mathbb{E}\left[Y|X\right]\|^2 = \arg\min_g \mathbb{E}\left\|g(X) - Y\right\|^2,$$

which means that the objective in Eq. 4 is equivalent to the so-called «denoising score matching» objective

$$\mathbb{E}\|s_\theta(\hat{X}) - \nabla_{\hat{x}} \log p_{\hat{X}|X}(\hat{X}|X)\|^2 \to \min_\theta. \tag{5}$$

Surprisingly, by addressing the problem of sharp density transitions, we also implicitly solved the intractability of the objective! The conditional score function is just the score function of the normal distribution, which can be calculated. For d-dimensional normal distribution

$$p_{\hat{X}|X}(\hat{x}|x) = \mathcal{N}(\hat{x}|\alpha x,\, \sigma^2 I) = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(-\frac{1}{2\sigma^2}\left\|\hat{x} - \alpha x\right\|^2\right)$$

log-density is equal to

$$\log p_{\hat{X}|X}(\hat{x}|x) = \text{const} - \frac{1}{2\sigma^2}\|\hat{x} - \alpha x\|^2,$$

which gives the score function

$$\nabla_{\hat{x}} \log p_{\hat{X}|X}(\hat{x}|x) = -\frac{1}{\sigma^2}(\hat{x} - \alpha x) = \frac{1}{\sigma^2}(\alpha x - \hat{x}).$$

Finally, we arrive at optimizing the objective

$$\mathbb{E}\left\| s_\theta(\hat{X}) - \frac{1}{\sigma^2}(\alpha X - \hat{X}) \right\|^2 \to \min_\theta,$$

Algorithm 1 Sampling from a score-based model

$X^0 \sim p_0$
for $m = 1, \dots, M$ **do**
  $\varepsilon^{(m)} \sim \mathcal{N}(0, I)$ — independent with $X^{(m-1)}$
  $X^{(m)} = X^{(m-1)} + \gamma_m s_\theta(X^{(m-1)}) + \sqrt{2\gamma_m}\,\varepsilon^{(m)}$ ▷ Langevin step
end for
return $X^{(M)}$

which minimal value is obtained at the score function of the perturbed data distribution:

$$s_{\theta^*}(\hat{x}) = \nabla_{\hat{x}} \log p_{\hat{X}}(\hat{x}) = \nabla_{\hat{x}} \log \int p_{data}(x)\mathcal{N}(\hat{x}|\alpha x,\, \sigma^2 I)\mathrm{d}x.$$

To sample from a trained score-based model, one can just apply Langevin dynamics, which is summarized in the Algorithm 1.

Formally, we obtained algorithm which allows to sample from a slightly modified data distribution. In theory, one can obtain a very close approximation of p_{data} by setting $\alpha \approx 1$ and $\sigma^2 \approx 0$. In practice, however, one should keep in mind that when they get close to these values, transitions of density become sharp, score function starts to take large values and the training procedure becomes unstable. This leads to a trade-off between precision of the approximation and stability of the model.

2.3 Noise Conditional Score Networks

To overcome the necessity of choosing α and σ^2 and balancing between two qualities of the model, authors of [15] present a very elegant idea: consider a sequence of modified variables $\{X_t\}_{t=1}^T$ instead of one $\hat{X}$. This sequence of variables will represent a process of gradual noising of the image and cover the whole spectrum of noisy distributions: from sharp distributions close to p_{data} to the pure noise like $\mathcal{N}(0, I)$. Formally, this sequence should possess 3 qualities:

1. The first variable X_1 should be close to the data distribution: $p_{X_1} \approx p_{data}$. This will ensure that sampling from p_{X_1} will be almost equivalent to sampling from p_{data};
2. The last variable X_T should be a very simple distribution to sample from, for example, $\mathcal{N}(0, \sigma^2)$;
3. Distributions of X_t and X_{t+1} should be close to each other. This will make X_{t+1} a good initialization point for sampling from p_{X_t}.

Below, we will use notation of the form $p_t(x_t) := p_{X_t}(x_t)$ or $p_{t|s}(x_t|x_s) := p_{X_t|X_s}(x_t|x_s)$ to make it shorter. Authors consider a so-called variance-exploding (VE) process and define

$$X_t = X_0 + \sigma_t^2 \varepsilon,$$

where $\varepsilon \sim \mathcal{N}(0, I)$ is independent from X_0 and σ_t is an increasing sequence of variances. Equivalently,

$$p_{t|0}(x_t|x_0) = \mathcal{N}(x_t|x_0, \sigma_t^2 I).$$

Taking $\sigma_0 \approx 0$, $\sigma_t \approx \sigma_{t+1}$ and σ_T^2 of such magnitude, that $p_T \approx \mathcal{N}(0, \sigma_T^2)$, one ensures to match all the 3 requirements. This sequence of distributions corresponds to adding more and more noise to the original distribution, until it becomes indistinguishable from the pure noise with large magnitude.

The more popular process now is the variance preserving process, which defines a Markov chain

$$X_{t+1} = \sqrt{1-\beta_t} X_t + \sqrt{\beta_t}\varepsilon_t,$$

where $\varepsilon_t \sim \mathcal{N}(0, I)$ is independent from X_t. This defines a conditional distribution given previous time step:

$$p_{t+1|t}(x_{t+1}|x_t) = \mathcal{N}(x_{t+1}|\sqrt{1-\beta_t}x_t, \beta_t I).$$

Given this, one can calculate the conditional distribution given the initial variable and obtain [7]

$$p_{t|0}(x_t|x_0) = \mathcal{N}(x_t|\alpha_t x_0, \sigma_t^2 I),$$

where $\alpha_t = \sqrt{\prod_{s=1}^{t}(1-\beta_s)} \to 0$ and $\sigma_t^2 = 1 - \prod_{s=1}^{t}(1-\beta_s) \to 1$ given a proper choice of β_t. Choosing small enough β_t to ensure $p_{t+1} \approx p_t$ and fulfilling $\alpha_T \approx 0$ and $\sigma_T^2 \approx 1$, one will satisfy all the 3 requirements.

Now, given a sequence of modified distributions with easy conditional distributions, one can train score for all of them with denoising score matching:

$$\mathbb{E}\|s_\theta(X_t, t) - \nabla \log p_{t|0}(X_t|X_0)\|^2 \to \min_\theta .$$

In practice, training T neural networks is very inefficient, that is why $s_\theta(x, t)$ is defined as one neural network with two inputs. Besides, scores for different t are connected and the training signal from one time step is beneficial for another. Thus, the final training procdure consists of taking the weighted sum of denoising score matching losses from all the time steps:

$$\sum_{t=1}^{T} \gamma(t)\mathbb{E}\|s_\theta(X_t, t) - \nabla \log p_{t|0}(X_t|X_0)\|^2 \to \min_\theta . \tag{6}$$

This model (paired with the sampling Algorithm 2) is called Noise Conditional Score Network [15]. Theoretically, $s_\theta(x, t)$ matches $\nabla \log p_t(x)$ after training. Sampling procedure consists of the same Langevin dynamics, but now performed sequentially for all of the time steps backwards. Here the 3 properties of the sequence p_t become crucial: generating from p_T is easy, sample from p_t is a good point to start dynamic for p_{t-1} and sample from p_1 is almost a sample from p_{data}. Formally, the sampling procedure is written in the Algorithm 2.

Algorithm 2 Sampling from a Noise Conditional Score Network (NCSN)

$X_T^{(M)} \sim p_T$
for $t = T-1, \dots, 1$ **do**
  $X_t^{(0)} = X_{t+1}^{(M)}$
  for $m = 1, \dots, M$ **do** ▷ Langevin dynamics for p_t
    $\varepsilon_t^{(m)} \sim \mathcal{N}(0, I)$ — independent with $X_t^{(m-1)}$
    $X_t^{(m)} = X_t^{(m-1)} + \gamma_m s_\theta(X_t^{(m-1)}, t) + \sqrt{2\gamma_m}\,\varepsilon_t^{(m)}$ ▷ Langevin step
  end for
end for
return $X_1^{(M)}$

We obtained a model with sequential sampling, which needs to use a neural network multiple times. At the same time, it has a very efficient *simulation-free* training procedure, which does not require sampling from the model (and which distinguishes it from energy-based models and continuous normalizing flows). This makes this procedure inefficient in terms of sampling time, but allows to extract a lot of additional information compared to using NN just once. This combination is believed to be very powerful in practice, that is why constructing the analogues for problems other than generation could be very beneficial.

The only major thing that is improved in the newer algorithms is the sampling scheme. It seems like running Langevin dynamics for each time step can be optimized in a way to perform just one step from t to $t-1$. One of the possible ways to derive such sampling scheme is through stochastic differential equations.

2.4 Ordinary and stochastic differential equations

References

[1] Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

[2] Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022.

[3] Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in Neural Information Processing Systems*, 34:17695–17709, 2021.

[4] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[6] Nikita Gushchin, Alexander Kolesov, Alexander Korotin, Dmitry Vetrov, and Evgeny Burnaev. Entropic neural optimal transport via diffusion processes. *arXiv preprint arXiv:2211.01156*, 2022.

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[8] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[9] Alexander Korotin, Daniil Selikhanovych, and Evgeny Burnaev. Neural optimal transport. *arXiv preprint arXiv:2201.12220*, 2022.

[10] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[11] Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A Theodorou, Weili Nie, and Anima Anandkumar. I^2 sb: Image-to-image schr\" odinger bridge. *arXiv preprint arXiv:2302.05872*, 2023.

[12] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[13] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

[14] Yuyang Shi, Valentin De Bortoli, Andrew Campbell, and Arnaud Doucet. Diffusion schr\" odinger bridge matching. *arXiv preprint arXiv:2303.16852*, 2023.

[15] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[16] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[17] Alexander Tong, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Kilian Fatras, Guy Wolf, and Yoshua Bengio. Conditional flow matching: Simulation-free dynamic optimal transport. *arXiv preprint arXiv:2302.00482*, 2023.

[18] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011.