Mathmatical Foundations for Deep Generative Models

Michael Jin

github repo

June 9, 2025

Ultimate Goal

$$\theta^* = \arg\min_\theta \mathcal{D}\left(p_{\text{data}}(x) \,||\, p_\theta\right)$$

- p_θ: model parameterized by θ; e.g., neural network in DGM
  - Q: what does the model stand for?
  - A: **probability density function** for data distribution
- p_{data}: real data distribution in terms of p.d.f.
  - Q: do we really have?
  - A: not really
- $\mathcal{D}$: a measure of 'distance'

Discrete v.s. Continuous

Given sample data:

$$A = \{x^1, x^2, \ldots, x^N\}, \quad x^i \in \mathbb{R}^d, \quad x^i \overset{\text{i.i.d.}}{\sim} p_{\text{data}}(x)$$

we can define the empirical distribution for estimating real distribution as:

$$\boxed{p_N(x) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}_{x^i \in A} = \frac{1}{N} \sum_{i=1}^{N} \delta(x - x^i)}$$

- Probability mass function:
$$\int p_N(x) \mathrm{d}x = \frac{1}{N} \sum_{i=1}^{N} \int \delta(x - x^i) \mathrm{d}x = 1$$
- On expectation: $\mathbb{E}_{x \sim p_N(x)}[f(x)] = \sum_{i=1}^{N} f(x^i)$

Measure divergence

By information theory, optimal bit size for x is $I(x) = -\log p(x)$. Hence we can define

- Information entropy for p:

$$H(p) = \mathbb{E}_{x \sim p}[-\log p(x)] = -\int p(x) \log q(x) \mathrm{d}x$$

- Cross entropy if use q to encode p:

$$H(p, q) = -\int p(x) \log q(x) \mathrm{d}x$$

So how many bits are wasted if we use q to encode p?

$$\mathrm{D}_{\mathrm{KL}}(p \,||\, q) = H(p, q) - H(p) = \int p(x) \log \frac{p(x)}{q(x)} \mathrm{d}x$$

Lower Bound

$$
\begin{aligned}
\mathrm{D}_{\mathrm{KL}}(p \mid\mid q) &= \int p(x) \log \frac{p(x)}{q(x)} \mathrm{d}x \\
&= -\int p(x) \log \frac{q(x)}{p(x)} \mathrm{d}x \\
&\geq -\log \int p(x) \frac{q(x)}{p(x)} \mathrm{d}x \\
&= -\log \int q(x) \mathrm{d}x = -\log(1) = 0
\end{aligned}
$$

What about upper bound?

Consider for some x, $q(x) = 0$; then $|\mathrm{D}_{\mathrm{KL}}| \longrightarrow \infty$
Problem: Exploding Gradients

Solution

We can measure the distance to average distribution instead. Let $m(x) = \frac{1}{2}[p(x) + q(x)]$, then we define

$$\boxed{\mathrm{D}_{\mathrm{JS}}(p \,||\, q) = \frac{1}{2}\mathrm{D}_{\mathrm{KL}}(p \,||\, m) + \frac{1}{2}\mathrm{D}_{\mathrm{KL}}(q \,||\, m)}$$

- $\mathrm{D}_{\mathrm{JS}}(p \,||\, q) = \mathrm{D}_{\mathrm{JS}}(q \,||\, p)$
- Lower bound: $\mathrm{D}_{\mathrm{JS}} \geq 0$ since $\mathrm{D}_{\mathrm{KL}} \geq 0$
- Upper bound:

Upper Bound

$$
\begin{aligned}
\mathrm{D}_{\mathrm{KL}}(p \,||\, m) &= \int p(x) \log \frac{p(x)}{m(x)} \mathrm{d}x \\
&= \int p(x) \log \frac{p(x)}{\frac{1}{2}[p(x) + q(x)]} \mathrm{d}x \\
&\leq \int p(x) \log \frac{2}{1 + \frac{q(x)}{p(x)}} \mathrm{d}x \\
&\leq \log 2 \int p(x) \mathrm{d}x = \log 2
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
\mathrm{D}_{\mathrm{JS}}(p \,||\, q) &= \frac{1}{2}\mathrm{D}_{\mathrm{KL}}(p \,||\, m) + \frac{1}{2}\mathrm{D}_{\mathrm{KL}}(q \,||\, m) \\
&\leq \frac{1}{2} \cdot 2 \log 2 = \log 2
\end{aligned}
$$

MLE

Back to $A = \{x^i\}_{i=1}^N$, their joint probability for appearing in p_θ:

$$p_\theta(x^1, x^2, \ldots, x^N) = \prod_{i=1}^N p_\theta(x^i)$$

We want to maximize this probability, so

$$\begin{aligned} \theta^* &= \arg\max_\theta \prod_{i=1}^N p_\theta(x^i) \\ &= \arg\max_\theta \sum_{i=1}^N \log p_\theta(x^i) \\ &= \arg\max_\theta \mathbb{E}_{x \sim p_N}[\log p_\theta(x)] \end{aligned}$$

$$\boxed{\hat{\theta}_{\text{MLE}} = \arg\max_\theta \mathbb{E}\left[\log p_\theta(x)\right]}$$

MLE & KLD

$$
\begin{aligned}
\mathrm{D}_{\mathrm{KL}}(p_N \,||\, p_\theta) &= \sum_{i=1}^{N} p_N \log \frac{p_N(x)}{p_\theta(x)} \\
&= \sum_{i=1}^{N} p_N \log p_N(x) - \sum_{i=1}^{N} p_N(x) \log p_\theta(x) \\
&= H(p_N) - \sum_{i=1}^{N} p_N(x) \log p_\theta(x)
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
\theta^* &= \arg\min_\theta \mathrm{D_{KL}}(p_{\text{data}} \,||\, p_\theta) \\
&= \arg\min_\theta \left[H(p_N) - \sum_{i=1}^{N} p_N(x) \log p_\theta(x) \right] \\
&= \arg\max_\theta \sum_{i=1}^{N} p_N(x) \log p_\theta(x) \\
&= \arg\max_\theta \mathbb{E}_{x \sim p_N(x)} [\log p_\theta(x)]
\end{aligned}
$$

$$
\boxed{\hat{\theta}_{\text{MLE}} = \hat{\theta}_{\mathrm{D_{KL}}}}
$$

Next Step

Here we finish some basic knowledge check. Back to the first slide, how can we build the model of $p_\theta(x)$? Some known approaches:

- Explicitly write down $p_\theta(x)$
- Implicitly get $p_\theta(x)$: use latent space
- learn a function about $p_\theta(x)$

Autoregressive

Write $x = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$, then

$$p(x) = p(x_1) \cdot p(x_2 \mid x_1) \cdot p(x_3 \mid x_1, x_2) \cdots p(x_d \mid x_{<d})$$

That is,

$$\boxed{p(x) = \prod_{t=1}^{d} p_\theta(x_t \mid x_{<t})}$$

Hence,

$$\begin{aligned}
\theta^* &= \arg\max_\theta \sum_{i=1}^{N} \log p_\theta(x^i) \\
&= \arg\max_\theta \sum_{i=1}^{N} \sum_{t=1}^{T} \log p_\theta(x_t^i \mid x_{<t}^i)
\end{aligned}$$

Normalizing Flow (1)

Introduce a simple distribution $z \sim p_Z(z)$ in the latent space $\mathbb{R}^d$. Assume there exist a bijective function $f_\theta : z \longrightarrow x$ such that $f_{\theta\#} p_Z = p_X$ (push-forward measure).

By change of variable:

$$\mathrm{d}x = \left| \det \frac{\partial f}{\partial x} \right| \mathrm{d}z := |J_f(z)\mathrm{d}z|$$

and the probability conservation, we have:

$$\begin{aligned} p_X(x)\mathrm{d}x &= p_Z(z)\mathrm{d}z \\ &= p_Z(z) \left| \det \frac{\partial f}{\partial z} \right|^{-1} \mathrm{d}x \qquad (*) \end{aligned}$$

Normalizing Flow (2)

Recall that we have $(f^{-1} \circ f)(z) = I_d(z)$ for $\forall z \sim p_Z(z)$, so by chain rule,

$$\frac{\partial f^{-1}}{\partial x} \cdot \frac{\partial f}{\partial z} = I_d \Rightarrow \left(\frac{\partial f}{\partial z}\right)^{-1} = \frac{\partial f^{-1}}{\partial x}$$

Hence $(*) \iff$

$$p_X(x) = p_Z(z) \left| \det \frac{\partial f_\theta^{-1}}{\partial x} \right|$$

Or,

$$\boxed{p_\theta(x) = p_Z(f_\theta^{-1}(x)) \left| \det \frac{\partial f_\theta^{-1}}{\partial x} \right|}$$

Normalizing Flow (3)

And it follows that

$$\begin{aligned}\theta^* &= \arg\max_\theta \mathbb{E}_{x\sim p_N} \log p_\theta(x) \\ &= \arg\max_\theta \left[\log p_Z(f_\theta^{-1}) + \log\left|\det \frac{\partial f_\theta^{-1}}{\partial x}\right|\right]\end{aligned}$$

Base log-density + volume correction

GAN (1)

- In NF: f has to be bijective, and requires a tractable Jacobian determinant vspace1em
- In GAN: Still assume simple distribution $z \sim p_Z(z)$ in the latent space $\mathbb{R}^k$.
  Define generator $G_\theta : \mathbb{R}^k \to \mathbb{R}^d$ such that $x = G_\theta(z)$, where G is not necessarily bijective.
  - Problem: Unable to write down $\log p_\theta(x)$ & perform MLE
  - Solution: Add discriminator $D_\phi : \mathbb{R}^d \to [0, 1]$ as supervising signal

Value function:

$$\boxed{V(D, G) = \mathbb{E}_{x \sim p_N}[\log D(x)] + \mathbb{E}_{x \sim p_\theta}[\log(1 - D(x))]}$$

GAN (2)

'Zero-sum game':

$$\min_G \max_D V(D, G)$$

Goal: given that ϕ is optimized, find an optimized θ.

1. Find the optimal D_ϕ given G_θ fixed:

$$f(D) = p_N(x)\log(D(x)) + p_\theta(x)\log(1 - D(x))$$

$$\Rightarrow \frac{\mathrm{df}}{\mathrm{d}D} = \frac{p_N(x)}{D(x)} - \frac{p_\theta(x)}{1 - D(x)} := 0$$

$$\Rightarrow D^*(x) = \frac{p_N(x)}{p_N(x) + p_\theta(x)}$$

GAN (3)

2. Find the optimal G_θ given optimal D_ϕ:

Denote $m(x) = \frac{1}{2}[p_N(x) + p_\theta(x)]$, then

$$\begin{aligned}
V(D*, G) &= \mathbb{E}_{x\sim p_N}\left[\log \frac{p_N(x)}{2m(x)}\right] + \mathbb{E}_{x\sim p_\theta}\left[\log \frac{p_\theta(x)}{2m(x)}\right] \\
&= 2\left(\frac{1}{2}\mathbb{E}_{x\sim p_N}\left[\log \frac{p_N(x)}{m(x)}\right] + \frac{1}{2}\mathbb{E}_{x\sim p_\theta}\left[\log \frac{p_\theta(x)}{m(x)}\right] - \log 2\right) \\
&= 2\mathrm{D}_{\mathrm{JS}}(p_N \,||\, p_\theta) - 2\log 2
\end{aligned}$$

Therefore,

$$\boxed{\min_G V(D^*, G) \iff \arg\min_\theta \mathrm{D}_{\mathrm{JS}}(p_N \,||\, p_\theta)}$$

VAE (1)

- So far, latent to real is in point to point fashion
- What about point to distribution?

Simple distribution $z \sim p_Z(z)$ in latent space $\mathbb{R}^k$, find its ouput in p_N as a distribution / p.d.f.: $p_\theta(x|z)$.

Then we can model $p_\theta(x)$ using marginalization:

$$p_\theta(x) = \int p_\theta(x|z)p(z)\mathrm{d}z \qquad (**)$$

- Problem: z is continuous and in high-dimensional space, how to avoid integrate w.r.t. it?
- Solution: see the following trick about ELBO:

VAE (2)

Introduce another distribution $q_\phi(z|x)$, and by $(**)$:

$$\begin{aligned}
\log p_\theta(x) &= \log \int q_\phi(z|x) \frac{p_\theta(x,z)}{q_\phi(z|x)} \mathrm{d}x \\
&\geq \int q_\phi(z|x) \log \frac{p_\theta(x,z)}{q_\phi(z|x)} \mathrm{d}x \\
&= \mathbb{E}_{z\sim q_\phi(z|x)} \left[\log \frac{p_\theta(x|z)p(z)}{q_\phi(z|x)} \right] \\
&:= \mathrm{ELBO}(x)
\end{aligned}$$

Where

$$\mathrm{ELBO}(x) = \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z) - (\log q_\phi(z|x) - \log p(z))] \Longrightarrow$$

$$\boxed{\mathrm{ELBO}(x) = \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - \mathrm{D}_{\mathrm{KL}}(q_\phi(z|x) \,||\, p_Z(z))}$$

VAE (3)

We want to maximize log-likelihood, so we should maximize the **ELBO**:

$$\theta^*, \phi^* = \arg\max_{\theta,\phi} \mathbb{E}_{z \sim q_\phi(z|x)}[\log p_\theta(x|z)] - \mathrm{D}_{\mathrm{KL}}(q_\phi(z|x) \,||\, p_Z(z))$$

Base log-density + posterior correction

EBM (1)

Still, we want to momdel a $p_\theta(x)$ such that it has MLE to $p_N(x)$. Lets borrow some ideas from physics.

Inspired by Boltzman Distribution $p_i = \frac{1}{Z}\exp(-\frac{\epsilon_i}{kT})$, we can model the p.d.f. of data as proportional to their energy $E(x)$ (lower energy means higher density):

$$p_\theta(x) = \frac{\exp(-E_\theta(x))}{Z(\theta)}$$

where $Z(\theta) = \int \exp(-E_\theta(x))\mathrm{d}x$ is called the partition function.

EBM (2)

If the data is in low dimension, we can design (or fit) $E_\theta(x)$ and calculate the partition function to get log-likelihood.

For example, Hopfiled Network (which is the 2024 Nobel Prize for physics):

$$E_\theta(x) = -\frac{1}{2}x^T W_\theta x + b_\theta^T x$$

Sampling from EBM: Inspired by Langevin Dynamics,

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -\nabla_x E_\theta(x) + \sqrt{2\beta^{-1}}\eta(t)$$

we can make the sampling process like

$$x_{k+1} = x_k - \frac{\epsilon}{2}\nabla_x E_\theta(x_k) + \sqrt{\epsilon}\eta_k$$

from the SDE's Euler–Maruyama solution.

EBM (3)

But if the data is in high dimension, the partition function is hard to compute.
Solution: we can directly learn $\nabla_x E_\theta(x)$, since it is the only parametrized term in the sampling process. Rather then know where is the destination, we can just learn where to go!
Therefore,

$$\log p_\theta(x) = \log\left[\frac{\exp(-E_\theta(x))}{Z(\theta)}\right] = -E_\theta(x) - \log Z(\theta)$$

becomes

$$\nabla_x \log p_\theta(x) = \nabla_x[-E_\theta(x)] - \nabla_x[\log Z(\theta)] = -\nabla_x[-E_\theta(x)]$$

and hence we can learn the score function

$$\boxed{S_\theta(x) = \nabla_x \log p_\theta(x)}$$

SBM (1)

Recall that the score function is in fact the gradient field for probability distribution.
Before write down its detailed formulae, we need to first introduce the **Fisher Divergence**. It measures the difference of gradient fields for probability distributions:

$$\mathrm{D_F}(p \,||\, q) = \frac{1}{2} \int p(x) ||\nabla_x \log p(x) - \nabla_x \log q(x)||^2 \mathrm{d}x$$

So we can try to make the goal for SBM as following:

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{x \sim p_N} [||s_\theta(x) - \nabla_x \log p_N(x)||^2]$$

but we cannot get the gradient for a discrete function $p_N(x)$.

SBM (2)

Solution: lets go back to Fisher Divergence,

$$
\begin{aligned}
\mathrm{D_F}(p \mid\mid q) &= \frac{1}{2}\mathbb{E}_{x\sim p}\left[||\nabla_x \log p(x) - \nabla_x \log q(x)||^2\right] \\
&= \frac{1}{2}\mathbb{E}_{x\sim p}[||\nabla_x \log p(x)||^2 - 2\langle \nabla_x \log p(x), \nabla_x \log q(x)\rangle \\
&\quad + ||\nabla_x \log q(x)||^2] \qquad (\#)
\end{aligned}
$$

where

$$
\begin{aligned}
&\mathbb{E}_{x\sim p}\langle \nabla_x \log p(x), \nabla_x \log q(x)\rangle \\
&= \int p(x)[\nabla_x \log p(x)]^T \nabla_x \log q(x) \mathrm{d}x \\
&= \int p(x) \left[\sum_i \nabla_x \log p(x_i) \nabla_x \log q(x_i)\right] \mathrm{d}x \\
&= \sum_i \int p(x_i) \cdot \frac{\nabla_x p(x_i)}{p(x_i)} \nabla_x \log q(x_i) \mathrm{d}x
\end{aligned}
$$

SBM (3)

$$
\begin{aligned}
&= \sum_i \int \nabla_x \log q(x_i) \mathrm{d}[p(x_i)] \\
&= \sum_i \left([p(x_i)\nabla_x \log q(x_i)]_{\infty}^{\infty} - \int p(x_i)\nabla_x^2 \log q(x_i)\mathrm{d}x \right) \\
&= -\sum_i \int p(x_i)\nabla_x^2 \log q(x_i)\mathrm{d}x \\
&= -\int p(x) \left(\sum_i \nabla_x^2 \log q(x_i) \right) \mathrm{d}x \\
&= -\int p(x)\Delta_x \log q(x)\mathrm{d}x \\
&= -\mathbb{E}_{x\sim p(x)}[\Delta_x \log q(x)]
\end{aligned}
$$

SBM (4)

Then plug this back to (#) :

$$\mathrm{D_F}(p \,||\, q)$$

$$= \mathbb{E}_{x\sim p}\left(\frac{1}{2}||\nabla_x \log p(x)||^2 + 2\Delta_x \log q(x) + \frac{1}{2}||\nabla_x \log q(x)||^2\right)$$

Therefore,

$$\mathrm{D_F}(p_N \,||\, p_\theta)$$

$$= \mathbb{E}_{x\sim p_N(x)}\left(2\Delta_x \log p_\theta(x) + \frac{1}{2}||\nabla_x \log p_\theta(x)||^2\right) + C$$

$$= \mathbb{E}_{x\sim p_N(x)}\left(2\nabla_x s_\theta(x) + \frac{1}{2}||s_\theta(x)||^2\right) + C$$

And hence

$$\boxed{\theta^* = \arg\min_\theta \mathbb{E}_{x\sim p_N(x)}\left[2\nabla_x s_\theta(x) + \frac{1}{2}||s_\theta(x)||^2\right]}$$

DSM (1)

Traditional score matching is uneasy & costly to compute, so another approach is introduced: add noise.

Let $x \sim p_N(x)$, we 'smooth' this sample to a distribution: $q(\tilde{x}|x) \sim \mathcal{N}(\tilde{x}; x, \sigma^2 \mathrm{I})$. It can be seen as $\tilde{x} = x + \epsilon$, where ϵ is a noise. Assume we have a $\tilde{x}$, we still need to let model learn 'where to go', that is to trace along the gradient field for log-likelihood, i.e., the score function.

We know by marginalization that

$$p(\tilde{x}) = \int p_N(x) q(\tilde{x}|x) \mathrm{d}x = \mathbb{E}_{x \sim p_N(x)}[q(\tilde{x}|x)]$$

Hence:

DSM (2)

$$
\begin{aligned}
s(\tilde{x}) &= \nabla_{\tilde{x}} \log p(\tilde{x}) = \frac{1}{p(\tilde{x})} \nabla_{\tilde{x}} p(\tilde{x}) = \frac{1}{p(\tilde{x})} \nabla_{\tilde{x}} \int p_N(x) q(\tilde{x}|x) \mathrm{d}x \\
&= \frac{1}{p(\tilde{x})} \int p_N(x) \nabla_{\tilde{x}} q(\tilde{x}|x) \mathrm{d}x = \int \frac{p_N(x) q(\tilde{x}|x)}{p(\tilde{x})} \nabla_{\tilde{x}} \log q(\tilde{x}|x) \mathrm{d}x \\
&= \int p(x|\tilde{x}) \nabla_{\tilde{x}} \log q(\tilde{x}|x) \mathrm{d}x = \mathbb{E}_{x \sim p(x|\tilde{x})} [\nabla_{\tilde{x}} \log q(\tilde{x}|x)]
\end{aligned}
$$

Since q is a normal distribution, $q(\tilde{x}|x) = k \exp\left(\frac{||\tilde{x} - x||^2}{-2\sigma^2}\right)$, so $\nabla_{\tilde{x}} \log q(\tilde{x}|x) = \frac{x - \tilde{x}}{\sigma^2}$. Therefore,

$$
\boxed{s(\tilde{x}) = \frac{1}{\sigma^2} \mathbb{E}_{x \sim p(x|\tilde{x})} [x - \tilde{x}]}
$$

DSM (3)

From their, we can then by linearity of expectation to get

$$\mathbb{E}_{x \sim p(x|\tilde{x})}[x] = \tilde{x} + \sigma^2 \nabla_{\tilde{x}} \log p(\tilde{x})$$

which shows that the score function is the direction of denoising.

For the LHS, we cannot directly get, but we can use a network f_θ: $s(\tilde{x}) = \dfrac{f_\theta(\tilde{x}) - \tilde{x}}{\sigma^2}$ to learn it, using simple MSE:

$$\theta^* = \arg\min_\theta \mathbb{E}_{x \sim p_N(x), \tilde{x} \sim q(\tilde{x}|x)}[||f_\theta(\tilde{x}) - x||^2]$$

For sampling process (x_T to x_0), still use Langevin Dynamics:

$$x_{t-1} = x_t + \frac{\epsilon}{2\sigma^2}[f_\theta(x_t) - x_t] + \sqrt{\epsilon}\eta_t$$

DSM (4)

From DSM to DDPM: multi-step noise adding as a Markov process (which means there is a new parameter t to control for time frame)

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t \mathrm{I})$$

which can lead to the following closed form simply by adding multiple normal distributions:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathrm{I})$$

which can also be written as

$$\boxed{x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon}$$

Therefore,

$$\nabla_{x_t} \log q(x_t|x_0) = -\frac{1}{1-\bar{\alpha}_t}(x_t - \sqrt{\bar{\alpha}_t}x_0) = \frac{1}{\sqrt{1-\bar{\alpha}_t}}\epsilon$$

DSM (5)

We can use a function (network) parametrized by θ, s_θ, to learn for the score of the noise sample. Back to the original score fisher-divergence loss:

$$\begin{aligned}
\theta^* &= \arg\min_\theta \mathbb{E}_{x_0 \sim q(x_0), x_t \sim q(x_t|x_0)}[||s_\theta(x_t, t) - s(x_t, t)||^2] \\
&= \arg\min_\theta \mathbb{E}_{x_0 \sim q(x_0), x_t \sim q(x_t|x_0)}[||s_\theta(x_t, t) + \frac{1}{\sqrt{1-\bar{\alpha}_t}}\epsilon||^2] \\
&= \arg\min_\theta \mathbb{E}_{x_0 \sim q(x_0), x_t \sim q(x_t|x_0)}[|| - \frac{1}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t) + \frac{1}{\sqrt{1-\bar{\alpha}_t}}\epsilon||^2]
\end{aligned}$$

$$= \arg\min_\theta \mathbb{E}[\epsilon - \epsilon_\theta(x_t, t)]$$

Which shows that it is equivalent to construct a network for predicting noise!

What's more?

If you think this is not enough, you can explore more on how differential equations work with generative models. In fact, diffusion process is closely related to Stochastic Differential Equations:

The forward process (add noise) is

$$\mathrm{d}x = -\frac{1}{2}\beta_t x \mathrm{d}t + \sqrt{\beta_t}\mathrm{d}W$$

where we call the coefficient to $\mathrm{d}t$ the drift term $f(x, t)$, and $\mathrm{d}W$ is a Brownian motion in $\mathbb{R}^d$, where its coefficient is the diffusion term $g(x, t)$. Its corresponding reverse process (denoise) can be computed using Kologorov equations, and the result looks like

$$\mathrm{d}x = [-\frac{1}{2}\beta_t x - \beta_t \nabla_x \log p(x)]\mathrm{d}t + \sqrt{\beta_t}\mathrm{d}\tilde{W}$$