Denoising Diffusion Probabilistic Models

August 9, 2022

Useful papers - [HJA20],

1 Introduction

This works is dedicated to generative modelling with *diffusion process*. As always our goal is to recover distribution of real data $p_{data}(x)$. For these purpose, authors proposed an approach based on two processes: *forward* and *reverse*. In the forward process we destroy our data step by step adding noise. We do a lot of small steps. In physics this calls *diffusion* (i'm not sure). In the reverse process we aim to obtain samples from noise, that is, we are going in the opposite direction. It is something like normalizing flows, but with significant differences: 1) we add noise to data (as we know adding noise in generative modelling is extremely necessary because of manifold hypothesis); 2) transformations can be arbitrary (as oppose to normalizing flows where we have to use invertible transformations). Lets discuss in detail both processes.

2 Forward process

As was said in forward (diffusion) process we add noise to data for several steps. This is pretty easy one, we do not need any neural networks to do it. Lets describe it more formally. Let $q(x_0)$ is data distribution. Diffusion process is a Markov Chain, thus we have:

- Sequence of random variables: $x_0, x_1, ..., x_T$, where $x_0 \sim q(x_0), x_T \sim$ noise.
- Proposal distribution: $q(x_t|x_{t-1}) = \mathcal{N}(x_t|\sqrt{1-\beta_t}x_{t-1}, \beta_t I)$. This distribution is necessary for transformation from variable x_{t-1} to x_t. So, this is core of Markov Chain. Here β_t could be constant (that is, not to depend on time), or gradually increase from small one to 1. Proposal distribution is really important because it gives us the form of all x, i.e.:

$$x_{t+1} = \sqrt{1-\beta_{t+1}}x_t + \sqrt{\beta_{t+1}}\epsilon, \epsilon \sim \mathcal{N}(\epsilon|0, I), x_0 \sim q(x_0) \tag{1}$$

- Joint distribution: $q(x_0, ..., x_T) = q(x_0)\prod_{t=1}^{T} q(x_t|x_{t-1})$. This is a markovity property, i.e., probability of next random variable depends only from previous random variable.

This three parts defines our diffusion process.

Lets see on some cool and necessary stuff. In future we will need $q(x_t|x_0)$. As we showed, $q(x_t|x_{t-1}) = \mathcal{N}(x_t|\sqrt{1-\beta_t}x_{t-1}, \beta_t I)$. So, it is logical to assume that $q(x_t|x_0)$ is also normal. Thus, we need to obtain its parameters. Lets do some easy tricks.

$$\begin{aligned} x_{t+1} &= \sqrt{1-\beta_{t+1}}x_t + \sqrt{\beta_{t+1}}\epsilon \\ &= \sqrt{1-\beta_{t+1}}\left(\sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon\right) + \sqrt{\beta_{t+1}}\epsilon \\ &= \sqrt{1-\beta_{t+1}}\sqrt{1-\beta_t}x_{t-1} + \sqrt{1-\beta_{t+1}}\sqrt{\beta_t}\epsilon + \sqrt{\beta_{t+1}}\epsilon \end{aligned} \tag{2}$$

Then,

$$\begin{aligned} \mathbb{E}\, x_{t+1} &= \sqrt{1-\beta_{t+1}}\sqrt{1-\beta_t}x_{t-1} \\ \mathbb{D}x_{t+1} &= (1-\beta_{t+1})(1-\beta_t) + \beta_{t+1} \\ &= \beta_t + \beta_{t+1}(1-\beta_t) \\ &= -\left(-\beta_t - \beta_{t+1}(1-\beta_t) + 1\right) + 1 \\ &= 1 - (1-\beta_t)(1-\beta_{t+1}) \end{aligned} \tag{3}$$

So, we can say that

$$\begin{aligned} q(x_{t+1}|x_{t-1}) &= \mathcal{N}(x_{t+1}|\sqrt{1-\beta_{t+1}}\sqrt{1-\beta_t}x_{t-1}, 1-(1-\beta_t)(1-\beta_{t+1})) \\ q(x_{t+1}|x_0) &= \mathcal{N}(x_{t+1}|\prod_{s=1}^{t+1}\sqrt{1-\beta_s}x_0, 1-\prod_{s=1}^{t+1}(1-\beta_s)) \end{aligned} \tag{4}$$

Let $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$, then $q(x_{t+1}|x_0) = \mathcal{N}(x_{t+1}|\sqrt{\bar{\alpha}_{t+1}}x_0, (1-\bar{\alpha}_{t+1})I)$. It is possible to show that, $\lim_{t\to\infty}\bar{\alpha}_t = 0$. But then, if T is large:

$$q(x_T|x_0) \approx \mathcal{N}(x_T|0, I) \tag{5}$$

3 Reverse process

In reverse process we again have Markov Chain. But this process is more complicated because we need to recover data from noise. So, to solve such a difficult task our efforts are not enough we need neural networks. So, again we have

- Sequence of random variables: $x_T, x_{T-1}, ..., x_0$, where $x_0 \sim p_\theta(x_0), x_T \sim$ noise. Here we are going back through time.
- Proposal distribution: $p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}|\mu_\theta(x_t, t), \sigma_t^2 I)$. Here $\mu_\theta(x_t, t)$ is neural network. In the papers authors fixed variance. Similar to the previous one, we can write:

$$x_{t-1} = \mu_\theta(x_t, t) + \sigma_t\epsilon, \epsilon \sim \mathcal{N}(\epsilon|0, I), x_T \sim \mathcal{N}(x_T|0, I) \tag{6}$$

- Joint distribution: $p_\theta(x_T, ..., x_0) = p_\theta(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$.

Knowing of proposal distribution gives us the ability to obtain real looked samples from noise.

4 Training

Now we have to discuss the training phase of the diffusion model. As always we want to maximize log likelihood

$$\theta^* = \underset{\theta}{\operatorname{argmax}}\, \mathbb{E}_{q(x_0)} \log p_\theta(x_0) \tag{7}$$

We can do the same trick like we did in VAE with ELBO:

$$\mathbb{E}_{p_{data}(x)} \log p_\theta(x) \geq \mathbb{E}_{p_{data}(x)} \int q(z|x) \log \frac{p_\theta(x,z)}{q(z|x)} dz \tag{8}$$

We can say that x_0 is observed variable and $x_1, ..., x_T$ is latent variables, i.e. $z = (x_1, ..., x_T)$. Then ELBO for our case:

$$\begin{gathered}
\mathbb{E}_{q(x_0)} \log p_\theta(x_0) \\
\geq \mathbb{E}_{q(x_0)} \int q(x_1, ..., x_T|x_0) \log \frac{p_\theta(x_0, ..., x_T)}{q(x_1, ..., x_T|x_0)} dx_1...dx_T \\
= \int q(x_0)dx_0 \int q(x_1, ..., x_T|x_0) \log \frac{p_\theta(x_0, ..., x_T)}{q(x_1, ..., x_T|x_0)} dx_1...dx_T \\
= \mathbb{E}_q(x_0, ..., x_T) \log \frac{p_\theta(x_0, ..., x_T)}{q(x_1, ..., x_T|x_0)} \\
= \mathbb{E}_q(x_0, ..., x_T) \log \frac{p_\theta(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)}{\prod_{t=1}^{T} q(x_t|x_{t-1}))} \\
= \mathbb{E}_q(x_0, ..., x_T) \left[\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})} + \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \right] \equiv
\end{gathered} \tag{9}$$

Here we can see that we have something like KL-divergence, but times are reversed. We can use Bayes rule, but we will meet following problem:

$$q(x_t|x_{t-1}) = \frac{q(x_{t-1}|x_t)q(x_t)}{q(x_{t-1})} \tag{10}$$

We do not know $q(x_t)$ and $q(x_{t-1})$. To avoid this lets use Markov property:

$$\begin{gathered}
\equiv \mathbb{E}_q(x_0, ..., x_T) \left[\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1}, x_0)} + \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \right] \\
= \mathbb{E}_q(x_0, ..., x_T) \left[\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} \frac{q(x_{t-1}|x_0)}{q(x_t|x_0)} + \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \right] \\
= \mathbb{E}_q(x_0, ..., x_T) \left[\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} + \sum_{t=2}^{T} \log \frac{q(x_{t-1}|x_0)}{q(x_t|x_0)} + \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \right] \\
= \mathbb{E}_q(x_0, ..., x_T) \left[\log p_\theta(x_T) + \sum_{t=2}^{T} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} + \log \frac{q(x_1|x_0)}{q(x_T|x_0)} + \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \right]
\end{gathered} \tag{11}$$

$$= \mathbb{E}_q(x_0, ..., x_T) \left[\log \frac{p_\theta(x_T)}{q(x_T|x_0)} + \sum_{t=2}^{T} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} + \log p_\theta(x_0|x_1) \right]$$

$$= -\mathbb{E}_{q(x_0,x_T)} \log \frac{q(x_T|x_0)}{p_\theta(x_T)} - \sum_{t=2}^{T} \mathbb{E}_{q(x_0,x_{t-1},x_t)} \log \frac{q(x_{t-1}|x_t, x_0)}{p_\theta(x_{t-1}|x_t)} + \mathbb{E}_{q(x_0,x_1)} \log p_\theta(x_0|x_1) \equiv \tag{12}$$

Because of $q(x_0, x_{t-1}, x_t) = q(x_{t-1}|x_0, x_t)q(x_0, x_t)$

$$\equiv -\mathbb{E}_{q(x_0)} \mathbb{KL}(q(x_T|x_0)||p_\theta(x_T)) - \sum_{t=2}^{T} \mathbb{E}_{q(x_0,x_t)} \mathbb{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)) + \mathbb{E}_{q(x_0,x_1)} \log p_\theta(x_0|x_1)$$

$$= \mathbb{E}_q(x_0, ..., x_T) \left[-\mathbb{KL}(q(x_T|x_0)||p_\theta(x_T)) - \sum_{t=2}^{T} \mathbb{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)) + \log p_\theta(x_0|x_1) \right] \tag{13}$$

Looks cool. Lets calculate the second KL-divergence.

$$\begin{aligned} q(x_{t-1}|x_t, x_0) &= \frac{q(x_t|x_{t-1}, x_0)q(x_{t-1}|x_0)}{q(x_t|x_0)} \\ &= \frac{\mathcal{N}(x_t|\sqrt{1-\beta_t}x_{t-1}, \beta_t I)\mathcal{N}(x_{t-1}|\sqrt{\bar{\alpha}_{t-1}}x_0, (1-\bar{\alpha}_{t-1})I)}{\mathcal{N}(x_t|\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I)} \\ &= \mathcal{N}(x_{t-1}|\tilde{\mu}(x_t, x_0), \tilde{\beta}_t I) \\ \tilde{\mu}(x_t, x_0) &= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha}_t(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t, \tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \end{aligned} \tag{14}$$

Then, KL-divergence is

$$\begin{aligned} &\mathbb{KL}(\mathcal{N}(x_{t-1}|\tilde{\mu}(x_t, x_0), \tilde{\beta}_t I)||\mathcal{N}(x_{t-1}|\mu_\theta(x_t, t), \sigma_t^2 I)) \\ &= \int \mathcal{N}(x_{t-1}|\tilde{\mu}(x_t, x_0), \tilde{\beta}_t I) \log \frac{\mathcal{N}(x_{t-1}|\tilde{\mu}(x_t, x_0), \tilde{\beta}_t I)}{\mathcal{N}(x_{t-1}|\mu_\theta(x_t, t), \sigma_t^2 I)} dx_{t-1} \\ &= \int \mathcal{N}(x_{t-1}|\tilde{\mu}(x_t, x_0), \tilde{\beta}_t I) \log \frac{e^{||x_{t-1}-\tilde{\mu}(x_t,x_0)||^2}}{e^{||x_{t-1}-\mu_\theta(x_t,t)||^2}} + C \\ &= \int \mathcal{N}(x_{t-1}|\tilde{\mu}(x_t, x_0), \tilde{\beta}_t I)||\mu_\theta(x_t, t) - \tilde{\mu}(x_t, x_0)||^2 + C \\ &= \frac{1}{2\sigma_t^2}||\mu_\theta(x_t, t) - \tilde{\mu}(x_t, x_0)||^2 + C \end{aligned} \tag{15}$$

So, our final objective for train diffusion model (we will minimize this objective)

$$\boxed{\mathcal{L}_{DDPM}(\theta) = \sum_{t=2}^{T} \mathbb{E}_{q(x_0)q(x_t|x_0)} \left[\frac{1}{2\sigma_t^2}||\mu_\theta(x_t, t) - \tilde{\mu}(x_t, x_0)||^2 \right]} \tag{16}$$

What can we say here? We train diffusion model in such a way that reverse proposal distribution $p_\theta(x_{t-1}|x_t)$ equals to $q(x_{t-1}|x_t, x_0)$. However, it is not clear how we can interpreter the latter distribution. Because in forward process we define $q(x_{t-1}|x_t)$. I think that it is possible to say that $q(x_{t-1}|x_t, x_0)$ **is a true reverse proposal for samples from our dataset**. More precisely: we take some sample from data, x_0, apply forward process for him, i.e. $q(x_t|x_{t-1})$, and most important - **we build reverse process for that sample using Bayes rule**. Finally, we want to approximate this true reverse process for all samples x_0 using our model proposal distribution $p_\theta(x_{t-1}|x_t)$. A fair question: why we need some approximation if we have true reverse proposal distribution? Answer: we have this proposal only for samples from dataset and on inference stage we will not have condition on x_0. Informally, it is possible to say that in training phase we show how to transform from noise to data using samples from dataset. And on the inference, the model does it on its own without any x_0.

Okay, lets go more deeper.

References

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.