Diffusion models: From Theory to Experiments

Duc Q. Nguyen

URA Research Group
Ho Chi Minh University of Technology - VNU HCMC

March 04^{th} 2023

The fact that Science walks forward on two feet, namely theory and experiment...

Prof. Robert Millikan - Nobel Prize 1923

Table of Contents

1. Motivation
2. Probability background
3. Diffusion theory
4. Building diffusion model
5. Variants
6. Application example
7. Conclusion

Table of Contents

1 Motivation

2 Probability background

3 Diffusion theory

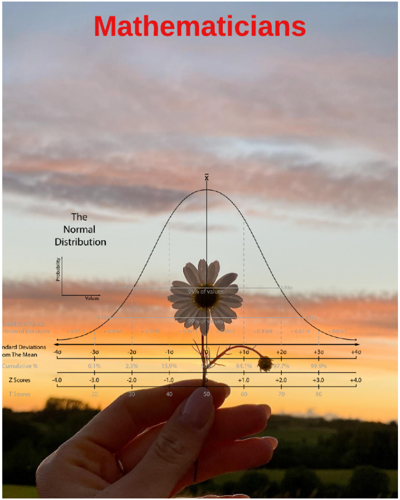
4 Building diffusion model

5 Variants

6 Application example

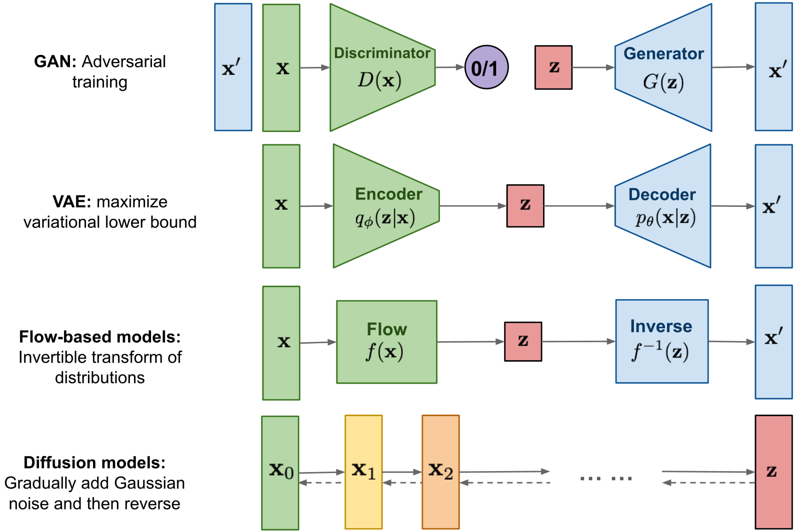
7 Conclusion

Flowers diffuse their fragrance with the wind



Flowers can generate fragrance theirself while data can't

Then, we might use somethings called "models" to perform generation.



Diffusion models can overcome previous limitations

- Implicit generative models: Generative Adversarial Networks. Due to adversarial training procedure. they can be unstable [1] and mode collapse [2].
- Likelihood-based models: Autoregressive models [3], Variational Auto-Encoders [4], Energy-Based Models [5], Normalizing Flow models [6]. These models require either surrogate objectives to optimize main objectives or strong restrictions on architecture.

Diffusion models [7, 8, 9, 10] are based on non-equilibrium thermodynamic basis. They do not require adversarial training nor surrogate objectives so that they overcome the limitations of previous models while allow very flexible design.

Table of Contents

1 Motivation

2 Probability background

3 Diffusion theory

4 Building diffusion model

5 Variants

6 Application example

7 Conclusion

Normal distribution

Normal distribution (a.k.a. Gaussian distribution) is a probability distribution on continous space. This distribution is commonly notated as

$$\mathcal{N}(\mu, \sigma^2),$$

where $\mu \in \mathbb{R}$ and $\sigma^2 \in \mathbb{R}_{>0}$ are the mean and variance, respectively.
The *Probability Density Function* (PDF) of Normal distribution is defined as

$$\phi(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

Merging two Normal distributions

Assuming that we have $x_1 \sim \mathcal{N}(\mu_1, \sigma_1^2)$ and $x_2 \sim \mathcal{N}(\mu_2, \sigma_2^2)$ and they are independent to each other. We need to merge these two distributions into one $\mathcal{N}(\bar{\mu}, \bar{\sigma}^2)$. Then, if we define the new $\bar{\mu}$ as

$$\bar{\mu} = \mathbb{E}[x] = \frac{n_1\mu_1 + n_2\mu_2}{n_1 + n_2},$$

the new variance can be calculated as

$$\bar{\sigma}^2 = \mathbb{E}[x^2] - \mathbb{E}[x]^2 = \frac{\sigma_1^2 n_1^2 + \sigma_2^2 n_2^2 + (n_1\mu_1 + n_2\mu_2)^2}{(n_1 + n_2)^2} - \bar{\mu}^2$$

Merging two Normal distributions

$$\mathbb{E}[x^2] = \mathbb{E}[(\frac{n_1 x_1 + n_2 x_2}{n_1 + n_2})^2] \quad (1)$$

$$= \frac{1}{(n_1 + n_2)^2}\mathbb{E}[n_1^2 x_1^2 + n_2^2 x_2^2 + 2n_1 n_2 x_1 x_2] \quad (2)$$

$$= \frac{1}{(n_1 + n_2)^2}(n_1^2\mathbb{E}[x_1^2] + n_2^2\mathbb{E}[x_2^2] + 2n_1 n_2 \mathbb{E}[x_1]\mathbb{E}[x_2]) \quad (3)$$

$$= \frac{(\sigma_1^2 + \mu_1^2)n_1^2 + (\sigma_2^2 + \mu_2^2)n_2^2 + 2n_1 n_2 \mu_1 \mu_2}{(n_1 + n_2)^2} \quad (4)$$

$$= \frac{\sigma_1^2 n_1^2 + \sigma_2^2 n_2^2 + (n_1\mu_1 + n_2\mu_2)^2}{(n_1 + n_2)^2} \quad (5)$$

Kullback–Leibler divergence

The *Kullback–Leibler divergence* (KL divergence), also known as relative entropy, is a measure of how different two probability distributions are from each other. It is a non-symmetric measure, meaning that the KL divergence from P to Q is not necessarily the same as the KL divergence from Q to P. The KL divergence between two probability distributions P and Q over the same random variable X is defined as:

$$D_{KL}(P||Q) = \int_{-\infty}^{\infty} p(x) \log \left(\frac{p(x)}{q(x)} \right) dx$$

where $p(x)$ and $q(x)$ are the probability density functions of P and Q, respectively. The logarithm is usually taken to the base 2 or e.

Kullback–Leibler divergence

The KL divergence is a non-negative quantity and is equal to zero if and only if the two distributions P and Q are identical. It is not a true distance metric, as it violates the triangle inequality and is not symmetric.

The KL divergence has many applications in information theory, statistics, machine learning, and data science. For example, it is commonly used as a measure of dissimilarity between two probability distributions in clustering, classification, and model selection. It is also used in variational inference, where it is used to minimize the difference between a true posterior distribution and an approximating distribution.

Bayes' theorem

Bayes' theorem, which is also called conditional probability, decribes a way to calculate the probability of an event if we already know the occurence of related event(s).
Assumming that we have two event A and B. The occurence of event A is depended on the occurence of event B. Then

$$P(A|B) = \frac{P(AB)}{P(B)} = \frac{P(A)P(B|A)}{P(B)}$$

Naming the components

$$P(A|B) = \frac{P(AB)}{P(B)} = \frac{P(A)P(B|A)}{P(B)}$$

- $P(A|B)$: The *posterior* probability of A given B
- $P(B|A)$: The *likelihood* of A given a fixed B
- $P(A)$: The *prior* probability of A
- $P(B)$: The *marginal* probability

We can rewrite the Bayes' theorem as follows

$$P(A|B) \propto P(A)P(B|A),$$

which can be interpreted as

$$Posterior \propto Prior \times Likelihood.$$

The mystery of AI power

In real life, usually we do not know the prior distribution of an event (If we know, we can alter the future as well). Normally, we can assume the prior distribution, observe the likelihood and then calculate the posterior distribution.

Instead of performing real experiments, *Artificial Intelligence* (AI) models try to maximize the likelihood of observing data. When maximizing the likelihood of data, the calculated posterior become the real posterior.

In short, the power of AI is maxmize the likelihood of data to make the parameterized prior distribution close to the real posterior.

$$P_\theta(A) \to P(A|B)$$

Table of Contents

1. Motivation
2. Probability background
3. Diffusion theory
4. Building diffusion model
5. Variants
6. Application example
7. Conclusion

Notations

Firstly, let us define some symbols for convenience

- T: The total number of steps
- β_t: The step size of of step t
- $\mathbf{x}_t$: A data point at step t
- θ: Learnable parameters
- I: Identity matrix

In many usecases, the data point presented as above are the results of generation process. For example, with the problem of generating facial images with target attributes, the data points are the images with attributes..

Formally, a diffusion process is a process of adding noises to original data point until the data point is completely noised, then denoising it to get the original data point.
The process is featurized with a reversible Markov chain. With total step T, the Markov chain can be written as

$$Forward : q(\mathbf{x}_t|\mathbf{x}_{t-1}), t = \overline{1..T}$$

$$Reverse : q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0), t = \overline{1..T}$$

We define a Markov chain with Gaussian transitions as follows

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}),$$

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}).$$

where $\beta_t \in (0,1)$ and $\beta_{t-1} < \beta_t$.

For further convenience, we define $\alpha_t = 1 - \beta_t$, so that $\alpha_t \in (0, 1)$ and $\alpha_{t-1} > \alpha_t$. Then,

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_{t-1}, (1-\alpha_t)\mathbf{I}).$$

Because the forward process is just adding noises, we can speed up this process for better efficiency. At each step $t \in [1..T]$, we can sample x_t using the reparameterization trick. With $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, we have

$$\begin{aligned}
\mathbf{x}_t &= \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{\alpha_t(1-\alpha_{t-1})}\boldsymbol{\epsilon}_{t-2} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\boldsymbol{\epsilon}}_{t-2} \\
&= \dots \\
&= \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \text{ where } \bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i
\end{aligned}$$

Forward diffusion process

$$\sqrt{\alpha_t(1-\alpha_{t-1})}\boldsymbol{\epsilon}_{t-2} \sim \mathcal{N}(0, \alpha_t(1-\alpha_{t-1})I)$$

$$\sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \sim \mathcal{N}(0, (1-\alpha_t)I)$$

Then the merge of these two distributions is $\mathcal{N}(\bar{\mu}, \bar{\sigma}^2)$ as

$$\bar{\mu} = 1 * 0 + 1 * 0 = 0$$

$$\bar{\sigma}^2 = \alpha_t(1-\alpha_{t-1}) * 1^2 + (1-\alpha_t) * 1^2 + (1 * 0 + 1 * 0)^2 - 0^2 = 1 - \alpha_t\alpha_{t-1}.$$

Therefore, we have

$$\sqrt{\alpha_t(1-\alpha_{t-1})}\boldsymbol{\epsilon}_{t-2} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} = \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\boldsymbol{\epsilon}}_{t-2}$$

Forward diffusion process

By the above property, we can rewrite the posterior distribution of x_t as follows

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I})$$

After having the noised data point x_T after T steps, we might need to denoise it to the original one.

Reverse diffusion process

As we know, after T steps, we have a noised data point. If the β_t is small enough, the noises will have Gaussian distribution. Thus, we can write down the reverse process as

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t\mathbf{I})$$

$$q(\mathbf{x}_{T:0}|\mathbf{x}_0) = q(\mathbf{x}_T)\prod_{t=1}^{T} q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$$

Reverse diffusion process

Also, using the Bayes' theorem and PDF, we have

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$$

$$= q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)}$$

$$= \sqrt{\frac{1-\bar{\alpha}_t}{2\pi\beta_t(1-\bar{\alpha}_{t-1})}} \exp\Big(-\frac{1}{2}\big(\frac{(\mathbf{x}_t - \sqrt{\alpha_t}\mathbf{x}_{t-1})^2}{\beta_t} + \frac{(\mathbf{x}_{t-1} - \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0)^2}{1-\bar{\alpha}_{t-1}}$$

$$- \frac{(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0)^2}{1-\bar{\alpha}_t}\big)\Big)$$

$$= \sqrt{\frac{1-\bar{\alpha}_t}{2\pi\beta_t(1-\bar{\alpha}_{t-1})}} \exp\Big(-\frac{1}{2}\big(\frac{\mathbf{x}_t^2 - 2\sqrt{\alpha_t}\mathbf{x}_t\mathbf{x}_{t-1} + \alpha_t\mathbf{x}_{t-1}^2}{\beta_t}$$

$$+ \frac{\mathbf{x}_{t-1}^2 - 2\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0\mathbf{x}_{t-1} + \bar{\alpha}_{t-1}\mathbf{x}_0^2}{1-\bar{\alpha}_{t-1}} - \frac{(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0)^2}{1-\bar{\alpha}_t}\big)\Big)$$

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \sqrt{\frac{1-\bar{\alpha}_t}{2\pi\beta_t(1-\bar{\alpha}_{t-1})}} \exp\Big(-\frac{1}{2}\big(\big(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\big)\mathbf{x}_{t-1}^2 - \big(\frac{2\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0\big)\mathbf{x}_{t-1} + C(\mathbf{x}_t, \mathbf{x}_0)\big)\Big)$$

Easily, we can infer the $\tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0)$ and $\tilde{\boldsymbol{\beta}}_t$ as belows

$$\tilde{\beta}_t = 1/\big(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\big) = 1/\big(\frac{\alpha_t - \bar{\alpha}_t + \beta_t}{\beta_t(1-\bar{\alpha}_{t-1})}\big) = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \cdot \beta_t$$

$$\begin{aligned}\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) &= \big(\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0\big)/\big(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\big) \\ &= \big(\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0\big)\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \cdot \beta_t \\ &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0\end{aligned}$$

Reverse diffusion process

Recall that we already have $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$, so that we can observe

$$\begin{aligned}
\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t) \\
&= \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right)
\end{aligned}$$

In a nutshell

In conclusion, the diffusion process can be summarized as follows

$$Forward: q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$$

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$$

$$Reverse: q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_{t-1}; \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right), \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \cdot \beta_t\right)$$

$$q(\mathbf{x}_{T:0}|\mathbf{x}_0) = q(\mathbf{x}_T)\prod_{t=1}^{T} q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$$

Table of Contents

1. Motivation
2. Probability background
3. Diffusion theory
4. Building diffusion model
5. Variants
6. Application example
7. Conclusion

Optimizing the prior to posterior distribution

Notice that in real application, we do not have $\mathbf{x}_0$, we only have context to generate $\mathbf{x}_0$. Therefore, we need to parameterize the likelihood estimation $p_\theta(\mathbf{x}_0|\theta)$ by θ parameter. Recall that maximizing likelihood problem can be solved by minimizing log likelihood. Thus,

$$
\begin{aligned}
-\log p_\theta(\mathbf{x}_0|\theta) &\leq -\log p_\theta(\mathbf{x}_0|\theta) + D_{\mathsf{KL}}(q(\mathbf{x}_{1:T}|\mathbf{x}_0) \| p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0, \theta)) \\
&= -\log p_\theta(\mathbf{x}_0|\theta) + \mathbb{E}_{\mathbf{x}_{1:T} \sim q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \left[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T}|\theta)/p_\theta(\mathbf{x}_0|\theta)} \right] \\
&= -\log p_\theta(\mathbf{x}_0|\theta) + \mathbb{E}_q \left[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T}|\theta)} + \log p_\theta(\mathbf{x}_0|\theta) \right] \\
&= \mathbb{E}_q \left[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T}|\theta)} \right]
\end{aligned}
$$

$$
\text{Let } L_{\mathsf{VLB}} = \mathbb{E}_{q(\mathbf{x}_{0:T})} \left[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T}|\theta)} \right] \geq -\mathbb{E}_{q(\mathbf{x}_0)} \log p_\theta(\mathbf{x}_0|\theta)
$$

then our mission becomes minimizing L_{VLB}.

Optimizing the prior to posterior distribution

$$
\begin{aligned}
L_{\textsf{VLB}} &= \mathbb{E}_{q(\mathbf{x}_{0:T})}\Big[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T}|\theta)}\Big] \\
&= \mathbb{E}_q\Big[\log \frac{\prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_T|\theta)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t,\theta)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T|\theta) + \sum_{t=1}^{T}\log \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t,\theta)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T|\theta) + \sum_{t=2}^{T}\log \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t,\theta)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1,\theta)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T|\theta) + \sum_{t=2}^{T}\log \Big(\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t,\theta)} \cdot \frac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}\Big) \\
&+ \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1,\theta)}\Big]
\end{aligned}
$$

Optimizing the prior to posterior distribution

$$L_{\text{VLB}} = \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T|\theta) + \sum_{t=2}^{T} \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \theta)} + \sum_{t=2}^{T} \log \frac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}$$

$$+ \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1, \theta)}\Big]$$

$$= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T|\theta) + \sum_{t=2}^{T} \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \theta)} + \log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{q(\mathbf{x}_1|\mathbf{x}_0)}$$

$$+ \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1, \theta)}\Big]$$

$$= \mathbb{E}_q\Big[\log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{p_\theta(\mathbf{x}_T|\theta)} + \sum_{t=2}^{T} \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \theta)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1, \theta)\Big]$$

Optimizing the prior to posterior distribution

$$L_{\mathsf{VLB}} = \mathbb{E}_q[\underbrace{D_{\mathsf{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_T|\theta))}_{L_T}$$

$$+ \sum_{t=2}^{T} \underbrace{D_{\mathsf{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \theta))}_{L_{t-1}} \underbrace{- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1, \theta)}_{L_0}]$$

Easily, we can consider that L_T is a constant because $\mathbf{x}_T$ is a Gaussian noised data point. The L_0 can be ignored or modeled using a separated discrete decoder $\mathcal{N}(\mathbf{x}_0; \boldsymbol{\mu}_\theta(\mathbf{x}_1, 1), \boldsymbol{\Sigma}_\theta(\mathbf{x}_1, 1))$ [7]. In short, we need to minimize

$$L_t = D_{\mathsf{KL}}(q(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_t|\mathbf{x}_{t+1}, \theta)) \text{ for } 1 \leq t \leq T-1$$

Linking to the mystery of optimization, minimizing the above loss term is equivalent to making the parameterized prior distribution become the true posterior distribution.

Noise estimation

Assume that $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \theta) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\sigma}_t)$. We already have

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_{t-1}; \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right), \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \cdot \beta_t\right).$$

It is easy to observe that, we need a function to estimate $\boldsymbol{\epsilon}_t$ using $\mathbf{x}_t$ and t. Thus,

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \theta) = \mathcal{N}\left(\mathbf{x}_{t-1}; \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right), \boldsymbol{\sigma}_t\right)$$

Noise estimation

$$
\begin{aligned}
&L_t \\
&= D_{\mathsf{KL}}(q(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_t|\mathbf{x}_{t+1}, \theta)) \\
&= \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\Big[\log \frac{q(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{x}_0)}{p_\theta(\mathbf{x}_t|\mathbf{x}_{t+1}, \theta)}\Big] \\
&= \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\Big[\frac{1}{2\|\boldsymbol{\sigma}_t\|_2^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2\Big] \\
&= \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\Big[\frac{1}{2\|\boldsymbol{\sigma}_t\|_2^2}\|\frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t - \mathbf{x}_t + \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\Big)\|^2\Big] \\
&= \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\Big[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t)\|\boldsymbol{\sigma}_t\|_2^2}\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\|^2\Big] \\
&= \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\Big[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t)\|\boldsymbol{\sigma}_t\|_2^2}\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, t)\|^2\Big]
\end{aligned}
$$

Noise estimation

We can simplify the weighting term in the beginning of the loss because we have controlled the learning rate [7]. The simplified loss then becomes

$$
\begin{aligned}
L_t^{\text{simple}} &= \mathbb{E}_{t\sim[1,T],\mathbf{x}_0,\boldsymbol{\epsilon}_t}\left[\left\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right\|^2\right] \\
&= \mathbb{E}_{t\sim[1,T],\mathbf{x}_0,\boldsymbol{\epsilon}_t}\left[\left\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, t)\right\|^2\right]
\end{aligned}
$$

This loss can be interpreted as optimizing network(s) for predicting the right noise at each step in forward process. If the esimation(s) are correct, the reverse process can decode noises to very good data point.

Training and Sampling procedure

The below of training and sampling procedure belows is belonged to *Denoising Diffusion Probabilistic Models* (DDPM) [7].

Algorithm 1: Training procedure

Assign value for $\beta_t, t = \overline{1..T}$;
$\alpha_t = 1 - \beta_t, t = \overline{1..T}$;
$\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i, t = \overline{1..T}$;
repeat
  $\mathbf{x}_0 \sim q(\mathbf{x}_0)$;
  $t \sim$ Uniform$(\{1, \ldots, T\})$;
  $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$;
  Opimize θ by gradient descent on

$$\nabla_\theta = \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2$$

until *converged*;

Training and Sampling procedure

Algorithm 2: Sampling procedure

$\mathbf{x}_T \sim \mathcal{N}(0, I)$;

for $t = T, \ldots, 1$ **do**

  $\mathbf{z} \sim \mathcal{N}(0, I)$ if $t > 1$ else $\mathbf{z} = 0$;

  $\boldsymbol{\sigma}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \cdot \beta_t$;

  $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \boldsymbol{\sigma}_t\mathbf{z}$;

end

Table of Contents

1. Motivation
2. Probability background
3. Diffusion theory
4. Building diffusion model
5. Variants
6. Application example
7. Conclusion

Here, we present some highlighted variants of diffusion models. Belong the followings, there are many other variants which are specified for differnt problems.

- **Denoising Diffusion Probabilistic Models** [7]: This is the main concept from the begining.
- **Denoising Diffusion Implicit Model** [8]: This model is different from DDPM by making the sampling procedure deterministic.

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\Big) + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$$

- **Latent Diffusion Model** [9]: Performing diffusion on latent space by adding an Encoder and a Decoder.

Variants of diffusion models II

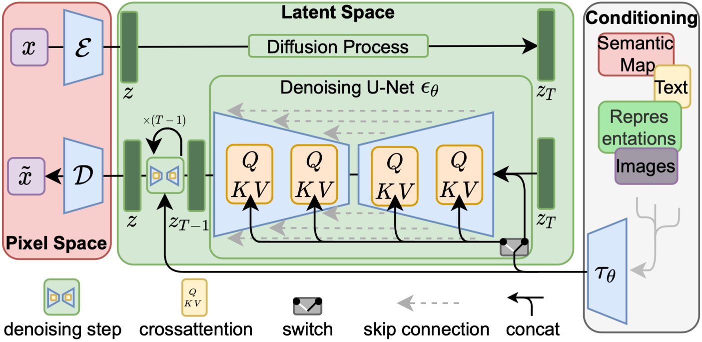


Figure: Latent Diffusion Model. Source [9]

Variants of diffusion models III

- **Score-based Diffusion Model** [10]: Score-based Diffusion Model learns to generate data points by maximizing a score function that measures the similarity between an input point and the training dataset. They use a deep neural network architecture that takes as input a noise vector and a data point, and outputs the score for that point.

- **Classifier Guided Diffusion:** The representative is *Ablated Diffusion Model* (ADM) [11]. This class of models using the gradient from an aditional classifier to guide the diffusion process. The noise estimation function of these models can be summarized as follows

$$\bar{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t) = \boldsymbol{\epsilon}_\theta(x_t, t) - \sqrt{1 - \bar{\alpha}_t}\, w \nabla_{\mathbf{x}_t} \log f_\phi(y|\mathbf{x}_t),$$

  where $f_\phi(y|\mathbf{x}_t, t)$ is a trainable classifier.

- **Classifier-Free Guidance:** Recent research [12] shows that without the explicit classifier, we still can control the generation process by carefully design the architecture of noise estimation network. Let c is the input context then $p_\theta(\mathbf{x}|\theta)$ unconditional denoising diffusion model

Contextual diffusion II

can be parameterized through a score estimator $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$ and the conditional model $p_\theta(\mathbf{x}|c, \theta)$ can be parameterized through $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, c)$.

$$
\begin{aligned}
\nabla_{\mathbf{x}_t} \log p(c|\mathbf{x}_t) &= \nabla_{\mathbf{x}_t} \log\left(\frac{p(\mathbf{x}_t|c)p(c)}{p(\mathbf{x}_t)}\right) \\
&= \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|c) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) \\
&= -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\Big(\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, c) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\Big) \\
\bar{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t, c) &= \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, c) - \sqrt{1-\bar{\alpha}_t}\, w\nabla_{\mathbf{x}_t} \log p(c|\mathbf{x}_t) \\
&= \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, c) + w\big(\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, c) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\big) \\
&= (w+1)\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t, c) - w\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)
\end{aligned}
$$

Table of Contents

1. Motivation
2. Probability background
3. Diffusion theory
4. Building diffusion model
5. Variants
6. Application example
7. Conclusion

A basic experiment from scratch

Problem: Given a string represented for a number, please generate the image(s) of that number.
Dataset: MNIST
Context: The string of a number (Maximum string length is 5 characters)
Generation target: Image(s) of a number
Models: DDPM and DDIM with classifier-free guidance
Link: Colab Notebook

Table of Contents

1. Motivation
2. Probability background
3. Diffusion theory
4. Building diffusion model
5. Variants
6. Application example
7. Conclusion

Takeaway

Advantages: Generative modeling faces a trade-off between tractability and flexibility. Tractable models are easy to evaluate and fit data inexpensively but cannot efficiently capture complex data structures. Conversely, flexible models can accommodate intricate data structures but are computationally expensive to evaluate, train, or sample from. Diffusion models offer the benefits of both tractability and flexibility by being both analytically tractable and capable of accommodating arbitrary data structures.

Disadvantages: Diffusion models rely on a lengthy Markov chain of diffusion steps to generate samples, which can result in significant time and compute costs. While newer techniques have been introduced to expedite this process, sampling from diffusion models is still slower than using GAN models.

– Thank you for your attention –

Acknowledgements:
Lilian Weng's blog [13] for detail formulas explainations

Contact me if you have any questions
nqduc@hcmut.edu.vn

References I

T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems*, vol. 29, Curran Associates, Inc., 2016.

L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, "Unrolled generative adversarial networks," in *International Conference on Learning Representations*, 2017.

B. Uria, M.-A. Côté, K. Gregor, I. Murray, and H. Larochelle, "Neural autoregressive distribution estimation," *J. Mach. Learn. Res.*, vol. 17, p. 7184–7220, jan 2016.

D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2014.

References II

M. Arbel, L. Zhou, and A. Gretton, "Generalized energy based models," in *International Conference on Learning Representations*, 2021.

L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real NVP," in *International Conference on Learning Representations*, 2017.

J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, (Red Hook, NY, USA), Curran Associates Inc., 2020.

J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *International Conference on Learning Representations*, 2021.

R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

References III

Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, (Red Hook, NY, USA), Curran Associates Inc., 2019.

P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," in *Advances in Neural Information Processing Systems* (M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), vol. 34, pp. 8780–8794, Curran Associates, Inc., 2021.

J. Ho and T. Salimans, "Classifier-free diffusion guidance," in *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

L. Weng, "What are diffusion models?," *lilianweng.github.io*, Jul 2021.