Derivation of Training Loss in Diffusion Model

Hiroaki Kubo · Following

7 min read · Nov 26, 2024

83

As you all know, diffusion model has had a tremendous impact in various areas in recent years. I wanted to learn more about the technology and therefore decided to build it myself from scratch. This article is primarily about the derivation of loss for the diffusion model, but if you want to see my article on the overall implementation, click **here**.

First of all, **I wanted to know about training loss**, but it is quite complicated and some parts were omitted in the paper, so I tried to research and derive it in my own way, and I have summarized it in this article. Basically, it follows the paper **Denoising Diffusion Probabilistic Models**.

. . .

First, difussion model has **forward process** and the **reverse process.** In the forward process, noise is added to the input data step by step. In the reverse process, the reverse of the forward process is performed to recover the original image from the noisy data. The graph below is easy to understand these processes.

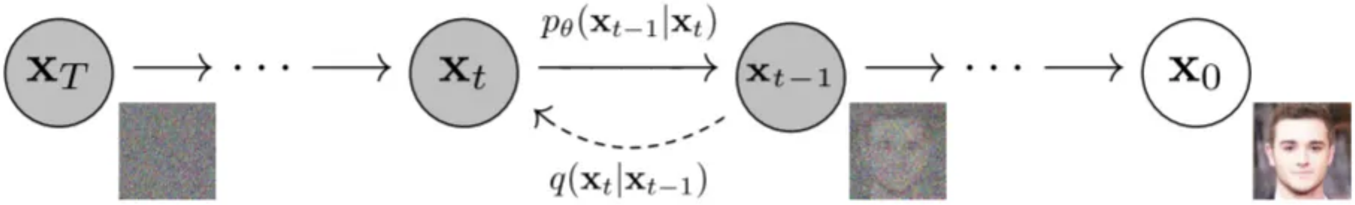
$\mathbf{x}_T$
$\mathbf{x}_t$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$\mathbf{x}_{t-1}$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$
$\mathbf{x}_0$

Figure 2: The directed graphical model considered in this work.

Diffusion model are latent variable models the form $p\theta(x0):=\int p\theta(x0:T)dx1:T$. The joint distribution $p\theta(x0:T)$ is called the reverse process, and it is defined as a Markov chain with learned Gaussian transitions starting at $p(xT)=\mathrm{N}(xT;0,I)$:

$$p_\theta(x_{0:T}) := p(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t), \quad p_\theta(x_{t-1} \mid x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \qquad (1)$$

Forward process is fixed to a Markov chain that gradually adds Gaussian noise to the data according to a variance schedule $\beta 1,\ldots,\beta T$:

$$q(x_{1:T}|x_0) := \prod_{t=1}^{T} q(x_t|x_{t-1}), \quad q(x_t|x_{t-1}) := \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \qquad (2)$$

The probability the generative model assigns to the data is as follows.

$$p_\theta(x_0) = \int dx_{1:T} p_\theta(x_{0:T}) \qquad (3)$$

In the original paper, the integral is intractable, so the fomula transformation is shown as follows.

Although not described in detail in the paper, I personally think that the formula transformation was performed using the forward process, which has a known probability distribution, probably because the probability distribution of the reverse process can be complicated and it is difficult to calculate the integral.

$$\begin{aligned} p_\theta(x_0) &= \int dx_{1:T} p_\theta(x_{0:T}) \frac{q(x_{1:T}|x_0)}{q(x_{1:T}|x_0)} \\ &= \int dx_{1:T} q(x_{1:T}|x_0) \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)} \\ &= \int dx_{1:T} q(x_{1:T}|x_0) p_\theta(x_T) \prod_{t=1}^{T} \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})} \qquad (4) \end{aligned}$$

Training is performed by optimizing the usual variational bound on negative log likelihood. Below equation has a upper bound provided by **Jensen's inequality**,

$$\begin{aligned} E[-\log p_\theta(x_0)] &= E\Big[-\log\Big[\int dx_{1:T}\, q(x_{1:T}|x_0)\, p(x_T)\prod_{t=1}^{T}\frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\Big]\Big] \\ &\leq E_q\Big[-\log\Big[p(x_T)\prod_{t=1}^{T}\frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\Big]\Big] \\ &\leq E_q\Big[-\log p(x_T) - \sum_{t\geq 1}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\Big] =: L \qquad (5) \end{aligned}$$

This equation can be further transformed as follows.

$$\begin{aligned} L &= E_q\Big[-\log p(x_T) - \sum_{t\geq 1}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\Big] \\ &= E_q\Big[-\log p(x_T) - \sum_{t>1}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})} - \log\frac{p_\theta(x_0|x_1)}{q(x_1|x_0)}\Big] \\ &= E_q\Big[-\log p(x_T) - \sum_{t>1}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t,x_0)}\frac{q(x_{t-1}|x_0)}{q(x_t|x_0)} - \log\frac{p_\theta(x_0|x_1)}{q(x_1|x_0)}\Big] \\ &= E_q\Big[-\log\frac{p(x_T)}{q(x_T|x_0)} - \sum_{t>1}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t,x_0)} - \log p_\theta(x_0|x_1)\Big] \\ &= E_q\Big[D_{KL}(q(x_T|x_0)\parallel p(x_T)) + \sum_{t>1}D_{KL}(q(x_{t-1}|x_t,x_0)\parallel p_\theta(x_{t-1}|x_t)) - \log p_\theta(x_0|x_1)\Big] \\ &= E_q\Big[L_T + \sum_{t>1}L_{t-1} + L_0\Big] \qquad (6) \end{aligned}$$

In the above equation deformation, $q(xt|xt-1)$ was transformed as follows.

$$\begin{aligned} q(x_t|x_{t-1}) &= q(x_t|x_{t-1}, x_0) \\ &= \frac{q(x_t, x_{t-1}|x_0)}{q(x_{t-1}|x_0)} \\ &= q(x_{t-1}|x_t, x_0)\frac{q(x_t|x_0)}{q(x_{t-1}|x_0)} \qquad (7) \end{aligned}$$

In the course of the above equation transformation, the following relationship is used.

$$q(x_t, x_{t-1}|x_0) = q(x_t|x_{t-1}, x_0) \cdot q(x_{t-1}|x_0) \qquad (8)$$

I also used the following equation transformation.

$$\begin{aligned} \sum_{t>1} \frac{q(x_{t-1}|x_0)}{q(x_t|x_0)}\frac{1}{q(x_1|x_0)} &= \frac{q(x_{T-1}|x_0)...q(x_1|x_0)}{q(x_T|x_0)...q(x_2|x_0)}\frac{1}{q(x_1|x_0)} \\ &= \frac{1}{q(x_T|x_0)} \qquad (9) \end{aligned}$$

DKL is called **KL Divergence** and is a type of statistical distance: a measure of how one reference probability distribution *P* is different from a second probability distribution *Q*.

$$\begin{aligned} KL(p \parallel q) &= \int p(x)\log(q(x))dx - \Big(- \int p(x)\log p(x)dx \Big) \\ &= - \int p(x)\log\frac{q(x)}{p(x)}dx \qquad (10) \end{aligned}$$

We will now simplify Eq(6).

. . .

Forward process and LT

We ignore the fact that the forward process variances βt are learnable by reparameterization and instead fix them to constants. Thus, in our implementation, the approximate posterior q has no learnable parameters, **so LT is a constant during training and can be ignored.**

. . .

Reverse process and L1:T−1

Now we discuss our choices in $p\theta(xt-1|xt)=N(xt-1:\mu\theta(xt,t),\Sigma\theta(xt,t))$ for $1<t\leq T$. First, we set $\Sigma\theta(xt,t)=\sigma t2I$ to untrained time dependent constants. Experimentally, both $\sigma t2=\beta t$ and $\sigma t2=\beta t\sim$ had similar results. Therefore, we can write $Lt-1$ as follows.

$$L_{t-1} = E_q\left[\frac{1}{2\sigma_t^2}|\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t)|^2\right] + C \qquad (11)$$

C is a constant that does not depend on θ. So, we see that the most straightforward parameterization of $\mu\theta$ is a model that predict $\mu t\sim$, the forward process posterior mean. Thus, we can use the following equation to expand the above equation. That transformation is called as the **reparameterization trick**.

$$x_t(x_0, \epsilon) = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \quad (\epsilon \sim \mathcal{N}(0, I)) \qquad (12)$$

We can get *x*0 from above equation.

$$x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t(x_0, \epsilon) - \sqrt{1-\bar{\alpha}_t}\epsilon) \qquad (13)$$

By using the above *x*0, we can update *Lt*−1.

$$\begin{aligned} L_{t-1} - C &= E_{x_0,\epsilon}\Big[\frac{1}{2\sigma_t^2}\Big|\tilde{\mu_t}\Big(x_t(x_0,\epsilon), \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t(x_0,\epsilon) - \sqrt{1-\bar{\alpha}_t}\epsilon)\Big) - \mu_\theta(x_t(x_0,\epsilon),t)\Big|^2\Big] \\ &= E_{x_0,\epsilon}\Big[\frac{1}{2\sigma_t^2}\Big|\frac{1}{\sqrt{\alpha_t}}\Big(x_t(x_0,\epsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\Big) - \mu_\theta(x_t(x_0,\epsilon),t)\Big|^2\Big] \qquad (14) \end{aligned}$$

We use below ut¯(xt,x0)*ut*¯(*xt*,*x*0) for above transformation. **Appendix** explains the derivation of *ut*¯(*xt*,*x*0) and *βt*¯.

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu_t}(x_t, x_0), \tilde{\beta}_t I),$$
$$where \quad \tilde{\mu_t}(x_t, x_0) := \frac{\sqrt{\alpha_{t-\bar{1}}\beta_t}}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\alpha_{\overline{t-1}})}{1-\bar{\alpha}_t}x_t \quad and \quad \tilde{\beta}_t := \frac{1-\alpha_{\overline{t-1}}}{1-\bar{\alpha}_t}\beta_t \qquad (15)$$

Since *xt* is available as input to the model, we may choose the parameterization

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\Big(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\Big) \qquad (16)$$

where $\epsilon\theta$ is a function approximator intended to predict ϵ from xt.

We can simplify the equation of $Lt–1–C$ by using the above equation.

$$\begin{aligned}
L_{t-1} - C &= E_{x_0,\epsilon}\Big[\frac{1}{2\sigma_t^2}\Big|\frac{1}{\sqrt{\alpha_t}}\Big(x_t(x_0,\epsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\Big) - \mu_\theta(x_t(x_0,\epsilon),t)\Big|^2\Big] \\
&= E_{x_0,\epsilon}\Big[\frac{1}{2\sigma_t^2}\Big|\frac{1}{\sqrt{\alpha_t}}\Big(x_t(x_0,\epsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\Big) - \frac{1}{\sqrt{\alpha_t}}\Big(x_t(x_0,\epsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t,t)\Big)\Big|^2\Big] \\
&= E_{x_0,\epsilon}\Big[\frac{\beta_t}{2\sigma_t^2\alpha_t(1-\alpha_{\bar{t-1}})}\Big|\epsilon - \epsilon_\theta(x_t,t)\Big|^2\Big] \\
&= E_{x_0,\epsilon}\Big[\frac{\beta_t}{2\sigma_t^2\alpha_t(1-\alpha_{\bar{t-1}})}\Big|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\Big|^2\Big] \qquad (17)
\end{aligned}$$

To summarize, **we can train the reverse process mean function approximator $\mu\theta$ to predict μt~, or by modifying its parameterization, we can train it to predict ϵ.**

. . .

Data scaling, reverse process decoder, and L0

We assume that image data consists of integers in 0,1,…,255 scaled linearly to [−1,1]. This ensures that the neural network reverse process operates on consistently scaled inputs starting from the standard normal prior *p*(*xT*). To obtain discrete log likelihoods, we set the last term of the reverse process to an independent discrete decoder derived from the Gaussian N(*x*0;*μθ*(*x*1,1),*σ*12*I*):

$$p_\theta(x_0|x_1) = \prod_{i=1}^{D} \int_{\delta_{-(x_0^i)}}^{\delta_{(x_0^i)}} \mathcal{N}(x_0; \mu_\theta^i(x1,1), \sigma_1^2)dx$$

$$\delta_+(x) = \begin{cases} \infty & (x = 1) \\ x + \frac{1}{255} & (x < 1) \end{cases} \qquad \delta_-(x) = \begin{cases} -\infty & (x = -1) \\ x - \frac{1}{255} & (x > -1) \end{cases} \tag{18}$$

where *D* is the data dimensionality and the *i* superscript indicates extraction of one coordinate. The above equation calculates the simulataneous probability of each pixel. *δ* means clipping bounds that

to each discrete value of *x*0*i*. **This ensures proper handling of discrete data in a continous framework.**

. . .

Simplified training objective

From Eq(17) and Eq(18), we can simplify training objective more.

$$L_{simple} := E_{t,x_0,\epsilon}\left[|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha_t}}x_0 + \sqrt{1-\bar{\alpha_t}}\epsilon, t)|^2\right] \qquad (19)$$

The *t*=1 case corresponds to *L*0 wit the integral in the discrete decoder difinition Eq(18) approximated by the Gaussian probability density function times the bin width, ignoring *σ*12 and edge effects. The *t*>1 cases correspond to an unweighted version of Eq(17). The details are described in the paper, but this simple formula is used because the accuracy is better when the weight portion of Eq(17) is removed.

. . .

Appendix: Derivation of mean and variance of q(xt−1|xt,x0)

The conditional distribution *q*(*xt*−1|*xt*,*x*0) is proportional to the product of the following two distributions.

$$q(x_{t-1}|x_t, x_0) \propto q(x_t|x_{t-1})q(x_{t-1}|x_0)$$

Those two distributions are defined as follows.

$$\begin{aligned} q(x_t|x_{t-1}) &= \mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, \beta_t I) \\ q(x_{t-1}|x_0) &= \mathcal{N}(x_{t-1}; \sqrt{\alpha_{\bar{t}-1}}x_0, (1-\alpha_{\bar{t}-1})I) \end{aligned}$$

When calculating variance, we use the product property of the Gaussian distribution. The inverse of the variance in a product of Gaussian distribution is expressed as the sum of the inverse variances of the individual distributions.

$$\begin{aligned}\frac{1}{\tilde{\beta}_t} &= \frac{1}{\beta_t} + \frac{1}{1-\bar{\alpha_{t-1}}} \\ &= \frac{1-\bar{\alpha_{t-1}}+\beta_t}{\beta_t(1-\bar{\alpha_{t-1}})}\end{aligned}$$

If we inverse both sides to obtain the variance βt~

$$\tilde{\beta}_t = \frac{\beta_t(1-\bar{\alpha_{t-1}})}{1-\bar{\alpha_{t-1}}+\beta_t}$$

Here, we use the following property.

$$1-\bar{\alpha}_t = (1-\bar{\alpha_{t-1}})+\beta_t$$

Substitute this into the variance fomula to transforme it.

$$\tilde{\beta}_t = \frac{(1-\bar{\alpha_{t-1}})}{1-\bar{\alpha}_t}\beta_t$$

Next, we consider the derivation of the mean. Letting $m1$ and $m2$ be the mean of each distribution and $\sigma 12$ and $\sigma 22$ be the variance of each distribution, the mean can be calculated as follows.

$$\tilde{\mu}_t = \frac{m_1\sigma_2^2 + m_2\sigma_1^2}{\sigma_1^2 + \sigma_2^2}$$

Therefore, we can calculate the mean as follows.

$$\begin{aligned}
\tilde{\mu}_t &= \frac{(\sqrt{\alpha_t}x_{t-1})(1-\tilde{\alpha_{t-1}}) + (\sqrt{\bar{\alpha_{t-1}}}x_0)\beta_t}{\beta_t + (1-\tilde{\alpha_{t-1}})} \\
&= \frac{\sqrt{\alpha_t}(1-\tilde{\alpha_{t-1}})}{\beta_t + (1-\tilde{\alpha_{t-1}})}x_{t-1} + \frac{\sqrt{\tilde{\alpha_{t-1}}}\beta_t}{\beta_t + (1-\tilde{\alpha_{t-1}})}x_0 \\
&= \frac{\sqrt{\alpha_{t-1}}\beta_t}{1-\tilde{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\tilde{\alpha_{t-1}})}{1-\tilde{\alpha}_t}x_t
\end{aligned}$$

Here, we use the following property again.

$$1-\bar{\alpha}_t = (1-\bar{\alpha_{t-1}}) + \beta_t$$

Honestly, I’m not sure if this derivation is correct. I also don’t know why x_{t-1} can be converted to x_t. This may be related to the fact that we first approximate $q(x_{t-1}|x_t,x_0)$ with two probability distributions.

. . .