The Maths Behind Denoising Diffusion Probabilistic Models

Giovani Tavares
giovanitavares@outlook.com

University of Sao Paulo — January 10, 2025

Motivation

I've been studying and using generative AI models for a while now. I have always put at least some effort in understanding Large Language Model's (LLMs) architecture and training process, but state-of-the-art (SOTA) image generative models have always been kept as a future topic to be grasped. I say that the personal main reason for this is that I hadn't had the opportunity to use this models professionally until recently, so I had little interest in dedicating some time for them.

By the last months of 2024, I was invited by my boss to give a worshop on Generative Adversarial Networks (GANs) and Denoising Diffusion Probabilistic Models (DDPMs) during the Third School on Data Science and Machine Learning that took place in the International Centre for Theoretical Physics (ICTP) in the São Paulo State University (UNESP, Sao Paulo, Brazil) in December 6th, 2024. The workshop was my greatest academic challenge when it comes to presenting. The public was made of high performing graduate and postgraduate students from the best universities in South America.

As mentioned, DDPMs was one of the topics I had to give a workshop on. In order to do so, I had to study its architecture and training procedure. The concepts I learned are all presented here.

1 What are DDPMs?

Denoising Diffusion Probabilistic Models (DDPMs) are models capable of predicting *noise* from a noisy input. By using such prediction, a sampling algorithm can be used to remove the noise from the input which results in a denoised output. DDPMs were first introduced in 2015 by Sohl-Dickstein and other USA researchers in [2], but were made popular by Jonathan Ho and others in [1]. The latter paper used such model to generate images outputs from pure random noise, which is basically an image generation model.

DDPMs are made of two processes: a **forward** and a **reversion** process. The former is responsible for gradually adding noise to a image by sampling from a normal distribution according to a Markov Chain. The latter removes added noise by sampling from another normal distribution. In simple terms, the training of DDPMs involve learning the reversion process' distribution's parameters.

1.1 Forward Process

The process of adding noise to an input image (x_0) is a Markov Chain that generates a noisier image x_t from a less noisy image x_{t-1}. In [1] and here, an x's subscript represents its position in the Markov Chain so that the greater an x's subscript the noisier it is. Hence, x_t represents the result of adding noise to x_{t-1} by transitioning it once in the Markov Chain.

From the original DDPM paper, we know that in the forward process, a sample x_t is produced by adding noise to a sample x_{t-1} according to a normal distribution defined below:

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

Using $q(x_t|x_{t-1})$ is not very feasible from an implementation point of view, because it makes the production of each noisy sample x_t dependent on x_{t-1}, which makes it necessary to make t transitions starting from x_0 in order to get to the x_t sample. **Ideally, one would need a single transition from x_0 to get to x_t, with $t > 0$.**

The authors of [1] achieves such ideal scenario by defining a cumulative noise α_t presented below:

Definition 1.1 (Cumulative Noise)**.** *The Cumulative Noise (α_t) added to an input x_0 up to the t-th step is defined as:*

$$\alpha_t := 1 - \beta_t \tag{2}$$

$$\bar{\alpha_t} := \prod_{s=1}^{t} a_s \tag{3}$$

which leads to (4)

$$q(x_t|x_0) := \mathcal{N}(x_t; \sqrt{\bar{\alpha_t}} \times x_0; (1 - \bar{\alpha_t})I) \tag{5}$$

Definition 1.1 permits the sampling of x_t to be done in a single transition from x_0.

According to the original DDPM paper, the **Variance Schedule** $\beta_1, ..., \beta_T$ sequence that defines the noisy images distribution can be learned or held constant as hyperparameters. The authors uses the latter approach, which makes the forward process' variances constant.

1.2 Reverse Process

As previously mentioned, the reverse process is responsible for removing the noise from an input. Hence, this process is responsible for generating $\mathbf{x_0}$ given a noisy input $\mathbf{x_t}$. It is a *Markov Chain* with learned Gaussian transitions defined below:

Definition 1.2 (Reverse Process Transitions)**.** *The Reverse Process is a Markov Chain with the following transitions:*

$$p(x_T) = \mathcal{N}(x_T; 0; 1) \tag{6}$$

$$p_\theta(x_{0:T}) := p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t) \tag{7}$$

$$p_\theta(x_{t-1}|x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t); \Sigma_\theta(x_t, t)) \tag{8}$$

Notice that the definition above is simply a Markov Chain with a Standard Gaussian initial state and Gaussian transition distributions parametrized by μ_θ and Σ_θ. We want to create a model that is capable of recovering the best x_0 possible given a set of observed variables $Z := \mathbf{x_{1:T}}$. Theoretically, this can be achieved if we have a process to sample from the following distribution.

Definition 1.3 (Reverse Process Prior)**.** *Ideally, the Reverse Process will let us sample from:*

$$p(\mathbf{x_0}) = \int p(\mathbf{x_0}, \mathbf{x_{1:T}}) \mathbf{d_{1:T}} \tag{9}$$

$$\tag{10}$$

We see that the computation of $p(\mathbf{x_0})$ is highly complex and intractable due to its multidimensionality, which makes the training based on optimizating $p(\mathbf{x_0})$ directly infeasible. To solve this problem, the author's of the DDPM paper used the *Variational Lower Bound* of the expectation log-likelihood function, also called evidence lower bound (ELBO).

1.2.1 Variational Lower Bound / Evidence Lower Bound / ELBO

The Variational Lower Bound is a tight lower bound that limits $log(p(\mathbf{x_0}))$ from below. Hence, when $p(\mathbf{x_0})$ is intractable as in the case of DDPMs, one can always maximize such lower bound as a means to ensure that $log(p(\mathbf{x_0}))$ is as large as possible. Such lower bound is often called **ELBO** or simply L and will be demonstrated using two different approaches: the **Jensen's Inequality** and the **KL Divergence**. The Variational Lower Bound is a tight lower bound that limits $log(p(\mathbf{x_0}))$ from below. Hence, when $p(\mathbf{x_0})$ is intractable as in the case of DDPMs, one can always maximize such lower bound as a means to ensure that $log(p(\mathbf{x_0}))$ is as large as possible. Such lower bound is often called **ELBO** or simply L and will be demonstrated using two different approaches: the **Jensen's Inequality** and the **KL Divergence**.

Definition 1.4 (Variational Lower Bound - Jensen's Inequality)**.** *Let's use the Rule Of Total Probability to find an upper bound for the log-likelihood function.*

$$log[p_\theta(\mathbf{x_0})] = log \int_{\mathbf{x_{1:T}}} p(\mathbf{x_0}, \mathbf{x_{1:T}}) d\mathbf{x_{1:T}} \tag{11}$$

$$log[p_\theta(\mathbf{x_0})] = log \int_{\mathbf{x_{1:T}}} p(\mathbf{x_0}, \mathbf{x_{1:T}}) \frac{q(\mathbf{x_{1:T}}|\mathbf{x_0})}{q(\mathbf{x_{1:T}}|\mathbf{x_0})} d\mathbf{x_{1:T}} \tag{12}$$

$$log[p_\theta(\mathbf{x_0})] = log(\mathbb{E}_q \Big[\frac{p(\mathbf{x_{0:T}})}{q(\mathbf{x_{1:T}}|\mathbf{x_0})} \Big]) \tag{13}$$

the Jensen's inequality tells us (14)

$$f(\mathbf{E}(\mathbf{X})) \geq \mathbf{E}(f(\mathbf{X}))) \tag{15}$$

for any concave function f. (16)

log is concave, hence: (17)

$$log[p_\theta(\mathbf{x_0})] \geq \mathbb{E}_q \Big[log \frac{p(\mathbf{x_{0:T}})}{q(\mathbf{x_{1:T}}|\mathbf{x_0})} \Big] \tag{18}$$

If we define the Variational Lower Bound L as: (19)

$$L := \mathbb{E}_q \Big[log \frac{p(\mathbf{x_{0:T}})}{q(\mathbf{x_{1:T}}|\mathbf{x_0})} \Big] \tag{20}$$

$$\implies -log[p_\theta(\mathbf{x_0})] \leq -L \tag{21}$$

Definition 1.5 (Variational Lower Bound - KL Divergence)**.** *In order to reverse the forward process, we need that the forward process' distribution $q(\mathbf{x_{1:T}}|\mathbf{x_0})$ is as close to $p(\mathbf{x_{1:T}}|\mathbf{x_0})$ as possible. We can use the Kullback-Leibler (KL) divergence between q and p ($\mathbf{D_{KL}}$) to evaluate their difference as find a bound to $log[p_\theta(\mathbf{x}_0)]$.*

$$\mathbf{D_{KL}}\big[q(\mathbf{x_{1:T}}|\mathbf{x_0})||p(\mathbf{x_{1:T}}|\mathbf{x_0})\big] := \mathbb{E}_q\big[log(q(\mathbf{x_{1:T}}|\mathbf{x_0}) - log(p(\mathbf{x_{1:T}}|\mathbf{x_0}))\big] \tag{22}$$

using the Bayes' Rule we can write (23)

$$p(\mathbf{x_{1:T}}|\mathbf{x_0}) = \frac{p(\mathbf{x_{1:T}}, \mathbf{x_0})}{p(\mathbf{x_0})} \tag{24}$$

$$\implies \mathbf{D_{KL}}\big[q(\mathbf{x_{1:T}}|\mathbf{x_0})||p(\mathbf{x_{1:T}}|\mathbf{x_0})\big] = \mathbb{E}_q\big[log(q(\mathbf{x_{1:T}}|\mathbf{x_0}) - log(p(\mathbf{x_{1:T}}|\mathbf{x_0})) + log(p(\mathbf{x_0}))\big] \tag{25}$$

the prior of the latent variables does not depend on q (26)

$$\mathbf{D_{KL}}\big[q(\mathbf{x_{1:T}}|\mathbf{x_0})||p(\mathbf{x_{1:T}}|\mathbf{x_0})\big] = \mathbb{E}_q\big[log(q(\mathbf{x_{1:T}}|\mathbf{x_0}) - log(p(\mathbf{x_{1:T}}, \mathbf{x_0}))\big] + log(p(\mathbf{x_0})) \tag{27}$$

$$\implies log(p(\mathbf{x_0})) = \mathbf{D_{KL}}\big[q(\mathbf{x_{1:T}}|\mathbf{x_0})||p(\mathbf{x_{1:T}}|\mathbf{x_0})\big] - \mathbb{E}_q\big[log(q(\mathbf{x_{1:T}}|\mathbf{x_0}) - log(p(\mathbf{x_{1:T}}, \mathbf{x_0}))\big] \tag{28}$$

$$log(p(\mathbf{x_0})) = \mathbf{D_{KL}}\big[q(\mathbf{x_{1:T}}|\mathbf{x_0})||p(\mathbf{x_{1:T}}|\mathbf{x_0})\big] + \mathbb{E}_q\big[log(p(\mathbf{x_{1:T}}, \mathbf{x_0}) - log(q(\mathbf{x_{1:T}}|\mathbf{x_0}))\big] \tag{29}$$

but $\mathbf{D_{KL}}\big[q(\mathbf{x_{1:T}}|\mathbf{x_0})||p(\mathbf{x_{1:T}}|\mathbf{x_0})\big] \geq 0$ (30)

$$\implies -log(p(\mathbf{x_0})) \leq \mathbb{E}_q\big[- log \frac{p(\mathbf{x_{0:T}})}{q(\mathbf{x_{1:T}}|\mathbf{x_0})}\big] \tag{31}$$

If we define the Variational Lower Bound L as: (32)

$$L := \mathbb{E}_q \Big[log \frac{p(\mathbf{x_{0:T}})}{q(\mathbf{x_{1:T}}|\mathbf{x_0})} \Big] \tag{33}$$

$$\implies -log[p(\mathbf{x_0})] \leq -L \tag{34}$$

We have found lower bound L for the log-likelihood function that is tractable if rewritten properly. Our goal is to have a process to sample from p such that $logp(\mathbf{x_0})$ is as large as possible. This means that our process will output very likely outputs $\mathbf{x_0}$. In order to do so, DDPMs work in two steps: **noise prediction** and **sampling**. The former is the one responsible for predicting the a noise ϵ_θ of an input $\mathbf{x_t}$ which has has been sampled from the forward process' distribution q. The latter uses such prediction to sample $\mathbf{x_0}$ from p.

1.2.2 Noise Predictor Training Derivation

Having ELBO (L) as a lower bound for the log likelihood function means we can train our noise predictor model to maximize L, i.e., L is a candidate for our model's loss function. In order to do so, further algebraic manipulation must be performed with it in order to make it tractable in a way that the noise that has been added to the input appears somewhere. We demonstrate such manipulations here and will begin by showing that maximing L basically means approximating the distributions $p(\mathbf{x_{t-1}}|\mathbf{x_t})$ and $q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0})$.

Definition 1.6 (Noise Predictor's Loss Derivation)**.** *In order to build the Noise Predictor's loss function, we need to remember that both forward and reverse processes are Markov Chains and use this fact to manipulate L.*

$$L = \mathbb{E}_q\left[log\frac{p(\mathbf{x_{0:T}})}{q(\mathbf{x_{1:T}}|\mathbf{x_0})}\right] \tag{35}$$

The forward and reverse processes are Markov Chains, so (36)

$$\mathbb{E}_q\left[log\frac{p(\mathbf{x_{0:T}})}{q(\mathbf{x_{1:T}}|\mathbf{x_0})}\right] = \mathbb{E}_q\left[log\frac{p(\mathbf{x_T})\prod_{t=1}^{T}p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})}{q(\mathbf{x_1}|\mathbf{x_0})\prod_{t=2}^{T}q(\mathbf{x_t}|\mathbf{x_{t-1}}, \mathbf{x_0})}\right] \tag{37}$$

$$= \mathbb{E}_q\left[log\frac{p(\mathbf{x_T})\prod_{t=1}^{T}p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})}{q(\mathbf{x_T}|\mathbf{x_0})\prod_{t=2}^{T}q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0})}\right] \tag{38}$$

$$= \mathbb{E}_q\left[log\frac{p(\mathbf{x_T})p_\theta(\mathbf{x_0}|\mathbf{x_1})\prod_{t=2}^{T}p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})}{q(\mathbf{x_T}|\mathbf{x_0})\prod_{t=2}^{T}q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0})}\right] \tag{39}$$

$$= \mathbb{E}_q\left[log\frac{p(\mathbf{x_T})}{q(\mathbf{x_T}|\mathbf{x_0})}\right] + logp_\theta(\mathbf{x_0}|\mathbf{x_1}) + \sum_{t=2}^{T}\mathbb{E}_q\left[log\frac{p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})}{q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0})}\right] \tag{40}$$

$$= \mathbb{E}_q\left[log\frac{p(\mathbf{x_T})}{q(\mathbf{x_T}|\mathbf{x_0})}\right] + logp_\theta(\mathbf{x_0}|\mathbf{x_1}) + \sum_{t=2}^{T}\mathbf{D_{KL}}\left[p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})||q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0})\right] \tag{41}$$

If we pay attention to equation 41's terms above, we see that the first term is parameter free, because $p((x_T))$ is fixed and defined as a Gaussian, while $q((x_T|x_0))$ is also Gaussian from the definition of the forward process. Hence, we are left with the second and third terms.

As previously mentioned, we are interested in maximizing L. Using the equation 41, we see that doing so is equivalent to minimizing the KL Divergence between $p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})$ and $q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0})$. We know that both distributions are Gaussians, which makes computing the KL Divergence between them easier if we know their mean and variance. We will begin by calculating such moments for $q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0})$.

$$q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0}) = \frac{q(\mathbf{x_t}|\mathbf{x_{t-1}}, \mathbf{x_0})q(\mathbf{x_{t-1}}|\mathbf{x_0})}{q(\mathbf{x_t}|\mathbf{x_0})} \tag{42}$$

we know the q distribution from Definition 1.1, hence (43)

$q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0})$ is a product of known Gaussians over another known Gaussian that lets us define (44)

$$\mu_q(\mathbf{x_t}, \mathbf{x_0}) = \frac{(1-\bar{\alpha}_{t-1})\sqrt{\alpha_t}\mathbf{x_t} + (1-\alpha_t)\sqrt{\bar{\alpha}_{t-1}}\mathbf{x_0}}{(1-\bar{\alpha}_t)} \tag{45}$$

$$\Sigma_q(t) = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{I} \tag{46}$$

$$\implies q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0}) = \mathcal{N}(\mathbf{x_{t-1}}; \mu_q(\mathbf{x_t}, \mathbf{x_0}); \Sigma_q(t)) \tag{47}$$

We have just defined the $q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0})$ distribution. Now let's move on to the $p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})$ distribution.

$$p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t}) = \mathcal{N}(\mathbf{x_{t-1}}; \mu_\theta(\mathbf{x_t}); \Sigma_\theta(t)) \tag{48}$$

the reverse process variance is defined as the ground truth variance of the forward process: (49)

$$\Sigma_\theta(t) = \Sigma_q(t) \tag{50}$$

we are only left with the distribution's mean μ_θ (51)

$$p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t}) = \mathcal{N}(\mathbf{x_{t-1}}; \mu_\theta(\mathbf{x_t}); \Sigma_q(t)) \tag{52}$$

Equation 50 makes it much easier to calculate $\mathbf{D_{KL}}\big[p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})||q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0})\big]$: now we know we are trying to compute the KL Divergence between two Gaussians with the exact same variance. For that, there is the following result that arives from the definition of such divergence:

$$d_1(x) = \mathcal{N}(\mu_1, \sigma^2) \tag{53}$$

$$d_2(x) = \mathcal{N}(\mu_2, \sigma^2) \tag{54}$$

The KL divergence $D_{KL}(d_1|d_2)$ is given by: (55)

$$D_{KL}(d_1|d_2) = \frac{(\mu_1 - \mu_2)^2}{2\sigma^2} \tag{56}$$

Hence, (57)

$$\mathbf{D_{KL}}\big[p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})||q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0})\big] = \mathbf{D_{KL}}\big(\mathcal{N}(\mathbf{x_{t-1}}; \mu_\theta(\mathbf{x_t}); \Sigma_q(t)), \mathcal{N}(\mathbf{x_{t-1}}; \mu_q(\mathbf{x_t}, \mathbf{x_0}); \Sigma_q(t))\big) \tag{58}$$

$$= \frac{1 - \bar{\alpha}_t}{2(1 - \alpha_t)(1 - \bar{\alpha}_{t-1})}\big|\big|(\mu_\theta - \mu_q)_2^2\big|\big| \tag{59}$$

As our goal is to minimize the KL Divergence, from equation 59 we see that such goal comes down to basically minimizing the difference $\mu_\theta - \mu_q$, i.e., **we just need to minimize the difference between the means of the reverse and forward processes' distributions**. We need to define the reverse process' distribution mean prediction by taking a look at the forward process' one.

$$\mu_q(\mathbf{x_t}, \mathbf{x_0}) = \frac{(1 - \bar{\alpha}_{t-1})\sqrt{\alpha_t}\mathbf{x_t} + (1 - \alpha_t)\sqrt{\bar{\alpha}_{t-1}}\mathbf{x_0}}{(1 - \bar{\alpha}_t)} \tag{60}$$

we can define the prediction (61)

$$\hat{\mu}_q(\mathbf{x_t}, \mathbf{x_0}) = \mu_\theta(\mathbf{x_t}) \tag{62}$$

$$= \frac{(1 - \bar{\alpha}_{t-1})\sqrt{\alpha_t}\mathbf{x_t} + (1 - \alpha_t)\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_\theta}{(1 - \bar{\alpha}_t)} \tag{63}$$

In equation 63 we see that we are using our reverse process model's prediction $\mathbf{x}_\theta$ in the prediction of its distribution's mean, which let's us rewrite $\big|\big|(\mu_\theta - \mu_q)_2^2\big|\big|$ in terms of $\mathbf{x_0}$ and $\mathbf{x}_\theta$ which leves us with the following for the KL Divergence:

$$\mathbf{D_{KL}}\big(\mathcal{N}(\mathbf{x_{t-1}}; \mu_\theta(\mathbf{x_t}); \Sigma_q(t)), \mathcal{N}(\mathbf{x_{t-1}}; \mu_q(\mathbf{x_t}, \mathbf{x_0}); \Sigma_q(t))\big) = \frac{(1 - \bar{\alpha}_t)(\bar{\alpha}_{t-1})}{2(1 - \alpha_t)(1 - \bar{\alpha}_{t-1})}\big|\big|(\mathbf{x}_\theta - \mathbf{x_0})_2^2\big|\big| \tag{64}$$

The author's of the DDPM paper mention that equation 64 can be used as the loss function to train the reverse process model. On the other hand, we now that the forward process actually predict the noise that was added to an input $\mathbf{x_t}$ intead of predicting $\mathbf{x}_\theta$ directly. This means that the loss function must account for the error prediction somehow. This is achieved by further analysing $\mathbf{x_0}$ and $\mathbf{x}_\theta$ and remembering how $\mathbf{x}_\theta$ was defined in the forward process.

$$q(x_t|x_0) := \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t} \times x_0; (1-\bar{\alpha}_t)\mathbb{I}) \tag{65}$$

which let's us write (66)

$$\mathbf{x_t} = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \tag{67}$$

$$\implies \mathbf{x_0} = \frac{\mathbf{x_t} - \sqrt{1-\bar{\alpha}_t}\epsilon}{\sqrt{\bar{\alpha}_t}} \tag{68}$$

for a Standard Gaussian Noise ϵ. (69)

We can now define our prediction $\hat{\epsilon} = \epsilon_\theta$ (70)

$$\mathbf{x}_\theta = \frac{\mathbf{x_t} - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta}{\sqrt{\bar{\alpha}_t}} \tag{71}$$

$$\implies \frac{(1-\bar{\alpha}_t)(\bar{\alpha}_{t-1})}{2(1-\alpha_t)(1-\bar{\alpha}_{t-1})}\big|\big|(\mathbf{x}_\theta - \mathbf{x_0})_2^2\big|\big| = \frac{(1-\bar{\alpha}_t)(\bar{\alpha}_{t-1})}{2(1-\alpha_t)(1-\bar{\alpha}_{t-1})}\frac{(1-\alpha_t)^2}{(1-\bar{\alpha}_t)\alpha_t}\big|\big|(\epsilon_\theta - \epsilon)_2^2\big|\big| \tag{72}$$

Even though equation 72 could be used directly as the loss function for the noise predictor, the DDPM paper authors mention that optimizing $\big|\big|(\epsilon_\theta - \epsilon)_2^2\big|\big|$ without the scaling factor with the cumulative noise α_t is enough. Hence, we have finally defined a function to be minimized for the noise predictor training and hence write its algorithm.

Algorithm 1: Noise Predictor Training

1: **repeat**
2: $\mathbf{x_0} \sim \mathbf{q}(\mathbf{x_0})$ ▷ Sample image from training set
3: $\mathbf{x_0} \sim \mathbf{Uniform}(\{\mathbf{1}, \ldots, \mathbf{T}\})$ ▷ Sample the step of the Forward Process Markov Chain
4: $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ ▷ Sample standard gaussian noise to be added to the input
5: $\mathbf{x_t} = \sqrt{\bar{\alpha}_\mathbf{t}}\mathbf{x_0} + \sqrt{\mathbf{1}-\bar{\alpha}_\mathbf{t}}\epsilon$ ▷ Forward Process/ Generating Noisy Image
6: Take Gradient Descent Step on $\nabla_\theta(||\epsilon - \epsilon_\theta(\mathbf{x_t}, \mathbf{t})||)$
7: **until** converged

1.2.3 Sampling Algorithm Derivation

Now that we have defined a way to predict an input image $\mathbf{x_t}$'s noise, we need a way to use such prediction to reconstruct the original de-noised image $\mathbf{x_0}$, i.e., we need a way to sample from $p(\mathbf{x_0})$. To do so, let's recall how we have defined the $p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})$ distribution.

$$p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t}) = \mathcal{N}(\mathbf{x_{t-1}}; \mu_\theta(\mathbf{x_t}); \Sigma_q(t)) \tag{73}$$

$$\mu_\theta(\mathbf{x_t}) = \frac{(1-\bar{\alpha}_{t-1})\sqrt{\alpha_t}\mathbf{x_t} + (1-\alpha_t)\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_\theta}{(1-\bar{\alpha}_t)} \tag{74}$$

$$\mathbf{x}_\theta = \frac{\mathbf{x_t} - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta}{\sqrt{\bar{\alpha}_t}} \tag{75}$$

$$\implies \mu_\theta(\mathbf{x_t}) = \frac{\mathbf{x_t}}{\sqrt{\alpha_t}} - \frac{(1-\alpha_t)(\sqrt{1-\bar{\alpha}_t})}{(1-\bar{\alpha}_t)(\sqrt{\alpha_t})}\epsilon_\theta = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x_t} - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta\right) \tag{76}$$

$$\Sigma_\theta(t) = \Sigma_q(t) = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{I} \tag{77}$$

We now have defined $\mathbf{x_{t-1}}$'s mean and variance given $\mathbf{x_t}$ which let's us write it as (78)

$$\mathbf{x_{t-1}} = \mu_\theta(\mathbf{x_t}) + \sqrt{\Sigma_\theta(t)}\mathbf{z} \tag{79}$$

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{80}$$

We have now a way to generate $\mathbf{x_{t-1}}$ from $\mathbf{x_t}$. This means that if we have $\mathbf{x_1}$ we can generate $\mathbf{x_0}$. This let's us finally define our sampling algorithm:

Algorithm 2: Sampling

repeat
$\mathbf{x_T} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ ▷ Sample random noisy image
$\mathbf{T} \sim \mathbf{Uniform}(\{\mathbf{1}, \ldots, \mathbf{1000}\})$ ▷ Sample random length of the Denoising Chain. Max chain size of 1000 was set arbitrarily
for $\mathbf{t} = \mathbf{T}, \ldots, \mathbf{1}$ **do**
$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$ else $\mathbf{z} = \mathbf{0}$
$\mathbf{x_{t-1}} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x_t} - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x_t}, \mathbf{t})\right) + \sqrt{\Sigma_\theta(t)}\mathbf{z}$ ▷ Sampling $\mathbf{x_{t-1}}$ from $p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})$
end for
return $\mathbf{x_0}$

1.3 Recap

We have studied both the forward and reverse process that make up DDPMs. We have seen that good samples (or images) are generated by maximing the reverse process' likelihood $logp_\theta(\mathbf{x_0})$ lower bound, the Evidence Lower Bound, ELBO, or simply L.

We have rewritten ELBO in a way that its maximization turns out to be dual with minimizing the Kullback-Leibler (KL) divergence between $p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})$ and $q(\mathbf{x_{t-1}}|\mathbf{x_t}, \mathbf{x_0})$. Such KL-Divergence minimization was then translated to a noise predictor training. After the predictor training algorithm was defined, we have shown how to use such prediction to sample from $p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t})$ and finally reconstructing $\mathbf{x_0}$, the original denoised image.

References

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *CoRR*, abs/2006.11239, 2020.

[2] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *CoRR*, abs/1503.03585, 2015.