Diffusion Models in Depth: From a Theoretical and a Practical Perspective

Gregory Igor Sedykh

June 28th 2024

UNIVERSITÉ
DE GENÈVE
FACULTY OF SCIENCE
Department of Informatics

Bachelor thesis for the degree of Bachelor of Science in Computer Science
Supervised by Prof. Stéphane Marchand-Maillet

Abstract

Contents

1 Introduction

Diffusion models have gained widespread popularity since 2020, as models such as DALL-E, Stable Diffusion and Midjourney have proven to be capable of generating high-quality images given a text prompt. Furthermore, OpenAI's recent announcement of Sora has shown that diffusion models have now also become highly capable of generating minute long high-definition videos from a text prompt. [1]

These models date back to 2015, where the idea of a diffusion model appeared, based on diffusion processes used in thermodynamics. [12]
Denoising Diffusion Probabilistic Models (DDPMs) were a development of the original diffusion probabilistic model introduced in 2015. [3]
Subsequently, OpenAI improved upon the original DDPMs which did not have ideal log likelihoods [3] while also using less forward passes and therefore speeding up the sampling process. [8]
The most recent progress done by OpenAI has allowed their diffusion models to obtain better metrics and better sample quality than Generative Adversarial Networks (GANs) which were previously considered the state-of-the-art in image generation. [2]

The fairly recent apparition of diffusion models means not only that there is still a lot to be discovered about them, but also that progress is being made rapidly.
The theory behind diffusion models was mainly founded when Ho et al. [3] introduced their DDPMs in 2020, but many improvements have been made upon their work since then.
To understand what these models are and how they work, it is crucial to understand how DDPMs were developed, what choices were made when developing them and why these choices were made, as well as why changes were made to the original model and how they were made.

This report aims to provide an overview of the theory behind diffusion models, as well as a practical guide on how to implement a simple diffusion model using PyTorch, in order to compare what the theory shows us and what the practical implementation gives us.

2 Denoising Diffusion Probabilistic Models

A **diffusion model** is a generative model that consists of two Markov chains, one forward and one reverse.
Given an input (e.g. an image), the forward process will destroy the information in the image by gradually adding Gaussian noise at each step of the process. [3]
The reverse process' objective is to "reverse" the forward process and learn how to do so, starting from the noisy image and step-by-step, estimating the noise that was added to the image at each step and removing it until the first step is reached, where we should obtain the original image. [3]

2.1 Forward Process

The forward process is a Markov process that starts from the original image x_0 and adds Gaussian noise during T steps which results in a more noisy image x_t. [3].
Each step $t \in [1, T]$ has a size $\beta_t \in \{\beta_1, ..., \beta_T\}$.
This process is defined by q which is a probability distribution that takes as input an image x_t and outputs its likelihood.

$$q : \mathbb{R}^D \to [0, 1] \tag{1}$$

where D is the data dimensionality (e.g. for a 64×64 RGB image, $D = 64 \times 64 \times 3$).
Pixels in the image x_t are assumed to be mutually independent since the covariance matrix $(\beta_t I)$ is diagonal.

Formally, we obtain the following Markov process:

$$q\left(x_1, ..., x_T | x_0\right) = \prod_{t=1}^{T} q\left(x_t | x_{t-1}\right) \tag{2}$$

where:

$$q\left(x_t | x_{t-1}\right) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t} x_{t-1}, \beta_t I\right) \tag{3}$$

Equation **??** gives us a single step forward: given the previous image x_{t-1}, x_t is obtained by sampling a D-dimensional Gaussian distribution with mean $\sqrt{1-\beta_t} x_{t-1}$ and variance $\beta_t I$.
Equation **??** gives us the full forward process from the original image x_0 to the final image x_T.

The reparametrisation trick says that for a univariate Gaussian distribution where $z \sim p\left(z|x\right) = \mathcal{N}\left(\mu, \sigma^2\right)$, a valid reparametrisation would be $z = \mu + \sigma\epsilon$ where $\epsilon \sim \mathcal{N}\left(0, 1\right)$ is just noise. [4]
In this case, we can see that the image becomes more random because of the added Gaussian noise from ϵ_t at each step, since we are sampling:

$$\begin{aligned} x_t &= q(x_t | x_{t-1}) && (4) \\ &= \sqrt{1-\beta_t} x_{t-1} + \sqrt{\beta_t} I \epsilon_t \qquad \epsilon_t \sim \mathcal{N}_D\left(0, I\right) && (5) \end{aligned}$$

It is important to note that as $T \to \infty$ and with a correct choice of β_t, x_T will become a sample of an isotropic Gaussian distribution $(\mathcal{N}\left(0, I\right))$. [8] [12].
This is important for the reverse process, as it will allow us to take a sample $x_T \sim \mathcal{N}\left(0, I\right)$ and reverse

the forward process to obtain the original image x_0 (however this cannot be done so simply, as seen in section 3) [8].

Ho et al. [3] use the reparametrisation trick to be able to sample x_t at any arbitrary step t of the forward process.

Let $\alpha_t = 1 - \beta_t$, then:

$$x_t \sim \mathcal{N}\left(\sqrt{1-\beta_t}x_{t-1}, \beta_t I\right)$$
$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}I\epsilon_{t-1} \qquad \epsilon_{t-1} \sim \mathcal{N}_D(0, I)$$
$$x_t = \sqrt{\alpha_t}x_{t-1} + \sqrt{1-\alpha_t}\epsilon_{t-1}$$

From this, we can apply it again to x_{t-1} and obtain:

$$x_{t-1} = \sqrt{\alpha_{t-1}}x_{t-2} + \sqrt{1-\alpha_{t-1}}\epsilon_{t-2}$$

Therefore, we get:

$$x_t = \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \sqrt{\alpha_t(1-\alpha_{t-1})}\epsilon_{t-2} + \sqrt{1-\alpha_t}\epsilon_{t-1}$$

We can write:

$$x_t = \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \mathcal{N}(0, \alpha_t(1-\alpha_{t-1})I) + \sqrt{1-\alpha_t}\epsilon_{t-1} \tag{6}$$
$$= \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \mathcal{N}(0, \alpha_t(1-\alpha_{t-1})I) + \mathcal{N}(0, (1-\alpha_t)) \tag{7}$$

The convolution of the two Gaussian distributions $\mathcal{N}(\mu_1, \sigma_1^2)$ and $\mathcal{N}(\mu_2, \sigma_2^2)$ gives us a new Gaussian distribution with mean $\mu_1 + \mu_2$ and variance $\sigma_1^2 + \sigma_2^2$ (see Appendix A):

$$x_t = \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \mathcal{N}(0, (\alpha_t(1-\alpha_{t-1}) + 1 - \alpha_t)I) \tag{8}$$

Which finally gives us:

$$x_t = \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\epsilon}_{t-2}$$

Where $\epsilon_{t-1}, \epsilon_{t-2}, \ldots \sim \mathcal{N}(0, I)$ and $\bar{\epsilon}_{t-2}$ is the convolution of the noise distributions ϵ_{t-1} and ϵ_{t-2}.

We can recursively apply this backwards until x_0.
Let $\bar{\alpha}_t = \prod_{i=0}^{t} \alpha_i$:

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \tag{9}$$

This will give us a way to find a noisy image at step t of the forward process, given the original image x_0 (we ignore the noise $\epsilon \sim \mathcal{N}_D(0, I)$):

$$q(x_t|x_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I\right) \tag{10}$$

We now also know that $1 - \bar{\alpha}_t$ is the equivalent variance of the noise added to x_0 to obtain x_t during the forward process. [8]

Choosing the variances β_t is an important step when developing the forward process.
Ho et al. [3] chose to use a linear schedule starting from $\beta_1 = 10^{-4}$ and increasing linearly until $\beta_T = 0.02$ in order for them to be small compared to the pixel values that were in $[-1, 1]$. [3]
However we will see in section 3 that this is not the best choice, as this destroyed the images too quickly closer to the end of the process while providing little to sample quality, which made the model be sub-optimal for 64×64 and 32×32 images. [8]

Finally, Ho et al. [3] chose $T = 1000$ in order to be able to compare their model with Sohl et al.'s [12] model, which also used $T = 1000$ for most image experiments.
We will again see in section 3 that so many steps can make the sampling slow and that there are better choices. [8]

2.2 Reverse Process

As mentioned previously, with a correct choice of β_t, x_T will become a sample of an isotropic Gaussian distribution over $\mathbb{R}^D$ as $T \to \infty$. [8, 12]
This should mean that we can take a sample $x_T \sim \mathcal{N}_D(0, I)$ and reverse the forward process to obtain the original image x_0. However, this means that we need to use the entire dataset in order to figure out $q(x_{t-1}|x_t)$, which in practice cannot be done. [8]
Therefore, an estimation $p_\theta(x_{t-1}|x_t)$ of $q(x_{t-1}|x_t)$ is found using a neural network, with θ the parameters of the neural network. [8]

The reverse process is defined as follows [3]:

$$p_\theta(x_0, ..., x_T) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t) \tag{11}$$

where

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \tag{12}$$

Equation **??** gives us the full reverse process, starting from the final image x_T to the original image x_0.
Equation **??** gives us the reverse process at step t, where we estimate the noise that was added to the image at step t of the forward process in order to find the image x_{t-1}.
Two parameters must be estimated to find the reverse process at step t (from x_t to x_{t-1}): the mean $\mu_\theta(x_t, t)$ and the variance $\Sigma_\theta(x_t, t)$.
At first, Ho et al. [3] used neural networks to estimate both the mean and the variance, but explain that estimating the variance was not necessary as there was little difference between the estimated variance $\sigma_t^2 = \tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}$ and the actual values $\sigma_t^2 = \beta_t$ [3]. Thus they set $\Sigma_\theta(x_t, t) = \sigma_t^2 I = \beta_t I$.
As for the mean, it is estimated using a neural network trained to optimise a variational lower bound (VLB) of the negative log likelihood $\mathbb{E}[-\log p_\theta(x_0)]$. [3]

2.2.1 Variational Lower Bound

In order to find the negative log likelihood $\mathbb{E}[-\log p_\theta(x_0)]$, we would need to know $p_\theta(x_0)$ which means going through T steps, which is computationally expensive.

Thus, Ho et al. [3] use a variational lower bound (VLB or more commonly known as $ELBO$) to estimate the negative log likelihood [3, 12].

The variational lower bound given by Dederik et al. is defined as: [4]

$$VLB = \mathbb{E}_{q_\phi(x|z)} \left[-\log q_\phi(z|x) + \log p_\theta(x,z) \right] \tag{13}$$

$$= -D_{KL}(q_\phi(z|x) \| \, p_\theta(z)) + \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x|z) \right] \tag{14}$$

where X, Z are jointly distributed random variables with distribution $p_\theta(x,z)$, $x \sim p_\theta$ and q_ϕ is any distribution.

This gives us:

$$VLB = \log p_\theta \left(x_0 \right) - D_{KL} \left(q \left(x_{1:T} | x_0 \right) \| p_\theta \left(x_{1:T} | x_0 \right) \right) \tag{15}$$

Since we need to minimise the loss with the negative log likelihood, we write:

$$-VLB = -\log p_\theta \left(x_0 \right) + D_{KL} \left(q \left(x_{1:T} | x_0 \right) \| p_\theta \left(x_{1:T} | x_0 \right) \right) \tag{16}$$

We know that the Kullback-Leibler divergence D_{KL} is non-negative, which means that we can write:

$$-\log p_\theta \left(x_0 \right) \leq -\log p_\theta \left(x_0 \right) + D_{KL} \left(q \left(x_{1:T} | x_0 \right) \| p_\theta \left(x_{1:T} | x_0 \right) \right) \tag{17}$$

$$= -\log p_\theta \left(x_0 \right) + \mathbb{E}_q \left[\log \frac{q \left(x_{1:T} | x_0 \right)}{p_\theta \left(x_{1:T} | x_0 \right)} \right] \tag{18}$$

$$= -\log p_\theta \left(x_0 \right) + \mathbb{E}_q \left[\log \frac{q \left(x_{1:T} | x_0 \right)}{\frac{p_\theta(x_0, x_{1:T})}{p_\theta(x_0)}} \right] \tag{19}$$

$$= -\log p_\theta \left(x_0 \right) + \mathbb{E}_q \left[\log \frac{q \left(x_{1:T} | x_0 \right)}{\frac{p_\theta(x_{0:T})}{p_\theta(x_0)}} \right] \tag{20}$$

$$= -\log p_\theta \left(x_0 \right) + \mathbb{E}_q \left[\log \frac{q \left(x_{1:T} | x_0 \right)}{p_\theta \left(x_{0:T} \right)} + \log p_\theta \left(x_0 \right) \right] \tag{21}$$

$$= \mathbb{E}_q \left[\log \frac{q \left(x_{1:T} | x_0 \right)}{p_\theta \left(x_{0:T} \right)} \right] \tag{22}$$

so that we obtain that:

$$-\log p_\theta \left(x_0 \right) \leq \mathbb{E}_q \left[\log \frac{q \left(x_{1:T} | x_0 \right)}{p_\theta \left(x_{0:T} \right)} \right] \tag{23}$$

We can subsequently define L_{VLB} as the loss to be minimised:

$$L_{VLB} = \mathbb{E}_q \left[\log \frac{q \left(x_{1:T} | x_0 \right)}{p_\theta \left(x_{0:T} \right)} \right] \tag{24}$$

Ho et al. [3] reformulate the loss in order to reduce variance:

$$
\begin{aligned}
L_{VLB} &= \mathbb{E}_q \left[\log \frac{q\left(x_{1:T}|x_0\right)}{p_\theta\left(x_{0:T}\right)} \right] && (25)\\
&= \mathbb{E}_q \left[\log \frac{\prod_{t\geq 1} q\left(x_t|x_{t-1}\right)}{p_\theta\left(x_T\right) \prod_{t\geq 1} p_\theta\left(x_{t-1}|x_t\right)} \right] && (26)\\
&= \mathbb{E}_q \left[-\log\left(p_\theta\left(x_T\right)\right) + \log \frac{\prod_{t\geq 1} q\left(x_t|x_{t-1}\right)}{\prod_{t\geq 1} p_\theta\left(x_{t-1}|x_t\right)} \right] && (27)\\
&= \mathbb{E}_q \left[-\log\left(p_\theta\left(x_T\right)\right) + \log \prod_{t\geq 1} q\left(x_t|x_{t-1}\right) - \log \prod_{t\geq 1} p_\theta\left(x_{t-1}|x_t\right) \right] && (28)\\
&= \mathbb{E}_q \left[-\log\left(p_\theta\left(x_T\right)\right) + \sum_{t\geq 1} \log q\left(x_t|x_{t-1}\right) - \sum_{t\geq 1} \log p_\theta\left(x_{t-1}|x_t\right) \right] && (29)\\
&= \mathbb{E}_q \left[-\log\left(p_\theta\left(x_T\right)\right) + \sum_{t\geq 1} \log q\left(x_t|x_{t-1}\right) - \log p_\theta\left(x_{t-1}|x_t\right) \right] && (30)\\
&= \mathbb{E}_q \left[-\log\left(p_\theta\left(x_T\right)\right) + \sum_{t\geq 1} \log \frac{q\left(x_t|x_{t-1}\right)}{p_\theta\left(x_{t-1}|x_t\right)} \right] && (31)\\
&= \mathbb{E}_q \left[-\log\left(p_\theta\left(x_T\right)\right) + \log \frac{q\left(x_1|x_0\right)}{p_\theta\left(x_0|x_1\right)} + \sum_{t\geq 2} \log \frac{q\left(x_t|x_{t-1}\right)}{p_\theta\left(x_{t-1}|x_t\right)} \right] && (32)\\
&= \mathbb{E}_q \left[-\log\left(p_\theta\left(x_T\right)\right) + \log \frac{q\left(x_1|x_0\right)}{p_\theta\left(x_0|x_1\right)} + \sum_{t\geq 2} \log \frac{q\left(x_{t-1}|x_t, x_0\right) q\left(x_t|x_0\right)}{q\left(x_{t-1}|x_0\right) p_\theta\left(x_{t-1}|x_t\right)} \right] && (33)\\
&= \mathbb{E}_q \left[-\log\left(p_\theta\left(x_T\right)\right) + \log \frac{q\left(x_1|x_0\right)}{p_\theta\left(x_0|x_1\right)} + \sum_{t\geq 2} \log \frac{q\left(x_{t-1}|x_t, x_0\right)}{p_\theta\left(x_{t-1}|x_t\right)} + \sum_{k\geq 2} \log \frac{q\left(x_t|x_0\right)}{q\left(x_{t-1}|x_0\right)} \right] && (34)\\
&= \mathbb{E}_q \left[-\log\left(p_\theta\left(x_T\right)\right) + \log \frac{q\left(x_1|x_0\right)}{p_\theta\left(x_0|x_1\right)} + \log \frac{q\left(x_T|x_0\right)}{q\left(x_1|x_0\right)} + \sum_{t\geq 2} \log \frac{q\left(x_{t-1}|x_t, x_0\right)}{p_\theta\left(x_{t-1}|x_t\right)} \right] && (35)\\
&= \mathbb{E}_q \left[-\log\left(p_\theta\left(x_T\right)\right) + \log \frac{q\left(x_T|x_0\right)}{p_\theta\left(x_0|x_1\right)} + \sum_{t\geq 2} \log \frac{q\left(x_{t-1}|x_t, x_0\right)}{p_\theta\left(x_{t-1}|x_t\right)} \right] && (36)\\
&= \mathbb{E}_q \left[-\log\left(p_\theta\left(x_0|x_1\right)\right) + \log \frac{q\left(x_T|x_0\right)}{p_\theta\left(x_T\right)} + \sum_{t\geq 2} \log \frac{q\left(x_{t-1}|x_t, x_0\right)}{p_\theta\left(x_{t-1}|x_t\right)} \right] && (37)\\
&= \mathbb{E}_q \Bigg[D_{KL}\left(q\left(x_T|x_0\right) \parallel p\left(x_T\right)\right) && (38)\\
&\qquad + \sum_{t\geq 2} D_{KL}\left(q\left(x_{t-1}|x_t, x_0\right) \parallel p_\theta\left(x_{t-1}|x_t\right)\right) - \log\left(p_\theta\left(x_0|x_1\right)\right) \Bigg]
\end{aligned}
$$

This formulation of the loss can be split up into 3 parts:

$$L_T = D_{KL}\left(q\left(x_T|x_0\right) \parallel p\left(x_T\right)\right) \tag{39}$$

$$L_{1:T-1} = \sum_{t\geq 2} D_{KL}\left(q\left(x_{t-1}|x_t, x_0\right) \parallel p_\theta\left(x_{t-1}|x_t\right)\right) \tag{40}$$

$$L_0 = -\log\left(p_\theta\left(x_0|x_1\right)\right) \tag{41}$$

It is worth noting that at equation **??**, the terms of q are conditioned on x_0; this is in order to be able to easily compute the forward process posteriors, as they are much easier to find given the original image x_0.
Ho et al. [3] define it as:

$$q\left(x_{t-1}|x_t, x_0\right) = \mathcal{N}\left(x_{t-1}; \tilde{\mu}_t\left(x_t, x_0\right), \tilde{\beta}_t I\right) \tag{42}$$

Using Bayes' theorem, we find:

$$q\left(x_{t-1}|x_t, x_0\right) = q\left(x_t|x_{t-1}, x_0\right) \frac{q\left(x_{t-1}|x_0\right)}{q\left(x_t|x_0\right)} \tag{43}$$

Assuming that $q\left(x_t|x_{t-1}, x_0\right) = \mathcal{N}\left(x_{t-1}; \tilde{\mu}_t\left(x_t, x_0\right), \tilde{\beta}_t I\right)$, then:

$$q\left(x_{t-1}|x_t, x_0\right) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I\right) \frac{\mathcal{N}\left(\sqrt{\bar{\alpha}_{t-1}}x_0, \left(1-\bar{\alpha}_{t-1}\right) I\right)}{\mathcal{N}\left(\sqrt{\bar{\alpha}_t}x_0, \left(1-\bar{\alpha}_t\right) I\right)} \tag{44}$$

$$= \frac{1}{\sqrt{2\pi\left(1-\bar{\alpha}_t\right)\left(1-\bar{\alpha}_{t-1}\right)\left(\sqrt{1-\beta_t}\right)}} e^{-\frac{1}{2}\left(\frac{\left(x_t-\sqrt{1-\beta_t}x_{t-1}\right)^2}{\beta_t} + \frac{\left(x_{t-1}-\sqrt{\bar{\alpha}_{t-1}}x_0\right)^2}{1-\bar{\alpha}_{t-1}} - \frac{\left(x_t-\sqrt{\alpha_t}x_0\right)^2}{1-\bar{\alpha}_t}\right)} \tag{45}$$

Note: we rewrote $q\left(x_t|x_{t-1}, x_0\right)$ as $q\left(x_t|x_{t-1}\right)$ since x_0 adds no new information not already present in x_{t-1}.

We can ignore the constant factor in front and thus obtain:

$$= \exp\left(-\frac{1}{2}\left(\frac{x_t^2 - 2\sqrt{\alpha_t}x_{t-1}x_t + \alpha_t x_{t-1}^2}{\beta_t} + \frac{x_{t-1}^2 - 2\sqrt{\bar{\alpha}_{t-1}}x_0 x_{t-1} + \bar{\alpha}_{t-1}x_0^2}{1-\bar{\alpha}_{t-1}} - \frac{\left(x_t-\sqrt{\alpha}_t x_0\right)^2}{1-\bar{\alpha}_t}\right)\right) \tag{46}$$

$$= \exp\left(-\frac{1}{2}\left(\frac{x_t^2}{\beta_t} - \frac{2\sqrt{\alpha_t}x_t x_{t-1}}{\beta_t} + \frac{\alpha_t x_{t-1}^2}{\beta_t} + \frac{x_{t-1}^2}{1-\bar{\alpha}_{t-1}} - \frac{2\sqrt{\bar{\alpha}_{t-1}}x_0 x_{t-1}}{1-\bar{\alpha}_{t-1}} + \frac{\bar{\alpha}_{t-1}x_0^2}{1-\bar{\alpha}_{t-1}} - \frac{\left(x_t-\sqrt{\alpha}_t x_0\right)^2}{1-\bar{\alpha}_t}\right)\right) \tag{47}$$

$$= \exp\left(-\frac{1}{2}\left(x_{t-1}^2\left(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right) + x_{t-1}\left(\frac{-2\sqrt{\alpha_t}x_t}{\beta_t} - \frac{2\sqrt{\bar{\alpha}_{t-1}}x_0}{1-\bar{\alpha}_{t-1}}\right) + C\left(x_t, x_0\right)\right)\right) \tag{48}$$

where:

$$C\left(x_t, x_0\right) = \frac{x_t^2}{\beta_t} + \frac{\bar{\alpha}_{t-1} x_0^2}{1-\bar{\alpha}_{t-1}} - \frac{\left(x_t - \sqrt{\alpha}_t x_0\right)^2}{1-\bar{\alpha}_t} \tag{49}$$

Considering $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=0}^{t} \alpha_i$, we can simplify the expression to:

$$= \exp\left(-\frac{1}{2}\left(x_{t-1}^2\left(\frac{\alpha_t}{1-\alpha_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right) - 2\left(\frac{\sqrt{\alpha_t} x_t}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}} x_0}{1-\bar{\alpha}_{t-1}}\right) x_{t-1} + C\left(x_t, x_0\right)\right)\right) \tag{50}$$

$$= \exp\left(-\frac{1}{2}\left(x_{t-1}^2\left(\frac{\alpha_t\left(1-\bar{\alpha}_{t-1}\right) + \left(1-\alpha_t\right)}{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}\right) - 2\left(\frac{\sqrt{\alpha_t} x_t}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}} x_0}{1-\bar{\alpha}_{t-1}}\right) x_{t-1} + C\left(x_t, x_0\right)\right)\right) \tag{51}$$

$$= \exp\left(-\frac{1}{2}\left(x_{t-1}^2\left(\frac{\alpha_t - \alpha_t\bar{\alpha}_{t-1} + 1 - \alpha_t}{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}\right) - 2\left(\frac{\sqrt{\alpha_t} x_t}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}} x_0}{1-\bar{\alpha}_{t-1}}\right) x_{t-1} + C\left(x_t, x_0\right)\right)\right) \tag{52}$$

$$= \exp\left(-\frac{1}{2}\left(x_{t-1}^2\left(\frac{1-\bar{\alpha}_{t-1}}{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}\right) - 2\left(\frac{\sqrt{\alpha_t} x_t}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}} x_0}{1-\bar{\alpha}_{t-1}}\right) x_{t-1} + C\left(x_t, x_0\right)\right)\right) \tag{53}$$

$$= \exp\left(-\frac{1}{2}\left(\left(\frac{1-\bar{\alpha}_{t-1}}{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}\right)\left(x_{t-1}^2 - 2\frac{\frac{\sqrt{\alpha_t} x_t}{1-\alpha_t} + \frac{\sqrt{\bar{\alpha}_{t-1}} x_0}{1-\bar{\alpha}_{t-1}}}{\frac{1-\bar{\alpha}_{t-1}}{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}} x_{t-1} + \frac{C\left(x_t, x_0\right)}{\frac{1-\bar{\alpha}_{t-1}}{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}}\right)\right)\right) \tag{54}$$

$$= \exp\left(-\frac{1}{2}\left(\left(\frac{1-\bar{\alpha}_{t-1}}{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}\right)\left(x_{t-1}^2 - 2\frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right) x_t + \sqrt{\bar{\alpha}_{t-1}}\left(1-\alpha_t\right) x_0}{1-\bar{\alpha}_t} + \frac{C\left(x_t, x_0\right)}{\frac{1-\bar{\alpha}_{t-1}}{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)}}\right)\right)\right) \tag{55}$$

As for the final term, since:

$$C\left(x_t, x_0\right) = \frac{x_t^2}{\beta_t} + \frac{\bar{\alpha}_{t-1} x_0^2}{1-\bar{\alpha}_{t-1}} - \frac{\left(x_t - \sqrt{\alpha}_t x_0\right)^2}{1-\bar{\alpha}_t} \tag{56}$$

$$= \frac{x_t^2}{1-\alpha_t} + \frac{\bar{\alpha}_{t-1} x_0^2}{1-\bar{\alpha}_{t-1}} - \frac{x_t^2 - 2\sqrt{\alpha}_t x_t x_0 + \alpha_t x_0^2}{1-\bar{\alpha}_t} \tag{57}$$

We obtain:

$$C\left(x_t, x_0\right) / \frac{1-\bar{\alpha}_{t-1}}{\left(1-\alpha_t\right)\left(1-\bar{\alpha}_{t-1}\right)} \tag{58}$$

$$= \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_{t-1}} \left(\frac{x_t^2}{1-\alpha_t} + \frac{\bar{\alpha}_{t-1} x_0^2}{1-\bar{\alpha}_{t-1}} - \frac{x_t^2 - 2\sqrt{\alpha}_t x_t x_0 + \alpha_t x_0^2}{1-\bar{\alpha}_t} \right) \tag{59}$$

$$= x_t^2 \left(\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} - \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)^2} \right) \tag{60}$$

$$+ x_0^2 \left(\frac{(1-\alpha_t)(\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} - \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})\bar{\alpha}_t}{(1-\bar{\alpha}_t)^2} \right)$$

$$+ \frac{2x_t x_0 \sqrt{\bar{\alpha}_t}(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)^2}$$

$$= x_t^2 \left(\frac{(1-\bar{\alpha}_t)(1-\bar{\alpha}_{t-1}) - (1-\alpha_t)(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)^2} \right) \tag{61}$$

$$+ x_0^2 \left(\frac{(1-\bar{\alpha}_t)(1-\alpha_t)\bar{\alpha}_{t-1} - (1-\alpha_t)(1-\bar{\alpha}_{t-1})\bar{\alpha}_t}{(1-\bar{\alpha}_t)^2} \right)$$

$$+ \frac{2x_t x_0 \sqrt{\bar{\alpha}_t}(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)^2}$$

$$= x_t^2 \frac{(1-\bar{\alpha}_{t-1})^2 \alpha_t}{(1-\bar{\alpha}_t)^2} + x_0^2 \frac{(1-\bar{\alpha}_t)^2 \bar{\alpha}_{t-1}}{(1-\bar{\alpha}_t)^2} + \frac{2x_t x_0 \sqrt{\bar{\alpha}_t}(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{(1-\bar{\alpha}_t)^2} \tag{62}$$

$$= \frac{\left(x_t \sqrt{\alpha_t}(1-\bar{\alpha}_{t-1}) + x_0 \sqrt{\bar{\alpha}_{t-1}}(1-\bar{\alpha}_t)\right)^2}{(1-\bar{\alpha}_t)^2} \tag{63}$$

$$= \left(\frac{\left(x_t \sqrt{\alpha_t}(1-\bar{\alpha}_{t-1}) + x_0 \sqrt{\bar{\alpha}_{t-1}}(1-\bar{\alpha}_t)\right)}{(1-\bar{\alpha}_t)} \right)^2 \tag{64}$$

We can now replace this value into the original expression:

$$q(x_{t-1}|x_t, x_0) = \exp\left(-\frac{1}{2}\left(\left(\frac{1}{\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_{t-1}}}\right)\right.\right. \tag{65}$$

$$\left.\left.\left(x_{t-1}^2 - 2\frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{t-1})\,x_t + \sqrt{\bar{\alpha}_{t-1}}\,(1-\alpha_t)\,x_0}{1-\bar{\alpha}_t}\right.\right.\right.$$

$$\left.\left.\left.+\left(\frac{\left(x_t\sqrt{\alpha_t}\,(1-\bar{\alpha}_{t-1}) + x_0\sqrt{\bar{\alpha}_{t-1}}\,(1-\bar{\alpha}_t)\right)}{(1-\bar{\alpha}_t)}\right)^2\right)\right)\right) \tag{66}$$

$$= \exp\left(-\frac{1}{2}\left(\frac{1}{\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_{t-1}}}\right)\right. \tag{67}$$

$$\left.\left(x_{t-1} - \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)x_0}{1-\bar{\alpha}_t}\right)^2\right)$$

$$= \mathcal{N}\left(x_{t-1}; \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)x_0}{1-\bar{\alpha}_t}, \frac{(1-\alpha_t)\,(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}I\right) \tag{68}$$

$$= \mathcal{N}\left(x_{t-1}; \tilde{\mu}(x_t, x_0), \tilde{\beta}_t I\right) \tag{69}$$

We have therefore obtained $\tilde{\mu}(x_t, x_0)$ and $\tilde{\beta}_t$ mentioned in Ho et al.'s [3] paper.
The authors also rearrange $\tilde{\mu}(x_t, x_0)$ in order to remove the dependency on x_0:

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \tag{70}$$

$$\Leftrightarrow x_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon}{\sqrt{\bar{\alpha}_t}} \tag{71}$$

$$\Rightarrow \tilde{\mu}(x_t, t) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon}{\sqrt{\bar{\alpha}_t}}}{1-\bar{\alpha}_t} \tag{72}$$

$$= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + (1-\alpha_t)\left(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon\right)\frac{\sqrt{\bar{\alpha}_{t-1}}}{\sqrt{\bar{\alpha}_t}}}{1-\bar{\alpha}_t} \tag{73}$$

$$= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t}{1-\bar{\alpha}_t} + \frac{(1-\alpha_t)\left(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon\right)\frac{1}{\sqrt{\alpha_t}}}{1-\bar{\alpha}_t} \tag{74}$$

$$= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{(1-\alpha_t)}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}x_t - \frac{(1-\alpha_t)(\sqrt{1-\bar{\alpha}_t})}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\epsilon \tag{75}$$

$$= \left(\frac{\sqrt{\alpha_t}\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)} + \frac{(1-\alpha_t)}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\right)x_t - \frac{(1-\alpha_t)(\sqrt{1-\bar{\alpha}_t})}{\sqrt{(1-\bar{\alpha}_t)}\sqrt{(1-\bar{\alpha}_t)}\sqrt{\alpha_t}}\epsilon \tag{76}$$

$$= \left(\frac{\alpha_t(1-\bar{\alpha}_{t-1}) + (1-\alpha_t)}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\right)x_t - \frac{(1-\alpha_t)}{\sqrt{(1-\bar{\alpha}_t)}\sqrt{\alpha_t}}\epsilon \tag{77}$$

$$= \left(\frac{\alpha_t - \bar{\alpha}_t + 1 - \alpha_t}{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}\right)x_t - \frac{1-\alpha_t}{\sqrt{(1-\bar{\alpha}_t)}\sqrt{\alpha_t}}\epsilon \tag{78}$$

$$= \frac{1}{\sqrt{\alpha_t}} x_t - \frac{1-\alpha_t}{\sqrt{(1-\bar{\alpha}_t)}\sqrt{\alpha_t}} \epsilon \tag{79}$$

$$= \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{(1-\bar{\alpha}_t)}} \epsilon \right) \tag{80}$$

Another important point is that L_T can be omitted since q has no learnable parameters and with the fact that it will almost become a Gaussian distribution, the Kullback-Leibler Divergence $D_{KL}\left(q\left(x_T|x_0\right) \| \, p\left(x_T\right)\right)$ between a nearly Gaussian distribution and a Gaussian distribution $p\left(x_T\right) = \mathcal{N}\left(x_t; 0, I\right)$ will be close to 0. [3]
We thus find ourselves with:

$$L_{VLB} = \mathbb{E}_q \left[\sum_{t \geq 2} D_{KL}\left(q\left(x_{t-1}|x_t, x_0\right) \| \, p_\theta\left(x_{t-1}|x_t\right)\right) - \log\left(p_\theta\left(x_0|x_1\right)\right) \right] \tag{81}$$

2.2.2 Simplified training objective

Ho et al. [3] simplify the training objective by making several changes in order to obtain an easy quantity for minimising the loss.

First of all, $L_{1:T-1}$ can be written as such:

$$L_{1:T-1} = D_{KL}\left(q\left(x_{t-1}|x_t, x_0\right) \| \, p_\theta\left(x_{t-1}|x_t\right)\right) \tag{82}$$

$$= D_{KL}\left(\mathcal{N}(x_{t-1}; \tilde{\mu}_t\left(x_t, t\right), \sigma_t^2 I) \| \, \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2 I\right)\right) \tag{83}$$

The Kullback-Leibler divergence between the two multivariate Gaussian distributions is given by:

$$D_{KL}\left(\mathcal{N}(x_{t-1}; \tilde{\mu}_t\left(x_t, t\right), \sigma_t^2 I) \| \, \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2 I\right)\right) \tag{84}$$

$$= \frac{1}{2}\left(tr\left((\sigma_t^2 I)^{-1}(\sigma_t^2 I)\right) - d + \left(\mu_\theta - \tilde{\mu}_t\right)^T \left(\sigma_t^2 I\right)^{-1} \left(\mu_\theta - \tilde{\mu}_t\right) + \log \frac{\det(\sigma_t^2 I)}{\det(\sigma_t^2 I)} \right) \tag{85}$$

$$= \frac{1}{2}\left(d - d + \left(\mu_\theta - \tilde{\mu}_t\right)^T \left(\sigma_t^2 I\right)^{-1} \left(\mu_\theta - \tilde{\mu}_t\right) + \log 1 \right) \tag{86}$$

$$= \frac{1}{2}\left(\left(\mu_\theta - \tilde{\mu}_t\right)^T \left(\sigma_t^2 I\right)^{-1} \left(\mu_\theta - \tilde{\mu}_t\right) \right) \tag{87}$$

$$= \frac{1}{2\sigma_t^2}\left(\left(\mu_\theta - \tilde{\mu}_t\right)^T \left(\mu_\theta - \tilde{\mu}_t\right) \right) \tag{88}$$

$$= \frac{1}{2\sigma_t^2} \|\mu_\theta - \tilde{\mu}_t\|_2^2 \tag{89}$$

However, Ho et al. [3] rewrite this formula so that it depends only on the noise ϵ:

$$\frac{1}{2\sigma_t^2} \|\mu_\theta - \tilde{\mu}_t\|_2^2 = \frac{1}{2\sigma_t^2} \left\| \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{(1-\bar{\alpha}_t)}} \epsilon \right) - \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{(1-\bar{\alpha}_t)}} \epsilon_\theta \right) \right\|_2^2 \tag{90}$$

$$= \frac{1}{2\sigma_t^2} \left\| \frac{1}{\sqrt{\alpha_t}} \frac{1-\alpha_t}{\sqrt{(1-\bar{\alpha}_t)}} (\epsilon - \epsilon_\theta) \right\|_2^2 \tag{91}$$

$$= \frac{1}{2\sigma_t^2} \frac{1}{\alpha_t} \frac{(1-\alpha_t)^2}{1-\bar{\alpha}_t} \|\epsilon - \epsilon_\theta\|_2^2 \tag{92}$$

$$= \frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \|\epsilon - \epsilon_\theta\|_2^2 \tag{93}$$

Furthermore, they found that simply ignoring the factor in front gave better results, giving us:

$$L_{1:T-1} = \|\epsilon - \epsilon_\theta\|_2^2 \tag{94}$$

As for the second term L_0, the authors define $p_\theta(x_0|x_1)$ as:

$$p_\theta(x_0|x_1) = \prod_{i=1}^{D} \int_{\delta_-(x_0^i)}^{\delta_+(x_0^i)} \mathcal{N}(x; \mu_\theta^i(x_1, 1), \sigma_1^2)\, dx \tag{95}$$

where $\delta_+(x)$ and $\delta_-(x)$ are defined as:

$$\delta_+(x) = \begin{cases} \infty & \text{if } x = 1 \\ x + \frac{1}{255} & \text{if } x < 1 \end{cases} \tag{96}$$

$$\delta_-(x) = \begin{cases} -\infty & \text{if } x = -1 \\ x - \frac{1}{255} & \text{if } x > -1 \end{cases} \tag{97}$$

with D the data dimensionality and i one coordinate from the image.

This is a decoder for the last step in order to get pixel values between $\{0, 1, ..., 255\}$ from the values in $[-1, 1]$ used for the neural network [3, 8]
However the authors also ignored the entire term, giving us:

$$L_{\text{simple}}(\theta) = \mathbb{E}_q \left[\|\epsilon - \epsilon_\theta\|_2^2\right] \tag{98}$$

2.2.3 Model architecture

Ho et al. [3] as well as the following papers [13, 8, 2] use a U-Net style architecture for the neural network to estimate the noise.

The U-Net architecture is a convolutional neural network (CNN) that was originally designed for image segmentation in biomedical imaging [10]. It consists of an encoder (or contracting path) and a decoder (or expansive path) that are connected by a bottleneck.
The encoder takes in the noisy image as input and repeatedly applies unpadded convolutions followed by max pooling layers in order to reduce the spatial dimensions of the image. [10]
The bottleneck contains 3 unpadded convolutions each followed by a rectified linear unit (ReLU) activation function. [10]
The decoder then applies up-convolutions followed by unpadded convolutions to increase the spatial dimensions of the image, with at each step a concatenation of the feature maps from the encoder. [10]

However, Ho et al. [3] made some changes to the original U-Net architecture.
First, they state that they used 4 feature map resolutions for their 32×32 resolution model, and 6 for the 256×256 resolution model. For each of these feature maps, they used two convolutional residual blocks similar of those of a Wide ResNet [16] consisting of a convolutional layer, a ReLU activation function, a convolutional layer, a group normalisation layer, and another ReLU activation function.
The "residual" part of the block comes from the fact that the result at the end of the block on the encoder side is concatenated with the result at the end of the block on the decoder side in order to consider features that could have been lost during the downsampling. [6]

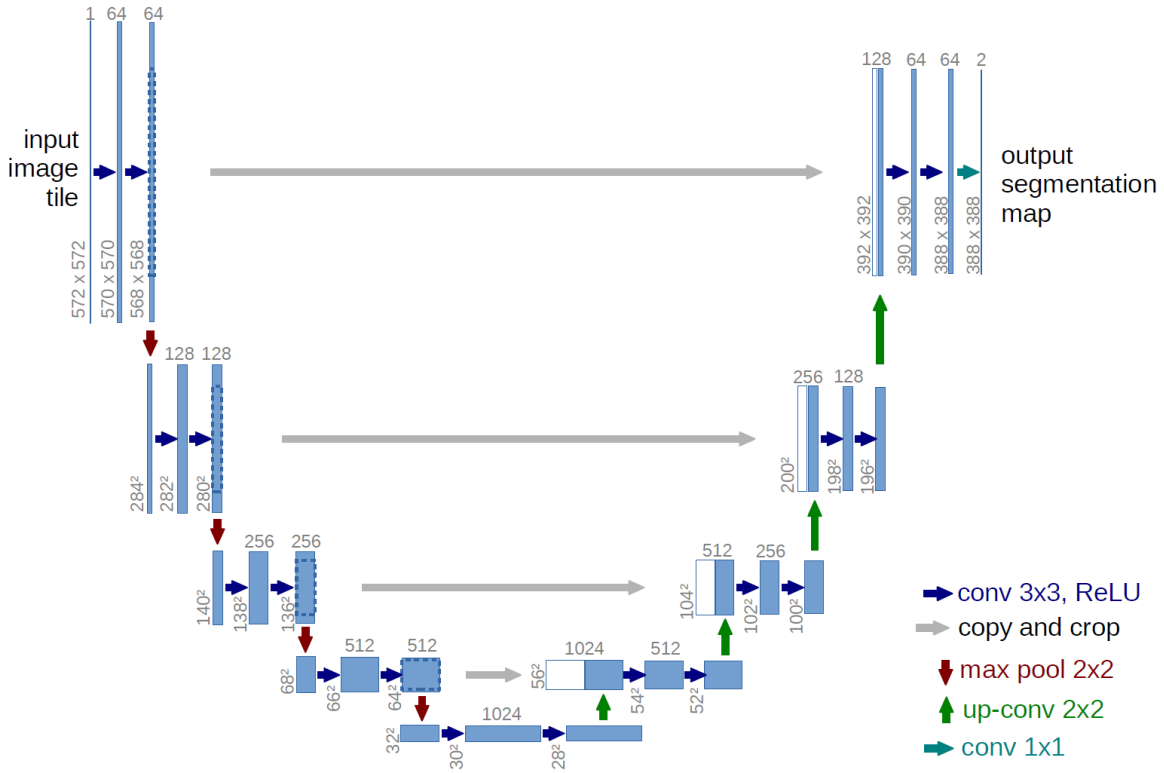

Figure 1: Original U-Net architecture [10]

They used group normalisation [15] rather than weight normalisation [11], which is a normalisation technique that divides the channels into groups and computes the mean and standard deviation for each group [3].
Finally, they state that they used self-attention layers from the Transformer architecture [14] at the 16×16 resolution between the two convolutional residual blocks. [3]

Since the model needs to be able to predict the noise at any timestep t, the authors added a time embedding to the input of the network using the sinusoidal positional embedding from the Transformer architecture [14], which allows the parameters to be shared across all timesteps.
The sinusoidal positional embedding is given by [14]:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{99}$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{100}$$

where pos is the position and i is the dimension

Ho et al. [3] present two algorithms: one for training the model and one for sampling from the model:

Algorithm 1 Training

repeat
  $x_0 \sim q(x_0)$
  $t \sim \text{Uniform}(\{1, \dots, T\})$
  $\epsilon \sim \mathcal{N}(0, I)$
  Take gradient descent step on:
    $\nabla_\theta \| \epsilon - \epsilon_\theta \left(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon, t \right) \|^2$
until converged

Algorithm 2 Sampling

$x_T \sim \mathcal{N}(0, I)$
for $t = T, \dots, 1$ **do**
  $z \sim \mathcal{N}(0, I)$ if $t > 0$, else $z = 0$
  $x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t z$
end for
return x_0

3 Improvements upon DDPMs

As mentioned previously, the DDPM paper [3] had certain choices that limited its performance. Several papers [8, 13] following it have improved upon the original DDPM in various ways.

3.1 Improved likelihood

Ho et al. [3] admit that their model, despite its high sample quality, did not have a competitive log-likelihood in comparison to other likelihood-based models such as VAEs and autoregressive models [8]

The *"Improved Denoising Diffusion Probabilistic Models"* paper published by Nichol et al. [8] in 2021 implemented certain changes in order to improve the log-likelihood of the model not only on simpler datasets such as CIFAR-10 [5] that the original DDPM was trained on, but also on more complex datasets such as ImageNet. [4, 9, 8].

In order to improve the log-likelihood, Nichol et al. [8] started by investigating the fixed variance $\Sigma_\theta(x_t, t) = \sigma_t^2 I = \beta_t I$ of the original DDPM and whether or not it could be worth learning it.
Recall that Ho et al. [3] saw no difference in sample quality when using a learned variance $\tilde{\beta}_t$ and therefore opted for using the fixed variance β_t.
Nichol et al. [8] came to the same conclusion, finding that as the number of steps T increased (the authors also mentioned that they used $T = 4000$ for their models rather than $T = 1000$ from the original paper), the difference between β_t and $\tilde{\beta}_t$ remained very small. However, they were still interested in the possibility of learning the variance in order to improve the log-likelihood. [8].
For this, they used an interpolation approach, where they parametrised the variance as an interpolation between the fixed variance β_t and a learned variance $\tilde{\beta}_t$ in the log domain, with v an output vector from the model containing one component per dimension:

$$\Sigma_\theta(x_t, t) = \exp\left(v \log \beta_t + (1 - v) \log \tilde{\beta}_t\right) \tag{101}$$

As the simplified training objective L_{simple} used in the original DDPM paper [3] did not include the variance, Nichol et al. [8] introduced a new training objective L_{hybrid}, defined as:

$$L_{\text{hybrid}} = L_{\text{simple}} + \lambda L_{\text{VLB}} \tag{102}$$

with $\lambda = 0.001$ in order to keep L_{simple} as the main training objective. [8]

At first, they found that the hybrid objective L_{hybrid} achieved a better log-likelihoods and was easier to optimise than just L_{VLB}, contrary to what they had expected. [8]
However, they obtained the best log-likelihoods by optimising L_{VLB} directly but with importance samping, which is a technique where samples of a difficult distribution are taken from a distribution that is easier to compute and then the original distribution is approximated by weighted samples. [8]

They defined it as:

$$L_{\text{VLB}} = \mathbb{E}_{t \sim p_t}\left[\frac{L_t}{p_t}\right] \qquad \text{where } p_t \propto \sqrt{\mathbb{E}\left[L_t^2\right]} \text{ and } \sum p_t = 1 \tag{103}$$

Another change that Nichol et al. [8] made was to use a cosine noising schedule rather than the linear noising schedule used in the original DDPM paper [3].
They explain this change by the fact that towards the end of the forward process, the image is already very noisy and yet more noise is added while not really improving the model, particularly for 32×32 and 64×64 images. [8]

In effect, the input images get destroyed by noise very quickly with the linear noising schedule converging to zero rapidly. [8]

They define the cosine schedule as:

$$f(t) = \cos\left(\frac{\frac{t}{T} + s}{1 + s} \cdot \frac{\pi}{2}\right)^2 \tag{104}$$

$$\bar{\alpha}_t = \frac{f(t)}{f(0)} \tag{105}$$

$$\beta_t = 1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}} \tag{106}$$

T is the total number of timesteps, t is the timestep and s is a small offset used to prevent β_t from being too small when $t = 0$ in order to allow the noise ϵ to be predicted more accurately at the beginning of the process. [8].

With these changes, Nichol et al. [8] were able to achieve a negative log-likelihood (NLL) of 2.94 bits/dim on CIFAR-10 [5] and 3.53 bits/dim on ImageNet [9], compared to the original DDPM paper [3] which had a NLL of 3.70 on CIFAR-10 and 3.77 on ImageNet. [8].

Model	ImageNet	CIFAR
Glow (Kingma & Dhariwal, 2018)	3.81	3.35
Flow++ (Ho et al., 2019)	3.69	3.08
PixelCNN (van den Oord et al., 2016c)	3.57	3.14
SPN (Menick & Kalchbrenner, 2018)	3.52	-
NVAE (Vahdat & Kautz, 2020)	-	2.91
Very Deep VAE (Child, 2020)	3.52	2.87
PixelSNAIL (Chen et al., 2018)	3.52	2.85
Image Transformer (Parmar et al., 2018)	3.48	2.90
Sparse Transformer (Child et al., 2019)	3.44	**2.80**
Routing Transformer (Roy et al., 2020)	**3.43**	-
DDPM (Ho et al., 2020)	3.77	3.70
DDPM (cont flow) (Song et al., 2020b)	-	2.99
Improved DDPM (ours)	**3.53**	**2.94**

Figure 2: Comparison of NLLs of different models [8]

Moreover, Nichol et al. [8] were able to improve sampling speed due to the aforementioned changes, even though they trained their model with $T = 4000$ timesteps.
They were able to train their model with 4000 timesteps, but use only 100 timesteps for sampling which brought them near-optimal FID (Fréchet Inception Distance) scores when using the L_{hybrid} training objective. [8]
They had tried to reduce the number of sampling steps with the original fixed variance versions from the DDPM paper [3] but found that the sample quality degraded a lot more. [8]

3.2 Denoising Diffusion Implicit Models (DDIMs)

As mentioned previously, the DDPM paper [3] has a forward process and a reverse process which are both Markov processes.
While the forward process can be easily calculated for a specific timestep t with equation **??**, the reverse process still requires 1000 timesteps in the case of Ho et al.'s [3] model, and Nichol et al.'s model requires 100 and 4000 timesteps for sampling and training respectively. [8]
This is time-consuming and computationally expensive compared to GANs which only require one forward pass through the network rather than thousands to produce a sample. [13]

To address this issue, Song et al. [13] introduced the Denoising Diffusion Implicit Models (DDIMs) in 2020.
The main objective of DDIMs was to accelerate sampling by making the forward process non-Markovian, which would allow the reverse process to require less iterations. [13]

Song et al. [13] define a family $\mathcal{Q}$ of forward processes, each indexed by $\sigma \in \mathbb{R}_+^T$:

$$q_\sigma(x_{1:T}|\, x_0) = q_\sigma(x_T|\, x_0) \prod_{t=2}^{T} q_\sigma(x_{t-1}|\, x_t, x_0) \tag{107}$$

where

$$q_\sigma(x_T|\, x_0) = \mathcal{N}(\sqrt{\alpha_T} x_0, (1-\alpha_T) I) \tag{108}$$

and for all $t > 1$:

$$q_\sigma(x_{t-1}|\, x_t, x_0) = \mathcal{N}\left(\sqrt{\alpha_{t-1}} x_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \frac{x_t - \sqrt{\alpha_t} x_0}{\sqrt{1-\alpha_t}},\ \sigma_t^2 I\right) \tag{109}$$

It is interesting to note that when $\sigma_t^2 = (1-\alpha_t)(1-\bar{\alpha}_{t-1})/(1-\bar{\alpha}_t)$, we obtain equation **??**, which was the definition of the forward process in the original DDPM paper [3, 13].
Song et al. [13] also note when $\sigma_t^2 = 0$, then x_{t-1} is fixed and known in advance.
Therefore, we can see that as Song et al. [13] state, the value of σ controls the stochasticity (or more simply, the randomness) of the forward process.

The authors use Bayes' rule similarly to the original DDPM paper [3] in equation **??**, but instead to find the forward process $q_\sigma(x_t|\, x_{t-1}, x_0)$:

$$q_\sigma(x_t|\, x_{t-1}, x_0) = \frac{q_\sigma(x_{t-1}|\, x_t, x_0)\, q_\sigma(x_t|\, x_0)}{q_\sigma(x_{t-1}|\, x_0)} \tag{110}$$

Since $q_\sigma(x_t|\, x_{t-1}, x_0)$ means that x_t not only depends on x_{t-1} but also on x_0, we see that the forward process is no longer Markovian, which was what the authors were aiming to achieve. [13]

However, since the reverse process in the original DDPM paper [3] was an estimation of the forward process which was Markovian, a new reverse process was defined.
Their idea of the reverse process is to sample an image x_T like in the original DDPM paper [3], predict x_0 from x_T and then use this prediction of x_0 to sample x_{T-1}. [13]
This is repeated until we reach x_1 and then x_0 is predicted from x_1. [13]
Similar to equation **??** from the DDPM paper, Song et al. [13] define the prediction of x_0 from x_t as:

$$f_\theta^{(t)}(x_t) = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\, \epsilon(x_t, t)}{\sqrt{\bar{\alpha}_t}} \tag{111}$$

Given this, the reverse process is defined as [13]:

$$p_\theta(x_T) = \mathcal{N}_D(0, I) \tag{112}$$

$$p_\theta^{(t)}(x_{t-1}|\, x_t) = \begin{cases} \mathcal{N}(f_\theta^{(1)}(x_1), \sigma_1^2 I) & \text{if } t = 1 \\ q_\sigma(x_{t-1}|\, x_t, f_\theta^{(t)}(x_t)) & \text{if } t > 1 \end{cases} \tag{113}$$

Therefore, x_{t-1} is sampled from x_t [13]:

$$x_{t-1} = \sqrt{\alpha_{t-1}} \left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\, \epsilon(x_t, t)}{\sqrt{\bar{\alpha}_t}} \right) + \sqrt{1 - \alpha_{t-1} - \sigma_t^2} \cdot \epsilon_\theta(x_t, t) + \sigma_t \epsilon_t \tag{114}$$

In order to train the model, Song et al. [13] define the following *variational inference objective*:

$$J_\sigma(\epsilon_\theta) = \mathbb{E}_{q_\sigma(x_{0:T})} \left[\log q_\sigma(x_{1:T}|\, x_0) - \log p_\theta(x_{0:T}) \right] \tag{115}$$

$$= \mathbb{E}_{q_\sigma(x_{0:T})} \left[\log q_\sigma(x_T|\, x_0) + \sum_{t=2}^{T} \log q_\sigma(x_{t-1}|\, x_t, x_0) - \sum_{t=1}^{T} \log p_\theta^{(t)}(x_{t-1}|\, x_t) - \log p_\theta(x_T) \right] \tag{116}$$

This would mean that for different choices of σ, we would have a different objective to optimise. [13] However, the authors [13] prove that J_σ is in fact equivalent to L_γ for certain choices of γ, where L_γ is defined as:

$$L_\gamma(\epsilon_\theta) = \sum_{t=1}^{T} \gamma_t \mathbb{E}_{x_0 \sim q(x_0), \epsilon_t \sim \mathcal{N}(0,I)} \left[\| \epsilon_\theta^{(t)}(\sqrt{\alpha_t} x_0 + \sqrt{1-\alpha_t}\epsilon_t) - \epsilon_t \|_2^2 \right] \tag{117}$$

where γ is a vector of length T with positive values that depend on α.
We easily see that when $\gamma = 1$, we get L_1 which is the training objective from the original DDPM paper. [3, 13]
Song et al. prove that

$$\forall \sigma > 0,\ \exists\, \gamma \in \mathbb{R}_+^T \text{ and } \exists\, C \in \mathbb{R} \text{ s.t. } J_\sigma = L_\gamma + C \tag{118}$$

This means that the optimal solution for J_σ is the same as the optimal solution for L_1, which means that the training objective for DDIMs can be kept the same as the one for DDPMs if the parameters being trained θ are not the same for all timesteps. [13]

In order to speed up the sampling process, Song et al. [13] skip some timesteps in the reverse process. To do this, the forward process is redefined on a subset of timesteps of size S, $\{x_{\tau_1}, ..., x_{\tau_S}\}$ and such that $q(x_{\tau_i}|\, x_0) = \mathcal{N}(\sqrt{\alpha_{\tau_i}} x_0, (1 - \alpha_{\tau_i})I)$
Since this is a subset of the timesteps, the training can be done all timesteps and the sampling can be done on the subset. [13]

With the same amount of training timesteps $T = 1000$ and the same training objective as the DDPM paper [3], Song et al. [13] were able to achieve solid FID scores on CIFAR-10 [5] and CelebA [7] datasets, with far fewer sampling timesteps, ranging from 10 to 100 timesteps. [13]
A new hyperparameter $\eta \in \mathbb{R}_+$ is introduced to control the interpolation between the DDPM σ and the deterministic σ of the DDIM. [13]:

$$\sigma_{\tau_i}(\eta) = \eta \sqrt{\frac{1 - \alpha_{\tau_{i-1}}}{1 - \alpha_{\tau_i}}} \sqrt{1 - \frac{\alpha_{\tau_i}}{\alpha_{\tau_{i-1}}}} \tag{119}$$

The results obtained by Song et al. [13] show that the DDIM outperformed the DDPM for 10, 20, 50, 100 and 1000 timesteps and was only slightly worse at 1000 timesteps and $\sigma_{\tau_i} = \hat{\sigma}_{\tau_i} = \sqrt{1 - \alpha_{\tau_i} / \alpha_{\tau_{i-1}}}$

		CIFAR10 (32 × 32)					CelebA (64 × 64)				
	S	10	20	50	100	1000	10	20	50	100	1000
η	0.0	**13.36**	**6.84**	**4.67**	**4.16**	4.04	**17.33**	**13.73**	**9.17**	**6.53**	3.51
	0.2	14.04	7.11	4.77	4.25	4.09	17.66	14.11	9.51	6.79	3.64
	0.5	16.66	8.35	5.25	4.46	4.29	19.86	16.06	11.01	8.09	4.28
	1.0	41.07	18.36	8.01	5.78	4.73	33.12	26.03	18.48	13.93	5.98
	$\hat{\sigma}$	367.43	133.37	32.72	9.99	**3.17**	299.71	183.83	71.71	45.20	**3.26**

Figure 3: Comparison of FID scores (lower is better) of DDPMs and DDIMs trained on S timesteps, $S < T$ and $\eta = 0$ is DDIM, $\eta = 1$ and $\hat{\sigma}$ are DDPMs [13]

3.3 Guidance

3.3.1 Classifier guidance

3.3.2 Classifier-free guidance

4 Comparison with other diffusion models

4.1 Score-based generative models

4.2 Latent diffusion models

5 Implementation

6 Conclusion

A Convolution of two Gaussian distributions

Equation **??** had given us:

$$x_t = \sqrt{\alpha_t \alpha_{t-1}} x_{t-2} + \mathcal{N}\left(0, \alpha_t \left(1 - \alpha_{t-1}\right) I\right) + \mathcal{N}\left(0, (1 - \alpha_t) I\right)$$

We can find the distribution of the sum of two independant 1D random variables $X \sim \mathcal{N}\left(\mu_X, \sigma_X^2\right)$ and $Y \sim \mathcal{N}\left(\mu_Y, \sigma_Y^2\right)$ as the convolution of two Gaussian distributions.

Let $Z = X + Y$ with X and Y independent.

$$\begin{aligned}
f_Z(z) &= \int_{-\infty}^{\infty} f_X(x) f_Y(z - x)\, dx \\
&= \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\sigma_X^2}} \exp\left[-\frac{(x - \mu_X)^2}{2\sigma_X^2}\right] \frac{1}{\sqrt{2\pi\sigma_Y^2}} \exp\left[-\frac{(z - x - \mu_Y)^2}{2\sigma_Y^2}\right] dx \\
&= \frac{1}{2\pi\sigma_X\sigma_Y} \int_{-\infty}^{\infty} \exp\left[-\frac{(x - \mu_X)^2}{2\sigma_X^2}\right] \exp\left[-\frac{(z - x - \mu_Y)^2}{2\sigma_Y^2}\right] dx \\
&= \frac{1}{2\pi\sigma_X\sigma_Y} \int_{-\infty}^{\infty} \exp\left[-\frac{(x - \mu_X)^2}{2\sigma_X^2} - \frac{(z - x - \mu_Y)^2}{2\sigma_Y^2}\right] dx \\
&= \frac{1}{2\pi\sigma_X\sigma_Y} \int_{-\infty}^{\infty} \exp\left[-\frac{x^2 - 2\mu_X x + \mu_X^2}{2\sigma_X^2} - \frac{z^2 + x^2 - 2xz - 2z\mu_Y + 2\mu_Y x + \mu_Y^2}{2\sigma_Y^2}\right] dx \\
&= \frac{1}{2\pi\sigma_X\sigma_Y} \int_{-\infty}^{\infty} \exp\left[-\frac{\sigma_Y^2(x^2 - 2\mu_X x + \mu_X^2)}{2\sigma_X^2\sigma_Y^2} - \frac{\sigma_X^2(z^2 + x^2 - 2xz - 2z\mu_Y + 2\mu_Y x + \mu_Y^2)}{2\sigma_X^2\sigma_Y^2}\right] dx \\
&= \frac{1}{2\pi\sigma_X\sigma_Y} \int_{-\infty}^{\infty} \exp\left[-\frac{\sigma_Y^2 x^2 - 2\mu_X\sigma_Y^2 x + \mu_X^2\sigma_Y^2 + \sigma_X^2 z^2 + \sigma_X^2 x^2 - 2xz\sigma_X^2 + 2z\mu_Y\sigma_X^2 - 2\mu_Y x\sigma_X^2 + \sigma_X^2\mu_Y^2}{2\sigma_X^2\sigma_Y^2}\right] dx \\
&= \frac{1}{2\pi\sigma_X\sigma_Y} \int_{-\infty}^{\infty} \exp\left[-\frac{\left(\sigma_X^2 + \sigma_Y^2\right) x^2 - 2x\left(\mu_X\sigma_Y^2 + z\sigma_X^2 - \mu_Y\sigma_X^2\right) + \sigma_X^2\left(z - \mu_Y\right)^2 + \mu_X^2\sigma_Y^2}{2\sigma_X^2\sigma_Y^2}\right] dx
\end{aligned}$$

Let $\sigma_Z = \sqrt{\sigma_X^2 + \sigma_Y^2}$.

$$\begin{aligned}
&= \frac{1}{\sqrt{2\pi}\frac{\sigma_X\sigma_Y}{\sigma_Z}} \frac{1}{\sqrt{2\pi}\sigma_Z} \int_{-\infty}^{\infty} \exp\left[-\frac{x^2 - 2x\frac{\sigma_X^2(z-\mu_Y) + \sigma_Y^2\mu_X}{\sigma_Z^2} + \frac{\sigma_X^2(z-\mu_Y)^2 + \mu_X^2\sigma_Y^2}{\sigma_Z^2}}{2\left(\frac{\sigma_X\sigma_Y}{\sigma_Z}\right)^2}\right] dx \\
&= \frac{1}{\sqrt{2\pi}\frac{\sigma_X\sigma_Y}{\sigma_Z}} \frac{1}{\sqrt{2\pi}\sigma_Z} \int_{-\infty}^{\infty} \exp\left[-\frac{\left(x - \frac{\sigma_X^2(z-\mu_Y) + \sigma_Y^2\mu_X}{\sigma_Z^2}\right)^2 - \left(\frac{\sigma_X^2(z-\mu_Y) + \sigma_Y^2\mu_X}{\sigma_Z^2}\right)^2 + \frac{\sigma_X^2(z-\mu_Y)^2 + \sigma_Y^2\mu_X^2}{\sigma_Z^2}}{2\left(\frac{\sigma_X\sigma_Y}{\sigma_Z}\right)^2}\right] dx \\
&= \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\frac{\sigma_X\sigma_Y}{\sigma_Z}} \exp\left[-\frac{\left(x - \frac{\sigma_X^2(z-\mu_Y) + \sigma_Y^2\mu_X}{\sigma_Z^2}\right)^2}{2\left(\frac{\sigma_X\sigma_Y}{\sigma_Z}\right)^2}\right]
\end{aligned}$$

$$\frac{1}{\sqrt{2\pi}\sigma_Z}\exp\left[-\frac{\sigma_Z^2\left(\sigma_X^2(z-\mu_Y)^2+\sigma_Y^2\mu_X^2\right)-\left(\sigma_X^2(z-\mu_Y)+\sigma_Y^2\mu_X\right)^2}{2\sigma_Z^2(\sigma_X\sigma_Y)^2}\right]\,dx$$

$$=\int_{-\infty}^{\infty}\frac{1}{\sqrt{2\pi}\frac{\sigma_X\sigma_Y}{\sigma_Z}}\exp\left[-\frac{\left(x-\frac{\sigma_X^2(z-\mu_Y)+\sigma_Y^2\mu_X}{\sigma_Z^2}\right)^2}{2\left(\frac{\sigma_X\sigma_Y}{\sigma_Z}\right)^2}\right]\frac{1}{\sqrt{2\pi}\sigma_Z}\exp\left[-\frac{(z-(\mu_X+\mu_Y))^2}{2\sigma_Z^2}\right]\,dx$$

$$=\frac{1}{\sqrt{2\pi}\sigma_Z}\exp\left[-\frac{(z-(\mu_X+\mu_Y))^2}{2\sigma_Z^2}\right]\int_{-\infty}^{\infty}\frac{1}{\sqrt{2\pi}\frac{\sigma_X\sigma_Y}{\sigma_Z}}\exp\left[-\frac{\left(x-\frac{\sigma_X^2(z-\mu_Y)+\sigma_Y^2\mu_X}{\sigma_Z^2}\right)^2}{2\left(\frac{\sigma_X\sigma_Y}{\sigma_Z}\right)^2}\right]\,dx$$

$$=\frac{1}{\sqrt{2\pi}\sigma_Z}\exp\left[-\frac{(z-(\mu_X+\mu_Y))^2}{2\sigma_Z^2}\right]$$

$$=\mathcal{N}\left(z;\mu_X+\mu_Y,\sigma_X^2+\sigma_Y^2\right)$$

Given this, we can refer back to our case and therefore we find:

$$\begin{aligned}x_t&=\sqrt{\alpha_t\alpha_{t-1}}x_{t-2}+\mathcal{N}\left(0,\alpha_t\left(1-\alpha_{t-1}\right)I\right)+\mathcal{N}\left(0,(1-\alpha_t)I\right)\\&=\sqrt{\alpha_t\alpha_{t-1}}x_{t-2}+\mathcal{N}\left(0,\alpha_t\left((1-\alpha_{t-1})+1-\alpha_t\right)I\right)\end{aligned}$$

References

[1] Tim Brooks et al. "Video generation models as world simulators". In: (2024). URL: https://openai.com/research/video-generation-models-as-world-simulators.

[2] Prafulla Dhariwal and Alex Nichol. *Diffusion Models Beat GANs on Image Synthesis*. 2021. arXiv: 2105.05233 [cs.LG].

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. *Denoising Diffusion Probabilistic Models*. 2020. arXiv: 2006.11239 [cs.LG].

[4] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2022. arXiv: 1312.6114 [stat.ML].

[5] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. "CIFAR-10 (Canadian Institute for Advanced Research)". In: (2024). URL: http://www.cs.toronto.edu/~kriz/cifar.html.

[6] Zhitong Lai et al. *Rethinking Skip Connections in Encoder-decoder Networks for Monocular Depth Estimation*. 2022. arXiv: 2208.13441 [cs.CV].

[7] Ziwei Liu et al. "Deep Learning Face Attributes in the Wild". In: *Proceedings of International Conference on Computer Vision (ICCV)*. Dec. 2015.

[8] Alex Nichol and Prafulla Dhariwal. *Improved Denoising Diffusion Probabilistic Models*. 2021. arXiv: 2102.09672 [cs.LG].

[9] Aaron van den Oord et al. *Conditional Image Generation with PixelCNN Decoders*. 2016. arXiv: 1606.05328 [cs.CV].

[10] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV].

[11] Tim Salimans and Diederik P. Kingma. *Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks*. 2016. arXiv: 1602.07868 [cs.LG].

[12] Jascha Sohl-Dickstein et al. *Deep Unsupervised Learning using Nonequilibrium Thermodynamics*. 2015. arXiv: 1503.03585 [cs.LG].

[13] Jiaming Song, Chenlin Meng, and Stefano Ermon. *Denoising Diffusion Implicit Models*. 2022. arXiv: 2010.02502 [cs.LG].

[14] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL].

[15] Yuxin Wu and Kaiming He. *Group Normalization*. 2018. arXiv: 1803.08494 [cs.CV].

[16] Sergey Zagoruyko and Nikos Komodakis. *Wide Residual Networks*. 2017. arXiv: 1605.07146 [cs.CV].