Diffusing the art of Diffusion Models

Kshitiz Kumar
ai22mtech02002@iith.ac.in

Tanay Yadav
ai20btech11026@iith.ac.in

Tanmay Goyal
ai20btech11021@iith.ac.in

Tanmay Shah
ee20btech11061@iith.ac.in

Abstract

The present era is truly exciting, with the advent of tools like ChatGPT and DALL-E 2. What is even more thrilling is that better and stronger versions of the same are under wraps, waiting to be unleashed. Moreover, the interleaving of Machine and Deep Learning with techniques from Physics, Chemistry, etc., leaves a lot to be desired, and the technology is making headway at rapid levels. One example of this is Physics-informed Neural Networks, which are making waves in the Deep Learning Community. In this paper, we shall explore Diffusion models, which arrive from a similar realm of interdisciplinary Machine Learning. It also forms the backbone of some of the state-of-the-art models, such as DALL-E 2 and GLIDE.

1. Introduction

The present era is truly exciting, with the advent of tools like ChatGPT and DALL-E 2. The interleaving of Machine and Deep Learning with techniques from Physics, Chemistry, etc., leaves much to be desired, and the technology is making headway rapidly. One example of this is Physics-informed Neural Networks, which are making waves in the Deep Learning Community. In this paper, we shall explore Diffusion models, which arrive from a similar realm of interdisciplinary Machine Learning. It also forms the backbone of some of the state-of-the-art models, such as DALL-E 2 and GLIDE.

In the field of thermodynamics, diffusion is defined as *the movement of particles from a region of higher concentration to that having a lower concentration.* In physics literature, diffusion results in an increase in entropy, whereas in information theory, it leads to a loss in information.

A diffusion model primarily works in the following way: It aims to add Gaussian noise successively and destroy the data, following which it rebuilds the image from the noisy and destroyed data. Diffusion models aim at modelling both the parts of this process (referred to as *forward* and *reverse diffusion*) as Markov Chains. Markov Chains allow for the gradual conversion of one distribution to another, an idea borrowed from non-equilibrium statistical physics. We shall touch upon the mathematics of the terms used here in the upcoming sections.

The idea of a diffusion model is very similar to that of Variational Auto-encoders (VAEs): to project the data into a latent space and recover the original data. However, diffusion models have several advantages over GANs and VAEs, such as little training (unlike the encoder in a VAE, training is minimal due to the presence of Markov Chains), and little to no adversarial training required. We shall touch upon these points in the relevant sections.

2. Problem Statement

Having introduced Diffusion Models, and in particular, the excitement and relevance of the same, the objective of this project is to explore this class of models and simplify much of the underlying principles behind it. Further, we wish to implement a *Simple Diffusion Model*.

3. Literature Review

A Markov chain is one in which the next state depends only on the current state and none of the previous states. In other words, the current state is a sufficient summary of the system's past history. We can represent a Markov Chain mathematically as:

$$\Pr(X_{N+1}|X_1, X_2, \ldots, X_N) = \Pr(X_{N+1}|X_N) \qquad (1)$$

The idea of a Markov chain in the forward diffusion model is used as follows:
Let $q(x_0)$ represent the first state of the Markov Chain, i.e., the initial distribution. At every stage, we choose a transition kernel, which depends only on the current state (Markov Chain), and none of the previous states, i.e

$$q(x_{t+1}|x_t) = T(x_{t+1}|x_t;\ \beta) \qquad (2)$$

where T is the Markov Kernel, and β is the diffusion rate.

Let us assume we have T steps in our Markov Chain. Thus, the distribution after T steps given initial distribution $q(x_0)$ is given by $q(x_1, x_2 \dots x_T|x_0)$.
Using simple probability, we can write

$$q(x_1, x_2 \dots x_T|x_0) = \frac{q(x_1, x_2 \dots x_0)}{q(x_0)}$$
$$= \frac{q(x_0) \prod_{i=1}^{T} q(x_i|x_0, x_1 \dots x_{i-1})}{q(x_0)}$$

$$\implies q(x_1, x_2 \dots x_T|x_0) = \prod_{i=1}^{T} q(x_i|x_{i-1}) \tag{3}$$

where we obtain the last step due to the properties of a Markov Chain. We choose our Transition Kernel T to be the Gaussian Kernel. Thus,

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t;\ x_{t-1}\sqrt{1-\beta_t},\ \beta_t\mathbf{I}) \tag{4}$$

where, $\beta_i\ ;\ 1 \leq i \leq T$ are pre-determined hyperparameters. Note that the variance of the Gaussian is $\beta_t\mathbf{I}$, resulting in a standard deviation of $\sqrt{\beta_t}\mathbf{I}$.

We can derive the closed analytical form of $q(x_t|x_0)$ using the reparameterization trick:

We can write x_t as

$$x_t = x_{t-1}\sqrt{1-\beta_t} + \sqrt{\beta_t}\epsilon\ ;\ \epsilon \sim \mathcal{N}(0,1)$$

Similarly,

$$x_{t-1} = x_{t-2}\sqrt{1-\beta_{t-1}} + \sqrt{\beta_{t-1}}\epsilon\ ;\ \epsilon \sim \mathcal{N}(0,1)$$

Thus, we can write

$$x_t = x_{t-1}\sqrt{1-\beta_t} + \sqrt{\beta_t}\epsilon_1$$
$$= \left(x_{t-2}\sqrt{1-\beta_{t-1}} + \sqrt{\beta_{t-1}}\epsilon_2\right)\sqrt{1-\beta_t} + \sqrt{\beta_t}\epsilon_1$$
$$= x_{t-2}\sqrt{(1-\beta_{t-1})(1-\beta_t)} + \sqrt{(1-\beta_t)\beta_{t-1}}\epsilon_2 + \sqrt{\beta_t}\epsilon_1$$
$$= x_{t-2}\sqrt{(1-\beta_{t-1})(1-\beta_t)} + \sqrt{(1-\beta_t)\beta_{t-1} + \beta_t}\epsilon_2^*$$

For simplicity, we assume $\alpha_t = 1 - \beta_t$. Also, we know that the sum of two normal distributions with the same mean μ and different variances σ_1^2 and σ_2^2 is given by:

$$\mathcal{N}(\mu, \sigma_1^2) + \mathcal{N}(\mu, \sigma_2^2) = \mathcal{N}(\mu, \sigma_1^2 + \sigma_2^2)$$

Thus, we have

$$x_t = x_{t-2}\sqrt{\alpha_t\alpha_{t-1}} + \sqrt{\alpha_t(1-\alpha_{t-1}) + (1-\alpha_t)}\epsilon_2^*$$
$$= x_{t-2}\sqrt{\alpha_t\alpha_{t-1}} + \sqrt{1-\alpha_t\alpha_{t-1}}\epsilon_2^*$$
$$\vdots$$
$$= x_0\sqrt{\prod_{i=1}^{t}\alpha_i} + \sqrt{1-\prod_{i=1}^{t}\alpha_i}\epsilon$$

Finally, we can show the analytical form of $q(x_t|x_0)$ to be:

$$q(x_t|x_0) = \mathcal{N}(x_t;\ x_0\sqrt{\hat{\alpha}_t}, (1-\hat{\alpha}_t)\mathbf{I}) \tag{5}$$

where $\hat{\alpha}_t = \prod_{s=1}^{t}\alpha_s = \prod_{s=1}^{t}(1-\beta_s)$

What this implies is, to sample x_t, we can use the following:

$$x_t = x_0\sqrt{\hat{\alpha}_t} + \sqrt{(1-\hat{\alpha}_t)}\,\mathcal{N}(0, \mathbf{I}) \tag{6}$$

This has an important implication. If the diffusion rates β are chosen properly to ensure $\hat{\alpha_T} \approx 0$, we can obtain an *isotropic Gaussian* such that $q(x_T) \approx \mathcal{N}(0, \mathbf{I})$. Thus, the forward diffusion process ensures almost all structures within the image are destroyed.

To generate new samples, we can derive an unstructured noise vector from the prior distribution and remove noise from it by running a Markov Chain in the reverse time direction. Specifically, the reverse Markov Chain has prior distribution $\Pr(x_T) = \mathcal{N}(x_T;\ 0, \mathbf{I})$, which is taken because for the forward Markov Chain is constructed such that $q(x_T) \approx \mathcal{N}(0, \mathbf{I})$.

We also have a learnable kernel given by:

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

where θ represents the model parameters, and the mean $\mu_\theta(x_t, t)$ and variance $\Sigma_\theta(x_t, t)$ are parametrized by deep neural nets.

Thus, for the reverse trajectory, we have

$$p_\theta(x_0, x_1, \dots, x_T) = p(x_T)\prod_{t=1}^{T} p(x_{t-1}|x_t) \tag{7}$$

The eventual goal is to learn the transition kernels in the reverse Markov Chain to be very similar to the kernels used in the forward Markov Chain. Since we wish to minimize the distance between two distributions, we shall use the KL divergence joint distributions of the forward Markov Chain and the reverse Markov Chain.

We now wish to find the loss function. The loss function can be understood by understanding the Evidence Lower Bound (ELBO) or the Variational Lower Bound(VLB) as follows:

Let us model our conditional distribution over the latent variables $p(z|x)$ as $q(z)$. Then, we wish to reduce the KL divergence between the two distributions as:

$$
\begin{aligned}
KL[q||p] &= \int q(z) \log \left[\frac{q(z)}{p(z|x)} \right] \, dz \\
&= \int q(z) \log \left[\frac{q(z)p(x)}{p(z,x)} \right] \, dz \\
&= \int q(z) \log \left[\frac{q(z)}{p(z,x)} \right] \, dz + \log p(x) \int q(z) \, dz \\
&= \int q(z) \log \left[\frac{q(z)}{p(z,x)} \right] \, d\theta + \log p(x) \\
&\implies KL[q||p] + ELBO(q) = \log p(x)
\end{aligned}
$$

Since the KL divergence is non-negative, we have the ELBO to be a lower bound on $\log p(x)$, which is called the evidence in Bayesian Literature and likelihood in Frequentist Literature.

Again, to find the loss function for our diffusion model, we can start with the likelihood under the learnt distributions $\log p(x)$. We shall use the following notation:

$$
(x_0, x_1 \ldots, x_T) = x_{0:T})
$$

Thus,

$$
\log p(x) = \log \int p(x_{0:T}) \, dx_{1:T} \tag{8}
$$

$$
= \log \int \frac{p(x_{0:T})q(x_{1:T}|x_0)}{q(x_{1:T}|x_0)} \, dx_{1:T} \tag{9}
$$

$$
= \log \mathbb{E}_{q(x_{1:T}|x_0)} \left[\frac{p(x_{0:T})}{q(x_{1:T}|x_0)} \right] \tag{10}
$$

$$
= \log \mathbb{E}_{q(x_{1:T}|x_0)} \left[\frac{p(x_T) \prod_{t=1}^{T} p(x_{t-1}|x_t)}{\prod_{i=1}^{T} q(x_i|x_{i-1})} \right] \tag{11}
$$

$$
\geq \mathbb{E}_{q(x_{1:T}|x_0)} \left[\log \frac{p(x_T) \prod_{t=1}^{T} p(x_{t-1}|x_t)}{\prod_{i=1}^{T} q(x_i|x_{i-1})} \right] \tag{12}
$$

$$
= \mathbb{E}_{q(x_{1:T}|x_0)} \left[\log \frac{p(x_T)p(x_0|x_1) \prod_{t=1}^{T-1} p(x_t|x_{t+1})}{q(x_T|x_{T-1}) \prod_{i=1}^{T-1} q(x_i|x_{i-1})} \right] \tag{13}
$$

$$
= \mathbb{E}_{q(x_{1:T}|x_0)} \left[\log \frac{p(x_T)p(x_0|x_1)}{q(x_T|x_{T-1})} \right]
$$

$$
+ \mathbb{E}_{q(x_{1:T}|x_0)} \left[\log \frac{\prod_{t=1}^{T-1} p(x_t|x_{t+1})}{\prod_{i=1}^{T-1} q(x_i|x_{i-1})} \right] \tag{14}
$$

where 8 is the result of x_0 being generated from the reverse Markov Chain being our main image of interest. 12 comes about from using the Jensen's Inequality and 13 uses re-indexing of variables under the product. We can now split the two terms as:

$$
\begin{aligned}
&\mathbb{E}_{q(x_{1:T}|x_0)} \left[\log \frac{p(x_T)p(x_0|x_1)}{q(x_T|x_{T-1})} \right] \\
&= \mathbb{E}_{q(x_1|x_0)} \left[\log p(x_0|x_1) \right] + KL[q(x_{T-1}, x_T|x_0)||p(x_T)]
\end{aligned}
$$

where the first term $\mathbb{E}_{q(x_1|x_0)} \left[\log p(x_0|x_1) \right]$ is called the *reconstruction term*, which predicts the probability of the original sample given the first denoised image.

The second term $KL[q(x_{T-1}, x_T|x_0)||p(x_T)]$ is the KL divergence between the final latent distribution and the Gaussian prior. Note that there is no trainable term here, and effectively becomes zero as soon as we assume $T \to \infty$. This term is called the *prior matching term*.

The second term from 14 can be written as:

$$
\begin{aligned}
&\mathbb{E}_{q(x_{1:T}|x_0)} \left[\log \frac{\prod_{t=1}^{T-1} p(x_t|x_{t+1})}{\prod_{i=1}^{T-1} q(x_i|x_{i-1})} \right] \\
&= \sum_{i=1}^{T-1} \mathbb{E}_{q(x_{T-1}, x_T, x_{T+1}|x_0)} \left[\log \frac{p(x_t|x_{t+1})}{q(x_t|x_{t-1})} \right] \\
&= -\mathbb{E}_{q(x_{t-1}, x_{t+1}|x_0)} \, KL[q(x_t|x_{t-1})||p(x_t|x_{t+1})]
\end{aligned}
$$

This term is the *consistency term* which ensures that the distribution at x_t in both forward and reverse directions is consistent, i.e the denoising step from the reverse Markov chain should match the noising step in the forward chain.

Ho et al. (2020) proposed reweighting certain terms to obtain the following loss formulation

$$
\mathbb{E}_{t \sim \mathcal{U}[1,T], x_0 \sim q(x_0), \epsilon \sim \mathcal{N}(0,\mathbf{I})} \left[\lambda(t) ||\epsilon - \epsilon_\theta(x_t, t)||^2 \right]
$$

where $\lambda(t)$ is a positive weighting function and ϵ_θ is a deep neural network with parameter θ which predicts noise vector ϵ given x_t and t.

Ho et al. (2020) also give many more mathematical insights, such as those related to sampling and many more, the mathematical proofs of which we shall skip.

Ho et al. (2020) shows that μ_θ predicts $\frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\hat{\alpha}_t}} \epsilon \right)$. Thus, we can sample $x_{t-1} \sim p_\theta(x_{t-1}|x_t)$ by using $x_{t-1} = \mu_\theta + \sigma_t z$ where $z \sim \mathcal{N}(0, \mathbf{I})$. The entire sampling algorithm is given in Algorithm 1

4. So how does DALL-E 2 work?

DALL-E 2 is the amalgamation of multiple models made by Open AI. DALL-E 2 first takes the caption and generates text embeddings from them using a model called CLIP

Algorithm 1 Sampling Algorithm

1 . $x_T \sim \mathcal{N}(0, \mathbf{I})$
2. for $t = T, \ldots, 1$ do
3. $\quad z \sim \mathcal{N}(0, \mathbf{I})$ if $t > 1$ else $z = 0$
4. $\quad x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\hat{\alpha}_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t z$
5. return x_0

(Contrastive Language-Image Pre-training). These text embeddings are then passed through a "prior." This prior is a diffusion model. Finally, the output of the prior as well as the text embeddings are passed to the decoder, which is Open AI's famous model GLIDE(Guided Language to Image Diffusion for Generation and Editing). The final image is then upsampled from 64×64 to 1024×1024 to get the final result.

Here, we shall skip the details of the CLIP model. In brief, the model is trained by generating embeddings from the caption and the text and minimizing the cosine distance between the two.

Diffusion is one of the most important concepts going into the prior and the decoder GLIDE. The prior is crucial because this is where most variations are generated. GLIDE simply goes above and beyond and uses both text and image embeddings to generate an image.

5. Then what is Stable Diffusion?

Stable diffusion helps overcome a major computational drawback of the Diffusion Process by shifting the process from the image space to the pixel space. It does so by compressing the image into a latent space and then restoring the image from the latent space. So, instead of generating a noisy image, it shall generate a random tensor in the latent space and corrupt it with noise. Finally, the decoder aims to convert this latent space representation to the normal image space.

6. Results

We first implemented the forward Markov Chain and verified the analytical formula for obtaining any indexed image. The results are shown in 1:

We have also implemented different schedulers: cosine, linear, quadratic, and sigmoid. The results are shown in 2:

We find the linear scheduler to be the best. This is because even though the sigmoid scheduler gives us a better value of α_T, it is also difficult to compute. We also implemented the basic training of a network and used it for some grayscale images, which is shown in the figures 3 , 4 , 5:

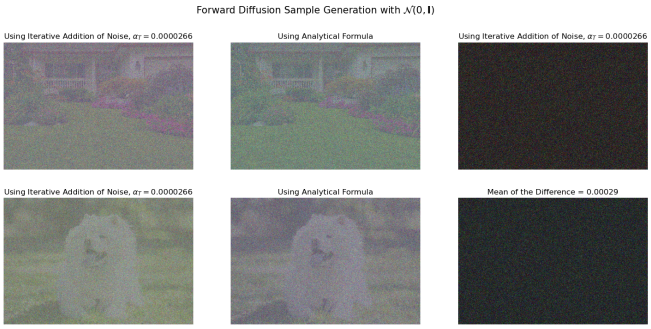

Figure 1. Forward generation

Figure 2. Different kind of schedulers

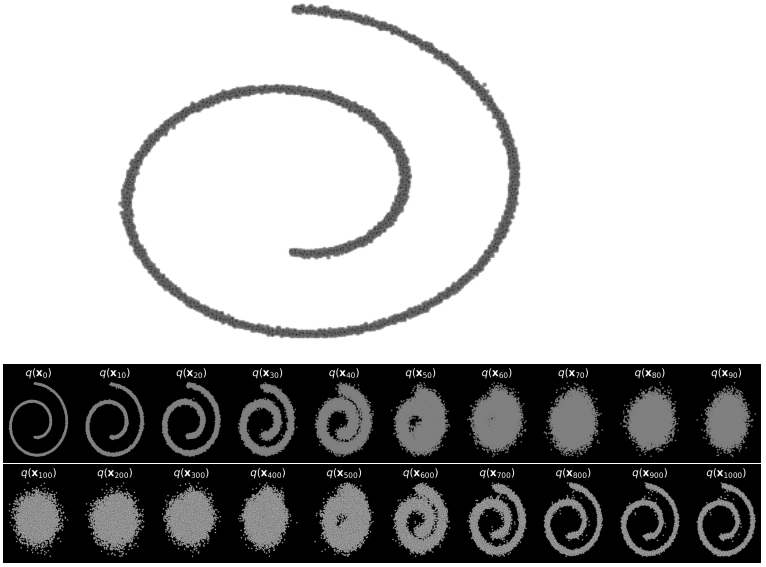

Figure 3. The original image of the spiral and the forward and reverse Markov Chains

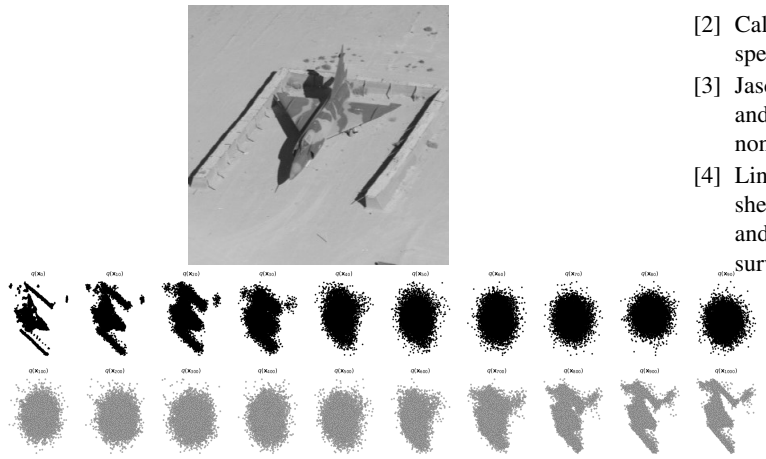

Figure 4. The original image of the plane and the forward and reverse Markov Chains

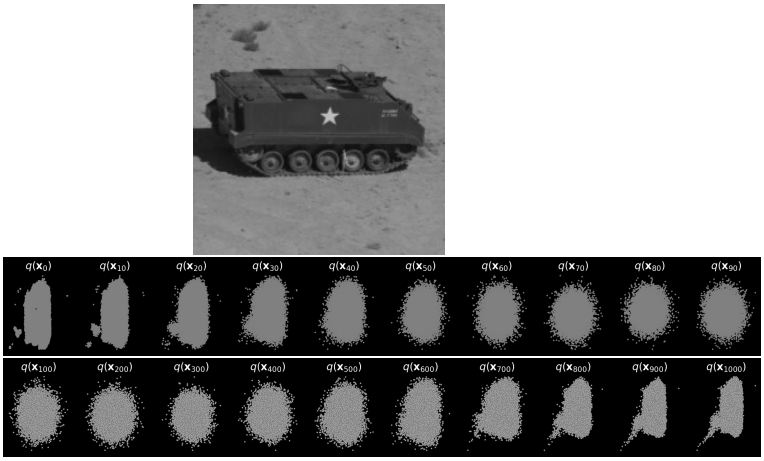

Figure 5. The original image of the tank and the forward and reverse Markov Chains

7. Conclusion

This project aimed to break down the science behind Diffusion Models and aims to explain the mathematics behind them. Due to its use in numerous applications in some of the top-achieving models in AI, it is one of the most sought-after models. However, with Deep Learning and Machine Learning progressing at the rates of knots, how long this model remains relevant is a question. Moreover, the interdisciplinary approach taken in the concept behind the model is evidence of ties between engineering fields that can only become stronger with time.

References

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models.

[2] Calvin Luo. Understanding diffusion models: A unified perspective.

[3] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics.

[4] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Yingxia Shao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications.