✨ Member-only story

Diffusion Models Made Easy

Understanding the Basics of Denoising Diffusion Probabilistic Models

J. Rafid Siddiqui, PhD · Follow

Published in Towards Data Science

7 min read · May 2, 2022

Listen Share ••• More

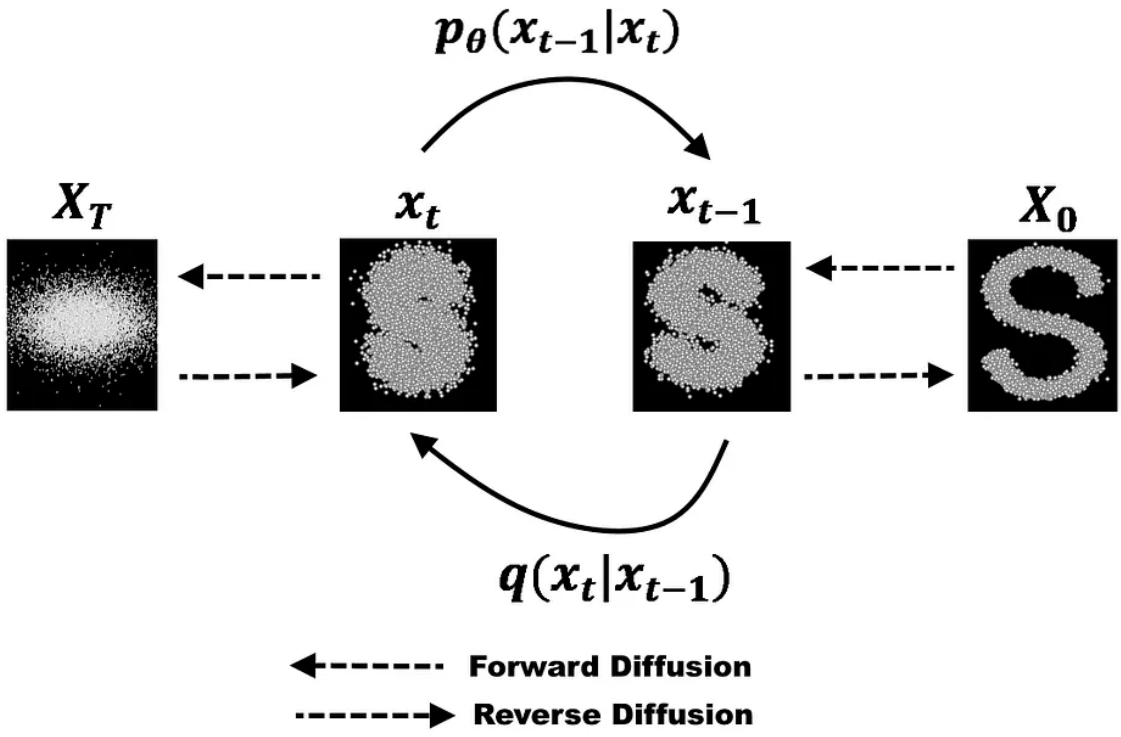


Figure 1: Process of Denoising Diffusion Probabilistic Model (Image by author)

1. Introduction

In the recent past, I have talked about GANs and VAEs as two important Generative Models that have found a lot of success and recognition. GANs work great for multiple applications however, they are difficult to train, and their output lack

diversity due to several challenges such as mode collapse and vanishing gradients to name a few. Although VAEs have the most solid theoretical foundation however, the modelling of a good loss function is a challenge in VAEs which makes their output to be suboptimal.

There is another set of techniques which originate from probabilistic likelihood estimation methods and take inspiration from physical phenomenon; it is called, Diffusion Models. The central idea behind Diffusion Models comes from the thermodynamics of gas molecules whereby the molecules diffuse from high density to low density areas. This movement is often referred in physics literature as the increase of entropy or heat death. In information theory, this equates to loss of information due to gradual intervention of noise.

The key concept in Diffusion Modelling is that if we could build a learning model which can learn the systematic decay of information due to noise, then it should be possible to reverse the process and therefore, recover the information back from the noise. This concept is similar to VAEs in the way that it tries to optimize an objective function by first projecting the data onto the latent space and then recovering it back to the initial state. However, instead of learning the data distribution, the system aims to model a series of noise distributions in a *Markov Chain* and "decodes" the data by undoing/denoising the data in a hierarchical fashion.

2. Denoising Diffusion Model

The idea of denoising diffusion model has been around for a long time. It has its roots in Diffusion Maps concept which is one of the dimensionality reduction techniques used in Machine Learning literature. It also borrows concepts from the probabilistic methods such as *Markov Chains* which has been used in many applications. The original Denoising Diffusion method was proposed in *Sohl-Dickstein et al.* [1].

A denoising diffusion modeling is a two step process: the forward diffusion process and the reverse process or the reconstruction. In the forward diffusion process, gaussian noise is introduced successively until the data becomes all noise. The reverse/ reconstruction process undoes the noise by learning the conditional probability densities using a neural network model. An example depiction of such a process can be visualized in Figure 1.

3. Forward Process

We can formally define the forward diffusion process as a *Markov Chain* and therefore, unlike an encoder in the VAEs it doesn't require a training. Starting with the initial data point , we add Gaussian noise for ***T*** successive steps, and obtain a set of noisy samples. The prediction of probability density at time ***t*** is only dependent on the immediate predecessor at time ***t-1*** and therefore, the conditional probability density can be computed as follows:

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

The complete distribution of the whole process can then be computed as follows:

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

Here, the mean and variance of the density function depends on a parameter βτ, which is a hyper parameter whose value can either be taken as a constant throughout the process or can be gradually changed in the successive steps. For a differential parameter value assignment, there can be range of function that can be used to model the behavior (e.g. sigmoid, tanh, linear etc.).

The above derivation is enough to predict the successive states, however, if we would like to sample at any given time interval ***t*** without going through all the intermediary steps, therefore, allowing an efficient implementation, then we can re-formulate the above equation by substituting the hyper-parameter as ατ = 1 — βτ. The reformulation of the above then becomes:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t; \sqrt{\overline{\alpha_t}}\boldsymbol{x}_0, (1-\overline{\boldsymbol{\alpha}_t})\boldsymbol{I}\right)$$

In order to produce samples at a time step ***t*** with probability density estimation available at time step ***t-1,*** we can employ another concept from thermodynamics called, '*Langevin dynamics'*. According to *stochastic gradient Langevin dynamics* [2] we can sample the new states of the system only by the gradient of density function in a *Markov Chain* updates. The sampling of a new data point at time ***t*** for a step size **ε** based on previous point at time ***t-1*** can then be computed as follows:

$$x_t = x_{t-1} + \frac{\epsilon}{2}\nabla_x p(x_{t-1}) + \sqrt{\epsilon} z_i \qquad z_i \sim \mathcal{N}(0, 1)$$

4. Reconstruction

The reverse process requires the estimation of probability density at an earlier time step given the current state of the system. This means estimating the **q**(χτ-1 | χτ) when *t*=*T* and thereby generating data sample from isotropic Gaussian noise. However, unlike the forward process, the estimation of previous state from the current state requires the knowledge of all the previous gradients which we can't obtain without having a learning model that can predict such estimates. Therefore, we shall have to train a neural network model that estimates the **ρθ**(χτ-1|χτ) based on learned weights θ and the current state at time ***t***. This can be estimated as follows:

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

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

The parameterization for mean function were proposed by *Ho. et al.* [3] and can be computed as follows:

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{a_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)\right)$$

The authors in *Ho. et al.* [3] suggested to use a fixed variance function as Σθ = βτ. The sample at time ***t-1*** can then be computed as follows:

$$x_{t-1} = \frac{1}{\sqrt{a_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t z$$

5. Training and Results

5.1. Construction of the Model

The model used in the training for diffusion model follows the similar patterns to a VAE network however, it is often kept much simpler and straight-forward compared

to other network architectures. The input layer has the same input size as that of the data dimensions. There can be multiple hidden layers depending on the depth of the network requirements. The middle layers are linear layers with respective activation functions. The final layer is again of the same size as that of the original input layer, thereby reconstructing the original data. In the *Denoising Diffusion Networks* the final layer consists of two separate outputs, each dedicated to the mean and variance of the predicted probability density respectively.

5.2. Computation of Loss Function

The objective of the network model is to optimize the following loss function:

$$\mathcal{L} = E_q\left(-log p(x_T) - \sum_{t\geq 1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right)$$

A reduced form of this loss function was proposed in *Sohl-Dickstein et al.*[1] that formulates the loss in terms of a linear combination of KL-divergence between two gaussian distributions and a set of entropies. This simplifies the computation and makes it easy to implement the loss function. The loss function then becomes:

$$\mathcal{L} = -E_q\left(D_{KL}(q(x_{t-1}|x_t, x_0) \,||\, p_\theta(x_{t-1}|x_t)) + H_q(x_T|x_0) - H_q(x_1|x_0) - H_p(x_T)\right)$$

A further simplification and improvement was proposed by *Ho et al.* [3] in the loss function whereby the parameterization for the mean is used as described in the previous section for the forward process. Therefore, the loss function then becomes:

$$\mathcal{L}_{simple} = -E_{t,x_0,\epsilon}\left(\left\|\epsilon - \epsilon_\theta\left(\sqrt{\overline{\alpha}_t}x_0 + \sqrt{1-\overline{\alpha}_t}\epsilon, t\right)\right\|^2\right)$$

5.3. Results

The results for the forward process which adds Gaussian noise by following a *Markov Chain* can be seen in the following figure. The total number of time steps were 100 while this figure shows 10 samples from the generated set of sequences.

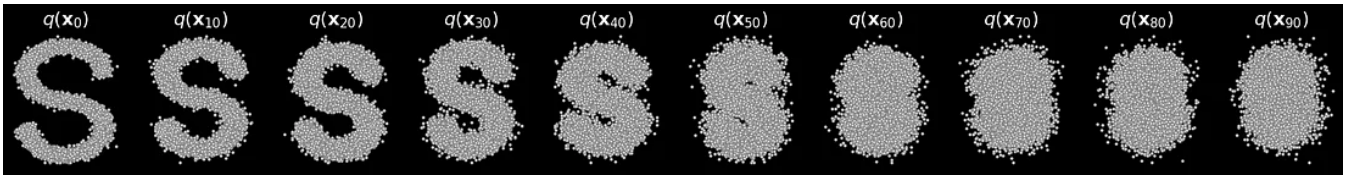


Figure 2: Results of a forward Diffusion process on synthetic dataset of S-Curve (Image by author)

The results for the reverse diffusion process can be seen in the following figure. The quality of the final output depends on the tuning of the hyper-parameters and number of training epochs.

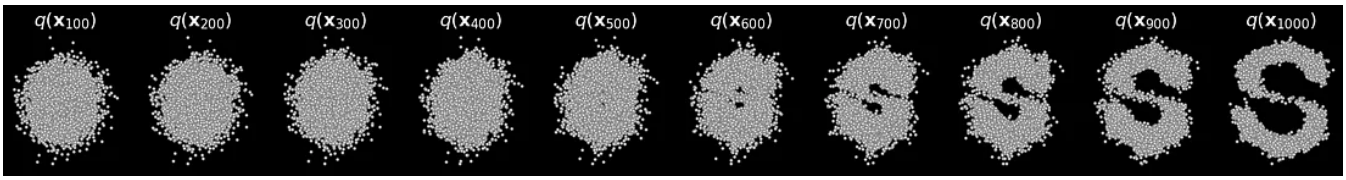


Figure 3: Results of reconstruction of data from isotropic Gaussian noise (Image by author)

Figure 4: Results of Denoising Diffusion Model on 3 different datasets: Swiss Roll, The Moon and The S-Curve (Image by author)

6. Conclusions

In this article, we have discussed the basics of Diffusion Models, and their implementation. Although Diffusion Models are computationally more expensive than other deep network architectures, however, they perform much better in certain applications. For example, recent applications to text and image synthesis tasks, Diffusion Models have out-performed over other architectures [4]. Further implementation details and code can be found at the following github repository: https://github.com/azad-academy/denoising-diffusion-model.git

Subscribe & Follow for further updates: azad-wolf.medium.com/

References

[1] Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N., & Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. arXiv preprint arXiv:1503.03585.

[2] Max Welling & Yee Whye Teh. "Bayesian learning via stochastic gradient langevin dynamics." ICML 2011.

[3] Ho, J., Jain, A., & Abbeel, P. (2020). *Denoising diffusion probabilistic models*. arXiv preprint arXiv:2006.11239.

[4] Prafulla Dhariwal, Alex Nichol, *Diffusion Models Beat GANs on Image Synthesis*, arXiv: 2105.05233

Deep Learning | Machine Learning | Data Science | Neural Networks | Deep Generative Model

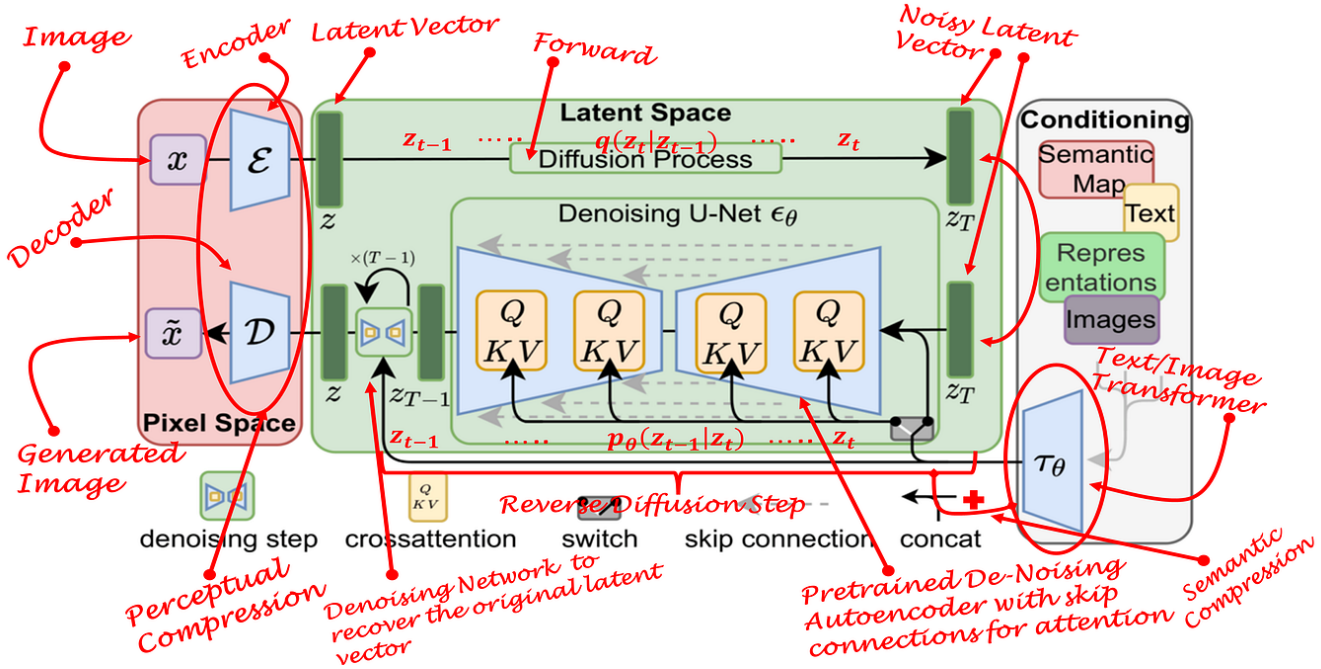
Follow

Written by J. Rafid Siddiqui, PhD

1K Followers · Writer for Towards Data Science

AI Research Scientist, Educator, and Innovator.Writes about Deep learning, Computer Vision, Machine Learning, AI, & Philosophy. bit.ly/MLMethodsBook

More from J. Rafid Siddiqui, PhD and Towards Data Science



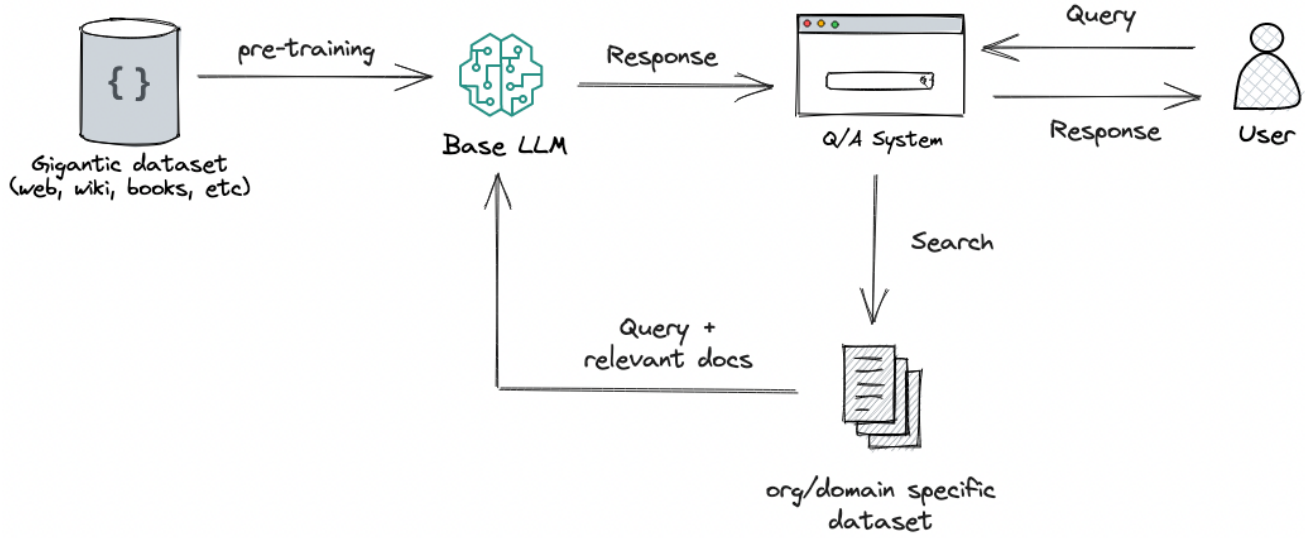
J. Rafid Siddiqui, PhD in Towards Data Science

What are Stable Diffusion Models and Why are they a Step Forward for Image Generation?

An Easy Guide to Latent Diffusion Models

✨ · 8 min read · Sep 20, 2022

249 2



Heiko Hotz in Towards Data Science

RAG vs Finetuning—Which Is the Best Tool to Boost Your LLM Application?

The definitive guide for choosing the right method for your use case

✦ · 19 min read · Aug 25

1.8K 16

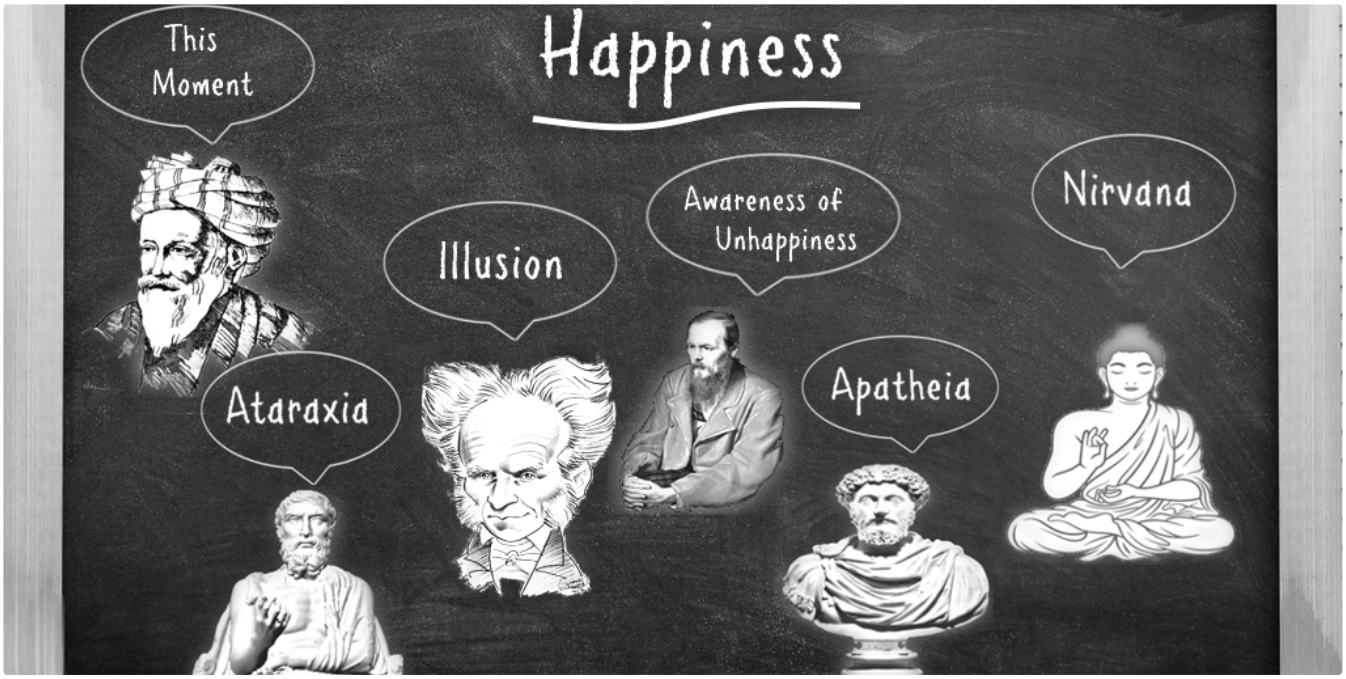
Cameron R. Wolfe, Ph.D. in Towards Data Science

Advanced Prompt Engineering

What to do when few-shot learning isn't enough...

✦ · 17 min read · Aug 7

1.2K 9



J. Rafid Siddiqui, PhD in ILLUMINATION

The Myth of Happiness: What is the Eastern and Western Philosophy of Happiness

Exploring the Epicureanism, Stoicism, Pessimism, and Spirituality for Understanding the Meaning of Happiness

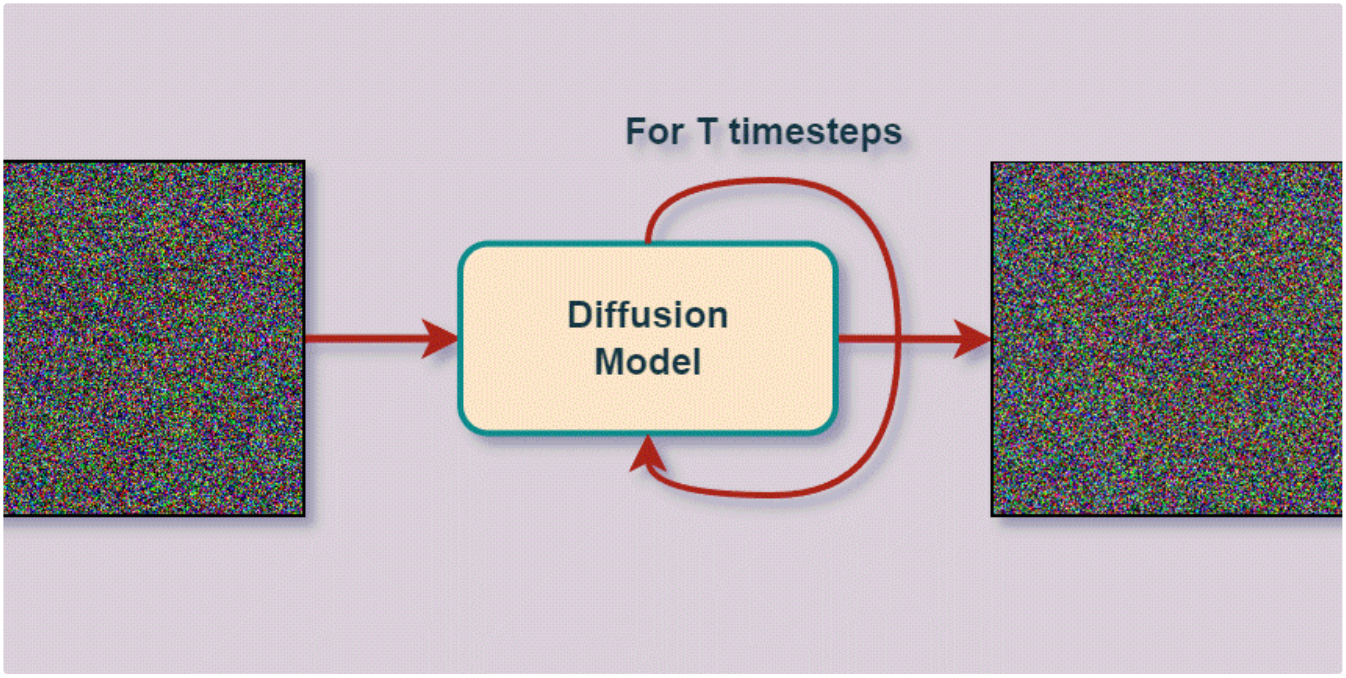
· 10 min read · Jul 1

986 21

See all from J. Rafid Siddiqui, PhD

See all from Towards Data Science

Recommended from Medium

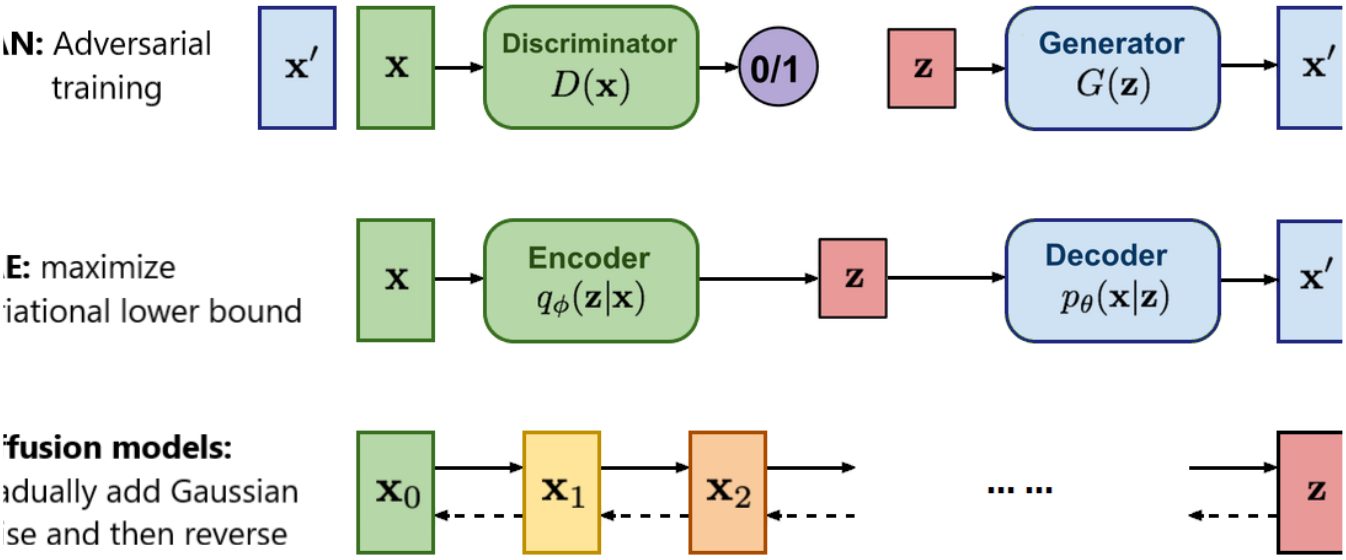


Gabriel Mongaras in Better Programming

Diffusion Models—DDPMs, DDIMs, and Classifier Free Guidance

A guide to the evolution of diffusion models from DDPMs to Classifier Free guidance

28 min read · Mar 13



Ainur Gainetdinov in Towards AI

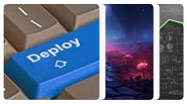
Diffusion Models vs GANs vs VAEs: Comparison of Deep Generative Models

Deep generative models are applied to diverse domains such as image, audio, video synthesis, and natural language processing. With the...

6 min read · May 12

208 2

Lists



Predictive Modeling w/ Python

20 stories · 381 saves

Practical Guides to Machine Learning

10 stories · 424 saves

Natural Language Processing

604 stories · 218 saves

New_Reading_List

174 stories · 104 saves

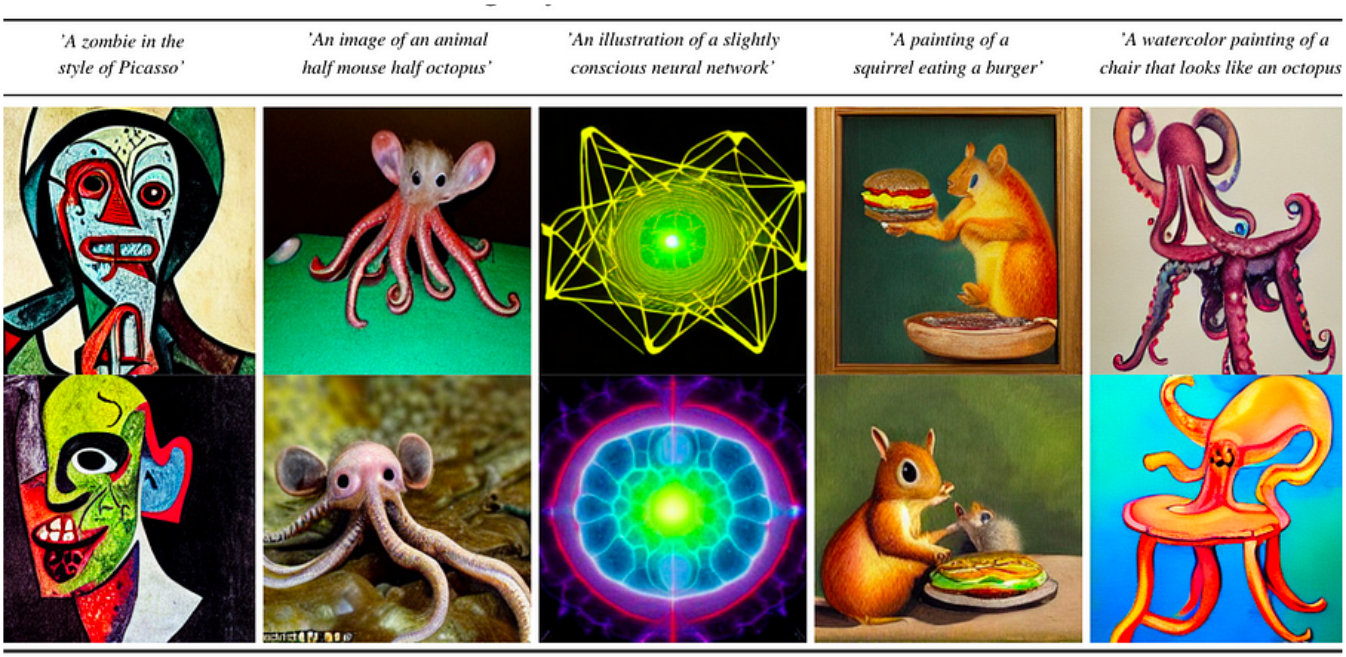
Joseph Rocca in Towards Data Science

Understanding Diffusion Probabilistic Models (DPMs)

Building, step by step, the reasoning that leads to DPMs.

23 min read · Dec 6, 2022

615 1



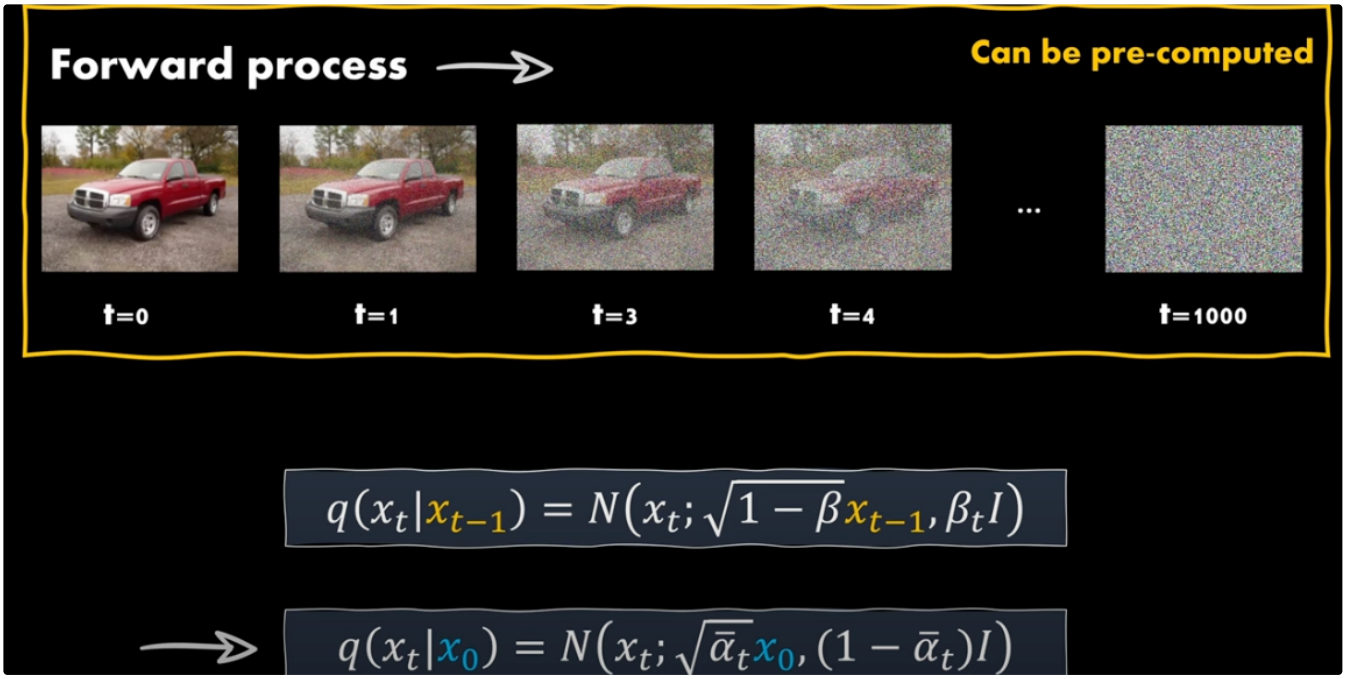
Onkar Mishra

Stable Diffusion Explained

How does Stable diffusion work? Explaining the tech behind text to image generation.

6 min read · Jun 8

336 3



Sunshine

Diffusion Model

Diffusion Model relative resources: https://github.com/heejkoo/Awesome-Diffusion-Models

2 min read · May 25

Altynanke in Data Analysis Center

Convolutions and Self-Attention or There and Back Again

It takes courage and decent level of madness to embark on a journey towards details of "who wore it better". Buckle up, it's a nasty trip!

10 min read · Mar 27

252

See more recommendations