

Primer on Generative Modelling and Diffusion

By Rohit Satishkumar

Math Prerequisites

Probability and Statistics :

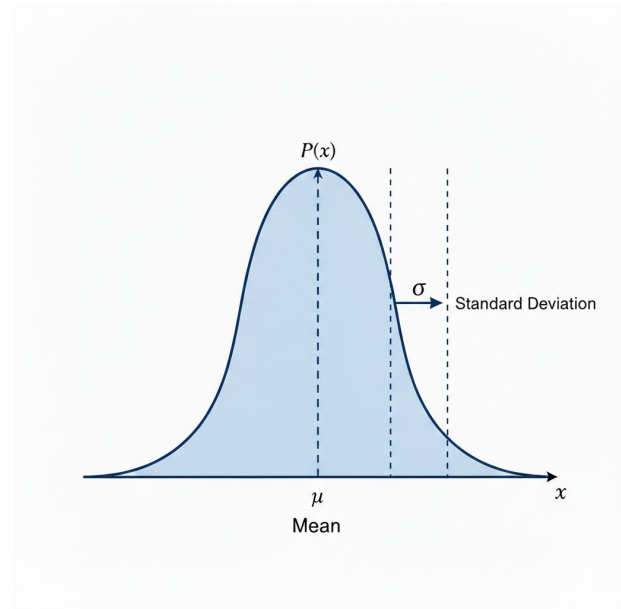
If X and Y are 2 random variables, then

- joint distribution - $p(X,Y)$
- Marginal distributions - $p(X)$ and $p(Y)$
- Conditional distribution - $p(X|Y)$ or $p(Y|X)$

Bayes theorem - $p(X|Y) = p(X,Y) / p(Y) = p(Y|X)p(X)/p(Y)$

Gaussian Distribution :

Parametrized through mean (μ) and variance(σ)

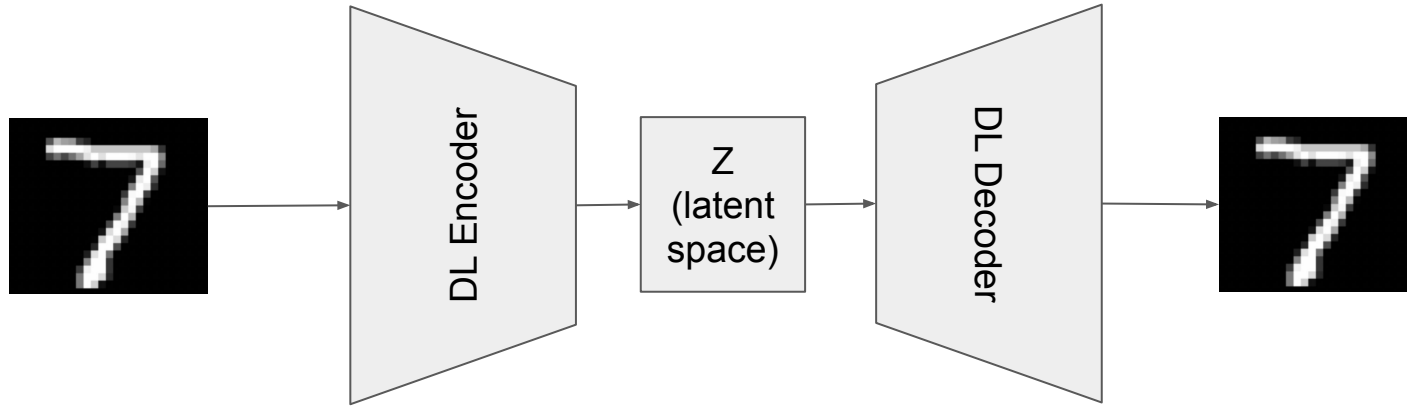


Generative Modelling Objective

Given a dataset $\{x\}$ -

- We want to model a distribution $p_{\theta}(x)$ where $p_{\theta}(x)$ is parametrized by a deep learning model, and $p_{\theta}(x)$ is very similar to original data distribution.
- By sampling from this distribution, we will be able to “generate” new data (an image, trajectory etc.) which has the properties of the original data distribution (We can't generate an image of a dog by sampling from a cat dataset distribution)
- Assumption made - Data Points in a dataset are part of a probability distribution.

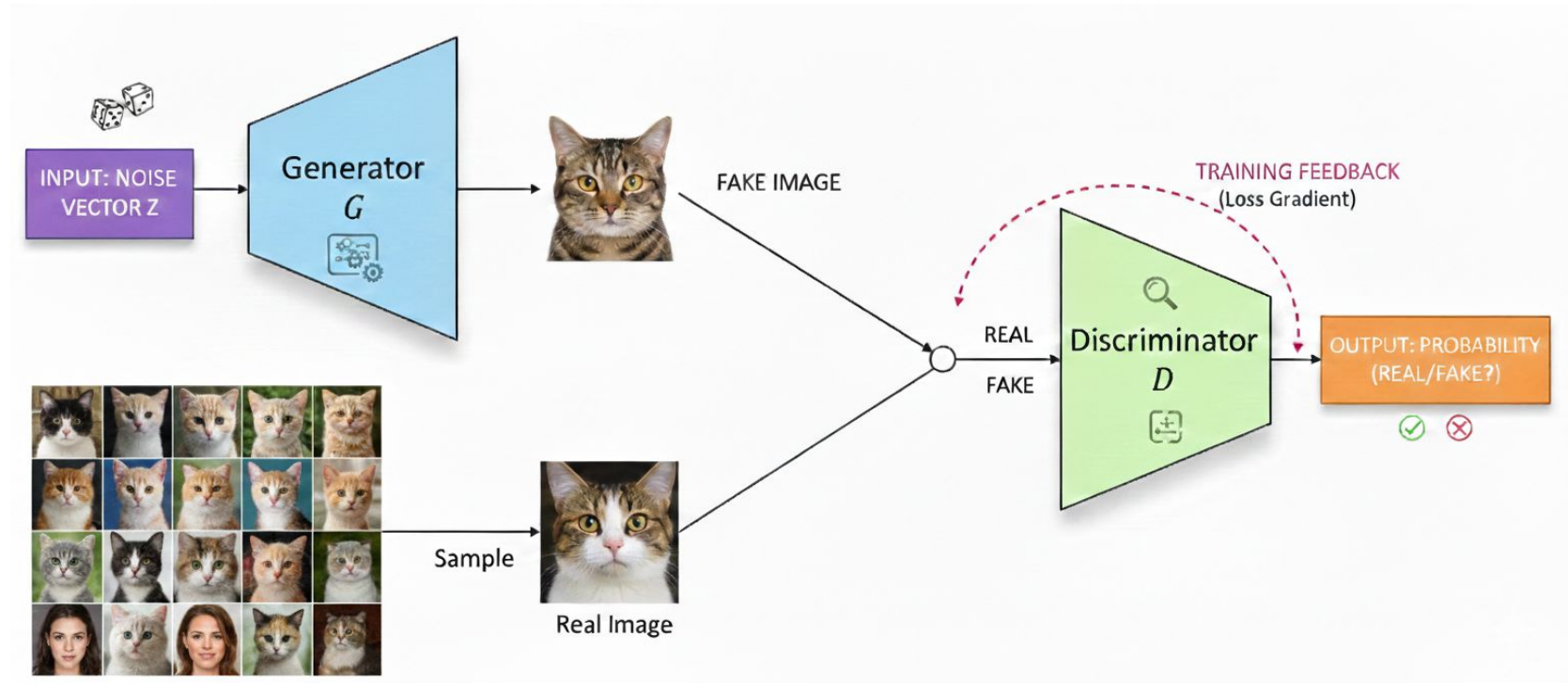
Previous Generative Models - VAE's



Latent space has lower dimensions than the data. We typically sample from a gaussian distribution to use the decoder for generation

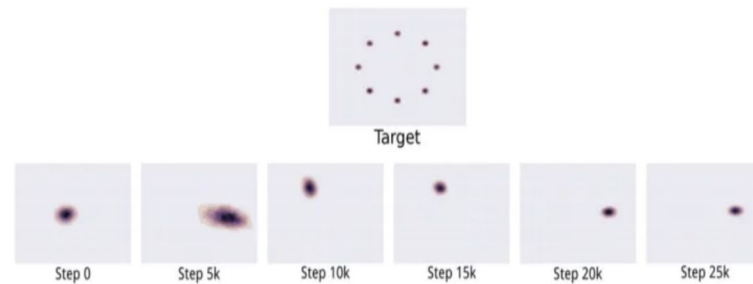
VAE's try to maximize the log likelihood of the original data by maximizing ELBO

Previous Generative Models - GAN's



Problems with VAE's and GAN's

- VAE's tend to output blurry images - this occurs because of the inherent gaussian assumptions we make about the encoder and decoder.
- GAN's are hard to optimize because of contrasting objectives of the Generator and Discriminator - leads to mode collapse where all images generated tend to look very similar.



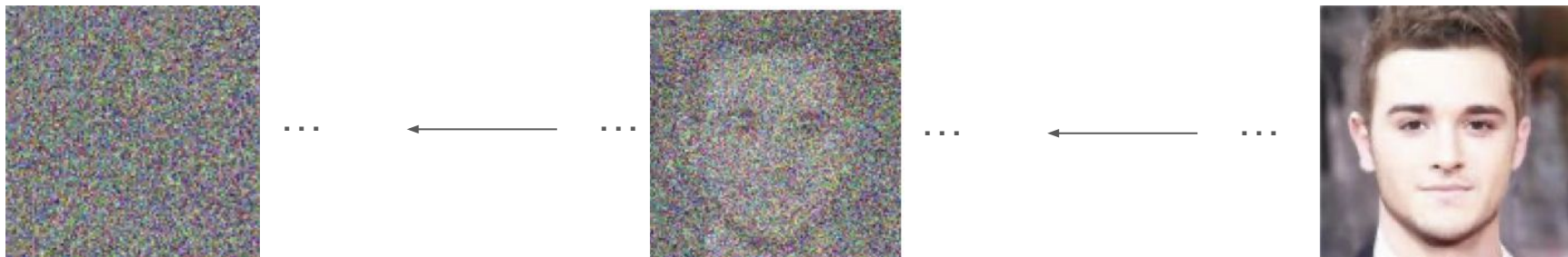
Trying to Generate images from noise in a single step creates problems

Diffusion Overview

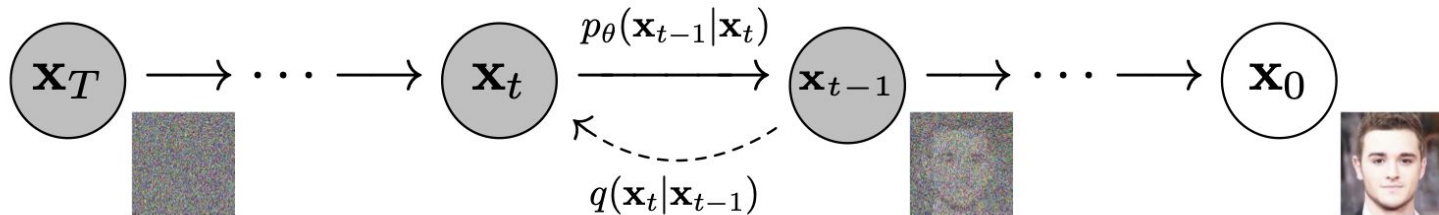
Forward Process



Reverse Process



Forward and Reverse Processes of Diffusion



- Forward Process: (Fixed)

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) := \prod_{t=1}^T q(\mathbf{x}_t|\mathbf{x}_{t-1}), \quad q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \quad (\beta \text{ is fixed})$$

- Reverse Process: (Parametrized By Deep Learning Model)

$$p_\theta(\mathbf{x}_{0:T}) := p(\mathbf{x}_T) \prod_{t=1}^T p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \quad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

- Assumption - Both processes follow the Markov Property
- Direct Formula to go to any timestep from base data(\mathbf{x}_0) $\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}$

Diffusion Objective for DDPM - ELBO

We try to increase the log likelihood for data from the dataset - $\log p_\theta(x_0)$

$$\begin{aligned} p_\theta(\mathbf{x}_0) &:= \int p_\theta(\mathbf{x}_{0:T}) d\mathbf{x}_{1:T}, \\ \log p_\theta(x_0) &= \log \int p_\theta(x_{0:T}) dx_{1:T} \\ &= \log \int q(x_{1:T}|x_0) \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)} dx_{1:T} \\ &= \log \mathbb{E}_{q(x_{1:T}|x_0)} \left[\frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)} \right] \\ &\geq \mathbb{E}_{q(x_{1:T}|x_0)} \left[\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)} \right] \end{aligned} \quad \xrightarrow{\text{Jensen's Inequality}}$$

Diffusion objective - continued

$$\begin{aligned}
 L &= \mathbb{E}_q \left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \right] \\
 &= \mathbb{E}_q \left[-\log p(\mathbf{x}_T) - \sum_{t \geq 1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})} \right] \\
 &= \mathbb{E}_q \left[-\log p(\mathbf{x}_T) - \sum_{t > 1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})} - \log \frac{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)} \right] \\
 &= \mathbb{E}_q \left[-\log p(\mathbf{x}_T) - \sum_{t > 1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} \cdot \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} - \log \frac{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)} \right] \\
 &= \mathbb{E}_q \left[-\log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T|\mathbf{x}_0)} - \sum_{t > 1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1) \right] \\
 &= \mathbb{E}_q \left[D_{\text{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \| p(\mathbf{x}_T)) + \sum_{t > 1} D_{\text{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \| p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1) \right]
 \end{aligned}$$

Loss based on ELBO

Bunch of math using markov property and Bayes theorem

Reconstruction term
 $\propto \|\mathbf{x}_0 - \hat{\mathbf{x}}_\theta(\mathbf{x}_1, 1)\|^2$
 $\propto \|\epsilon - \epsilon_\theta(\mathbf{x}_1, 1)\|^2$

Prior matching term - constant

KL matching terms
 $D_{\text{KL}}(q \| p_\theta) \propto \|\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \mu_\theta(\mathbf{x}_t, t)\|^2 \propto \|\epsilon - \epsilon_\theta(\mathbf{x}_t, t)\|^2$

$L \propto \|\epsilon - \epsilon_\theta\|^2$

DDPM Loss function

DDPM Algorithm - Training and Sampling

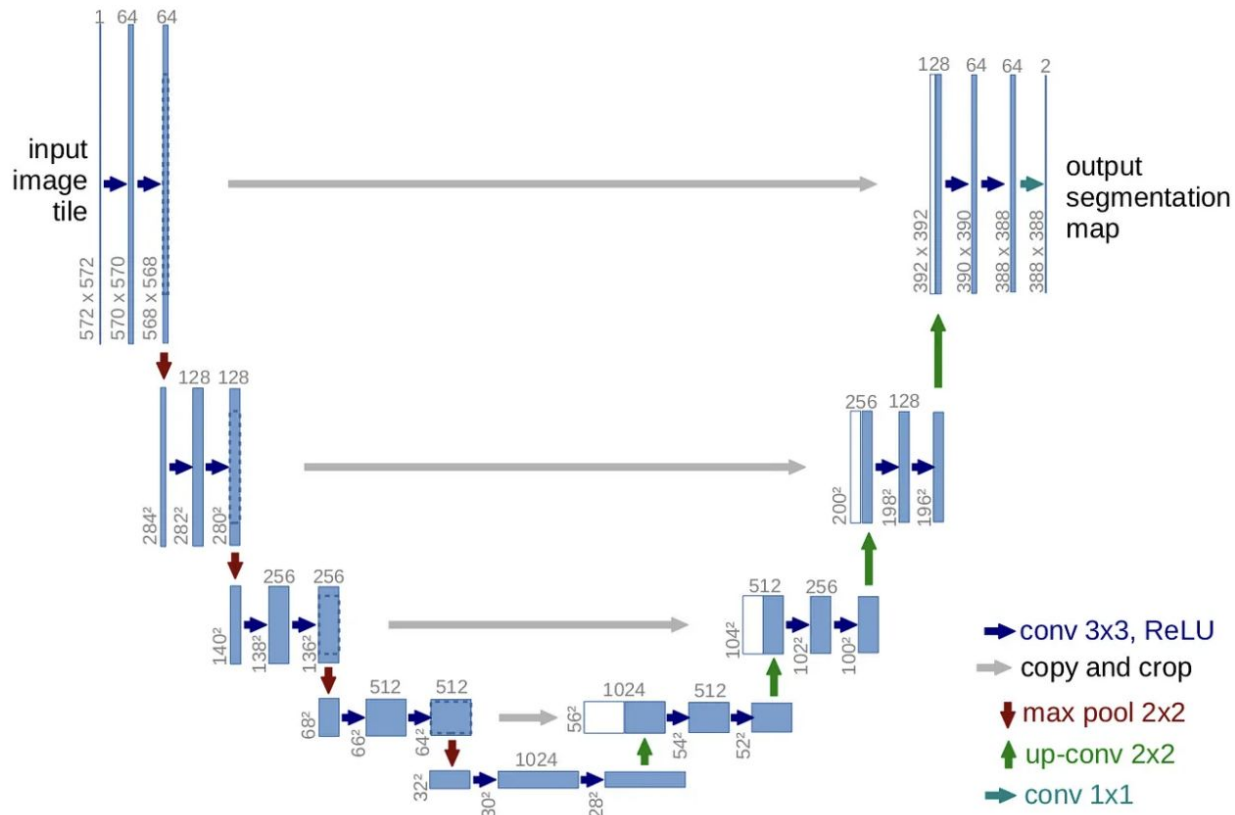
Algorithm 1 Training

- 1: **repeat**
 - 2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
 - 3: $t \sim \text{Uniform}(\{1, \dots, T\})$
 - 4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
 - 5: Take gradient descent step on
$$\nabla_{\theta} \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\theta}(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}, t) \right\|^2$$
 - 6: **until** converged
-

Algorithm 2 Sampling

- 1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
 - 2: **for** $t = T, \dots, 1$ **do**
 - 3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
 - 4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \boldsymbol{\epsilon}_{\theta}(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
 - 5: **end for**
 - 6: **return** \mathbf{x}_0
-

Model Architecture - UNet



Why Diffusion supports multimodality ?

- By learning to separate noise from clean signals, the model implicitly learns the gradient of the log likelihood function at every timestep t .
- Log likelihood function is the manifold containing the data distribution
- Since we condition the model on t and we take multiple timesteps to transition from the unimodal gaussian to the original data distribution, the model learns the landscape very well.
- This is unlike GAN's and VAE's which try to convert noise to a clean signal in a single step, which leads to problems.

Problems with DDPM - and improvements

Problems -

- Very slow sampling (~1000 steps), can't reduce it.
- High compute requirements

Some Improvements -

- Make stochastic forward and reverse processes deterministic → DDIM
- Enable inference and training in a latent space → Stable Diffusion

THANK YOU