Stable Diff from Scratch:

OUTLINE:

- Latent Diff Models (Stable Diffusion)
  Maths behind it
  Classifier free Guidance
  Text to image.
  Img to img.
  Inpainting.

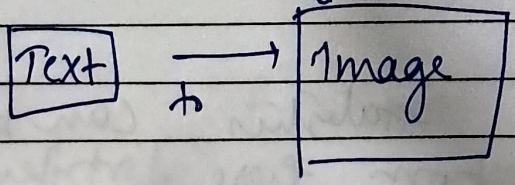
- Prereq:

  (i) Basics of Prob. & statistics (multivariate, Gaussian, conditional prob, marg. prob, Bayer's rule).

  (ii) Basics of Pytorch & NN.
  (iii) Attention Mechanism.
  (iv) How convolutional layers work.

⇒ Stable Diffusion:

↳ introduced by CompVis Group in 2022.



Deep Learning Model.

– we will also do img to img (chges in img if req) & also inpainting chges to do in images.

∃ Generative Models:

- learn from a probability distribution of the data set such that new instances of data are generated by sampling from it.

- eg: a large sample of cats present from them new samples can be created

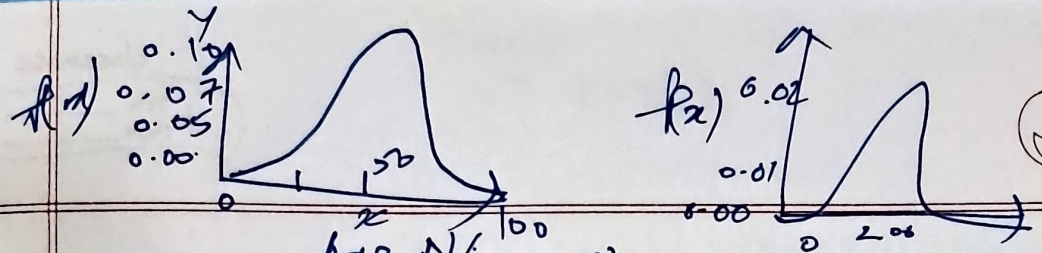
- Why do we model data as distributions? (probability?)

↳ imagine you are a criminal, trying to create fake identities.

every fake identity is made of variables representing characteristics of person (age, height).

Age & height statistics can be used to sample from these distributions to create a fake identities.

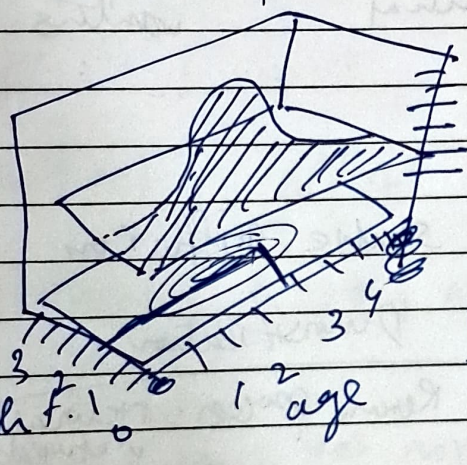
→ sample from distrib. means to throw a coin that gives a value having high probability.



Age $N(40, 30^2)$ Height $N(120, 100^2)$

- While producing fake identities let's say. foreg: assume age = 3, & Height = 130 cm which is unlikely.

To get reasonable o/p results, we can use joint prob. distributions represented as



$P(X, Y)$ represents the prob. of a certain (X, Y) combi to occur.

(*) To model something we need joint distributions of variables.

(*) This is done in images, as well for pixels.

(*) Marginalisation: for just Prob. of occur. of X; regardless of Y.

$$P(x) = \int P(x|y) P(y) dy.$$

or

$$P(x) = \int P(x, y) dy.$$

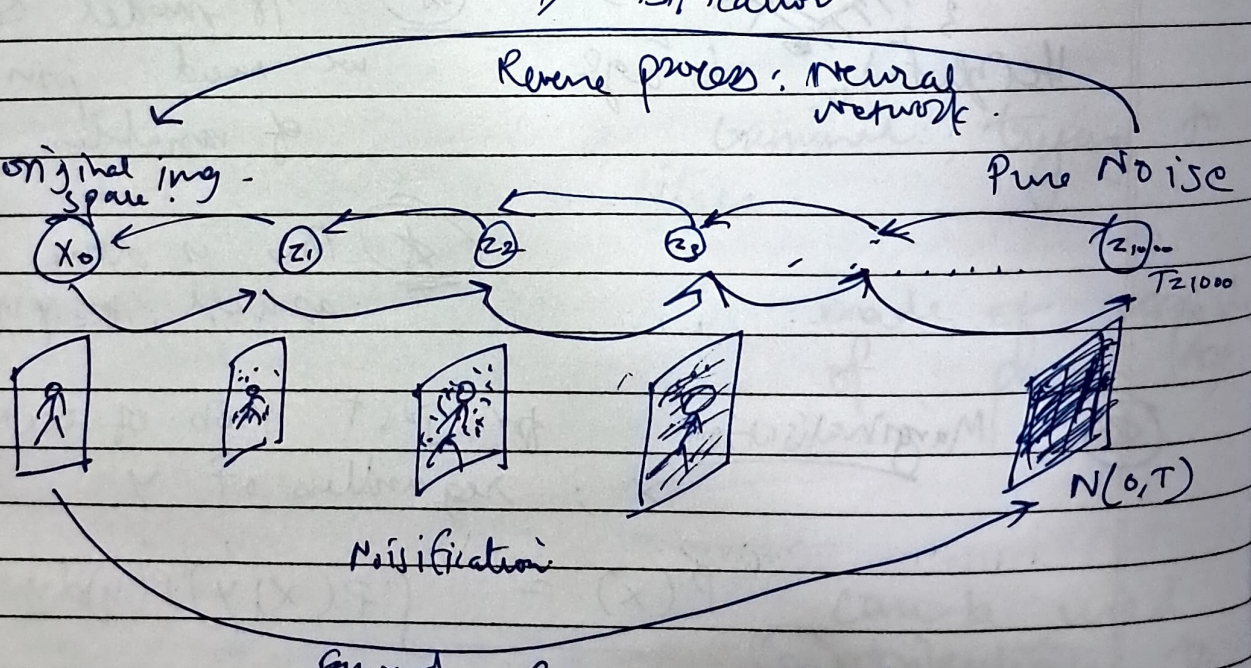
- So basically this is what happens in generative models where we learn from prob. distrib. of our data.

- we let the neural network learn the parameters of our distribution.

- we have dataset of images & we want to learn very complex distri that we then, use to sample from.

just like criminal wanting to generate fake identities.

=) Latent variables : Stable Diffusion



it is easy to add noise than to remove it.

latent var. models.

- Math: we have orig. img how to generated noisified image:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \underbrace{\sqrt{1-\beta_t}x_{t-1}}_{\text{Mean}}, \underbrace{\beta_t I}_{\text{variance}})$$

β = noise amt. Markov chain of var

Gaussian distrib.

forward process.

$$q_t(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I)$$

from orig img to img at t. without intermediate

noisy → less noisy → remove noise.

reverse process

neural network learns parameter. how to denoise

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

⇒ Diffusion Model: $p_\theta(x_0) = \int p_\theta(x_{0:T})\, dx_{1:T}$.

⇒ How do we train: ?

Algorithm:

1. repeat.
2. $x_0 \sim q(x_0)$
3. $t \sim \text{Uniform}(\{1, \dots T\})$
4. $\epsilon \sim \mathcal{N}(0, I)$
5. Gradient on $\nabla_\theta \|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\|^2$

6. until converged / min.

⇒ What we do is that we Optimise the lower bound.

like for eg:

Revenue ≥ Sales.

↑ lower bound

to inc. Revenue ↑ inc. Sales ↑.

same for us.

⇒ Based on Algo ϵ_θ + NN.

$$\epsilon_\theta\left(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\,\epsilon, t\right)$$

noisy img — time t at which noise added

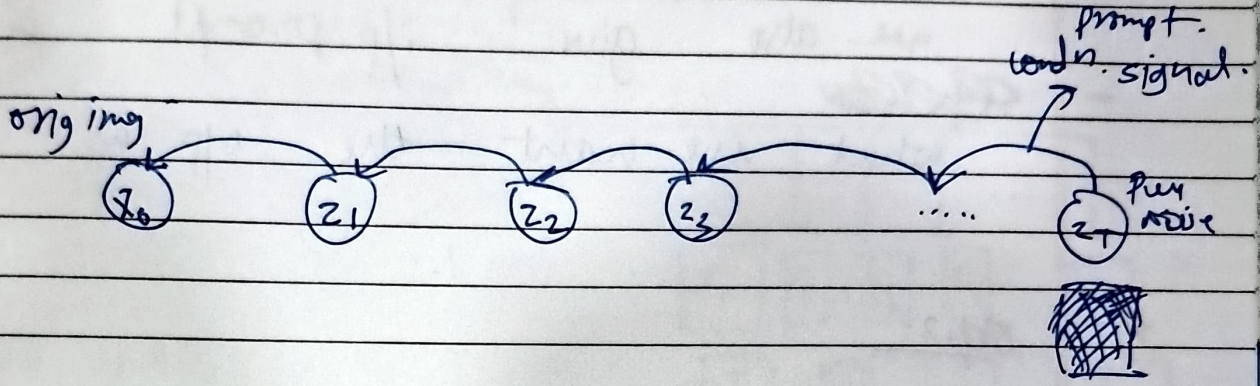
NN has to calc. how much noise added / present.

On applying GD, we maximise ELBo, & log likelihood.

⇒ Sampling New data from noisy images at T time

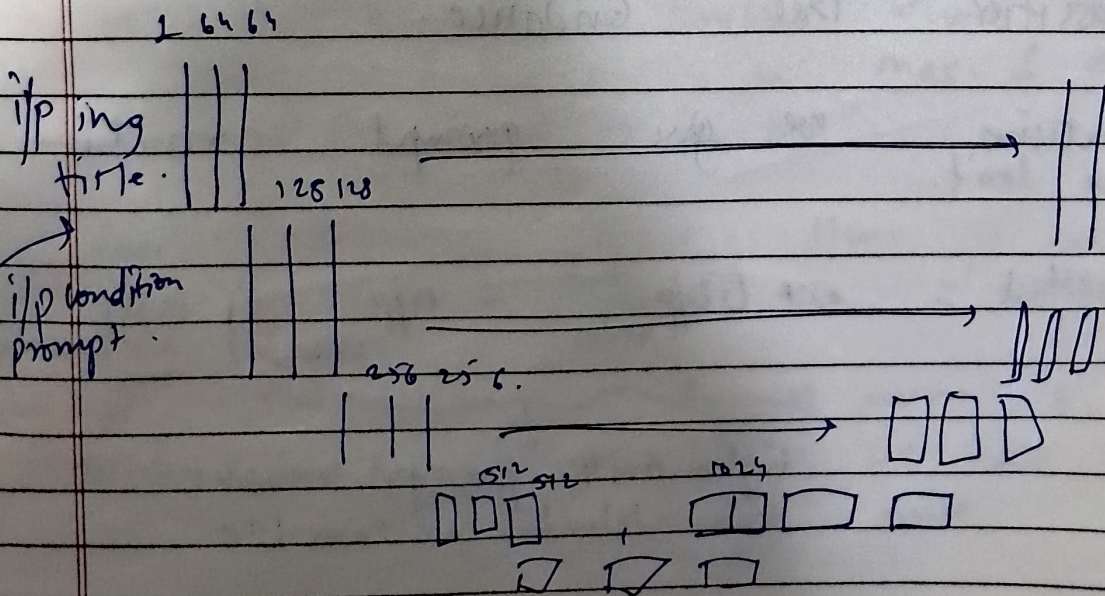
- During the Rev. process we recursively calc. how much noise there & remove it by help of NN.
- By recov. noise from pure noise we generate new data.

- But to create something we want sensible we want to control the noisification process.



- Our final goal is to model a distrib. Theta $P(\theta)$ such that $P_\theta(x)$ such that we maximize likelihood of our data to learn this distrib. we maximize the elbo (lower bound) by minimize the loss ∇_θ which learns the distrib.

- we use the UNet Model which receives a noisified img and has to find out noise.



* - step 1:

- give UNET model noisified image T = 1000
  & the model predicts the noise level.
  we also give i/p prompt as to what we want the o/p as. CAT etc

* - step 2:

- we give same i/p as noise but now we don't give i/p prompt, so the model will build some o/p to generate something.

- This way we combine o/p of the both steps to get type this & type of output we want. closer to prompt.

classifier free guidance.

- Sometimes we give prompt sometimes we don't.

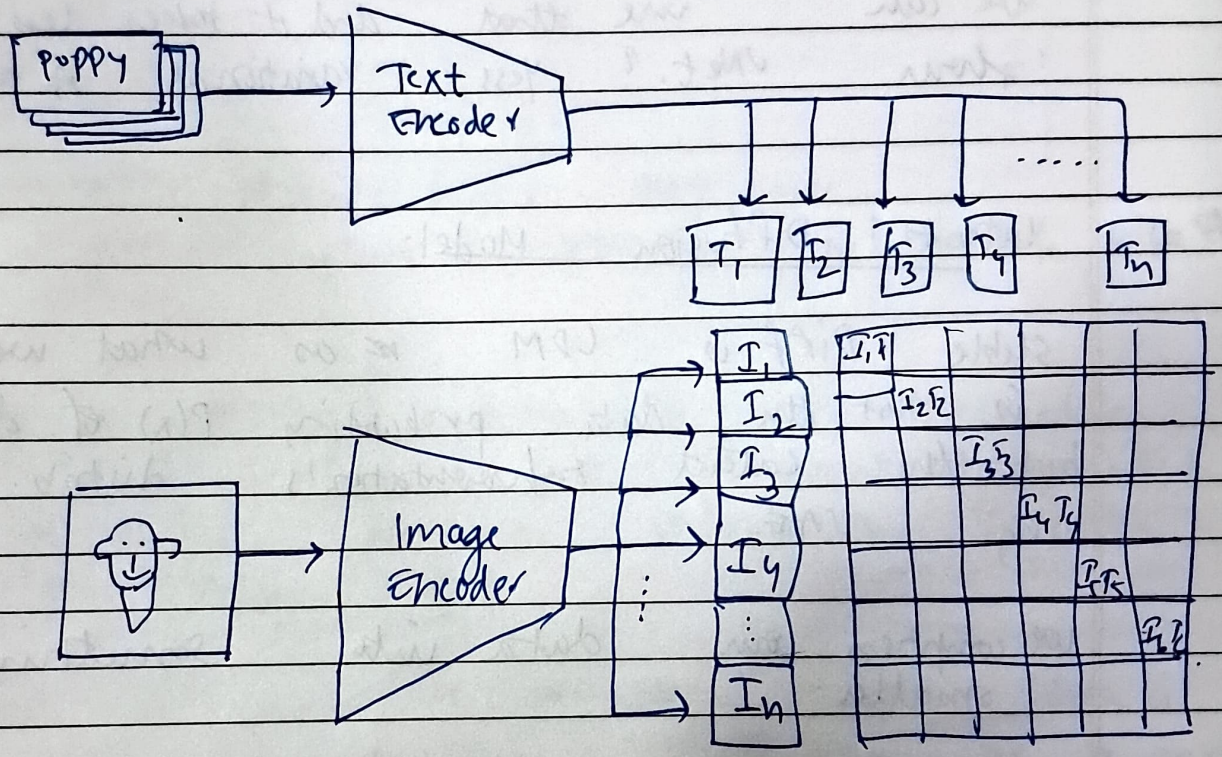
output = $w * (o/p_{condn.} - o/p_{uncondn}) + o/p_{uncondn.}$

higher this & prompt resembles.
lower value & less resemble.

* Scene based models.
↳ classifier Guid.

⇒ CLIP (Contrastive Language - Image Pre-training):

1.) Contrastive Pretraining: (OpenAI)



- We create a loss function where the diagnol must I_nI_n must be max. & the rest values of matrices to be 0.

Signals.

These embeddings act as the i/p conditions to the model. to denoise the image.

- Now the thing is it is seems big process to do repeatedly every time we have to go to UNET for finding noise.

- If img is big (512x512) then big matrix will be there.

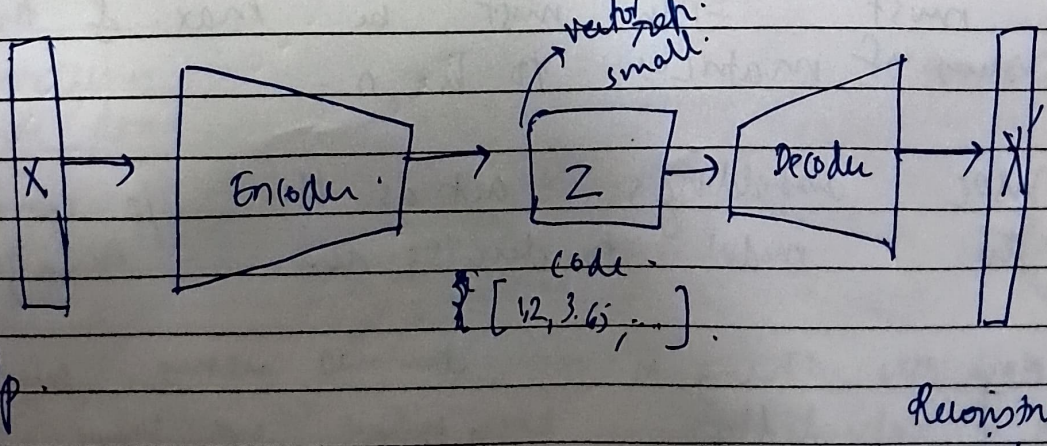
can we compress the image so that we can use that and it takes less time than UNet.? yess variational Auto Encoder.

⇒ Latent Diffusion Model:

Stable Diff is LDM as what we learn is not the data probability P(x) of dataset but the latent representation's distrib. by using VAE.

We compress our data into something smaller.

It is like zipping a large file. Something same.



- Orig image passes thru the Encoder and is conv. as vector em format resized & then passing them the decoder convert as small but same reconstr. i/p.

- Issues:

→ code doesn't make any sense.
→ doesn't capture semantic relationship b/w data.

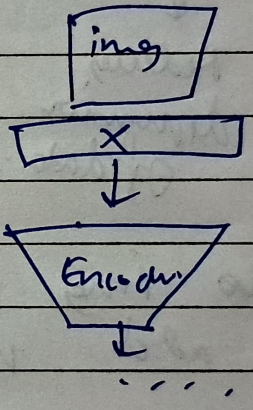
NOTE: A normal Autoencoder is pretty useless, what we do is we variational Autoencoder to capture semantic understanding between the ~~Images~~ data.

Here we use the variational AE. data is distri across multivar. distribution as a gaussia & find mean & Sigma.

⇒ Architecture of Stable Diffusion: (Text to Image)

- in case of img-to-image, we input img.

X and then provide i/p for any changes to be made. ∴



& the Rest is same.

we noisify the img & the amt that we noisify demonstrates how much freedom unet has to modify img.