GAN
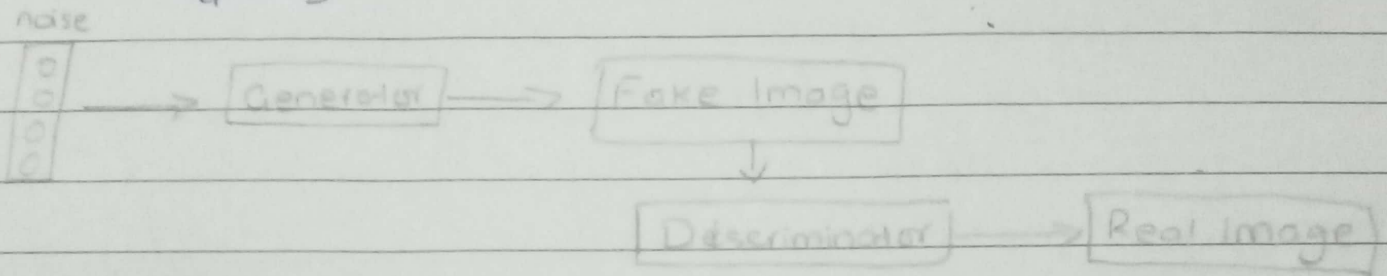
- To create or generate realistic distribution
- People not exist but can create realistic image
- Competition b/w 2 NN
- Generator ⟹ produce data
- Discriminator ⟹ classify ⟨ possible / not possible ⟩   yeh bolayega is it exact. Generator fools the discriminator.
- Min-Max
- It is Dual objective problem

Steps taken by GAN

$$\min_{G} \max_{D} (D, G)$$

noise

- train 2 times discriminator as they need to classify and get train more so it will be more powerful but not that regid that it won't accept any fake image. Both training works together
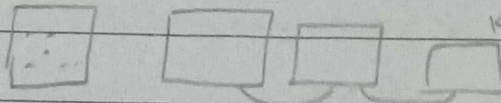- MINIST dataset also store. Generator is Inverse convolution networks where it trace image as output

Train the GAN

1) Build model with same no. of layers
2) true distribution & train discriminator by binary
3) Create input from noise & feed to discriminator
4) Generator output is their input
5) Uses Sigmoid due to binary classification
6) loss is fed-back as dist b/w fake image & real image So work by KL-divergence
Victory 7) Done till convergence

Challenges

- Convergence is slow especially in exploding gradient → weight main bohut change

Kharaj bhi hosakta
result ana mushkil hai

- Sensitive to hyperparameter
- Cannot use object count

  Stake Diffusion ⟹ Improved / powerful GAN
  ↳ Diff from GAN ⟹ stat diff can count

- Model Collapse (vanishing Gradient) ⟹ Gradient itna kum ke weight change na ho ⟹ leaky Relu

  Agr dis pehle hi learn karle to gen kabhi learn nhi hogu

- Unsupervised = GAN

Guidelines

- tanh for activation, sigmoid for generator
- freeze training after sometimes

TYPES

1) Vanilla GAN

2) CGAN (conditional for classifier for label to put in generator like 3)

3) Wasserstein we use loss func like earthmover

4) Style GAN we use for fashion. trace image as input

Context-aware Recommendation

- Context is important for personalize recommendation
- Views: Representational ⇒ do not change over time
  - like dob, first lang

  Interactional → that change over period of time
  - like budget change hota hai

- Sometime it is not given so infer it

- Knowledge user context:
  1) fully observable = factors known like signup karte hue details
  2) Partially observable = Some known like if pakistani so urdu language
  3) Unobservable = Nothing known like family member but useful if eg mood by Imdb so activity check

- Architecture
  1) Contextual pre-filtering: filter and recommend
     assign weight and make — re-ranking so comedy movie first ne release
     agr comedy chahiye tou remove emotion then recomm
  2) Contextual post-filtering: after recommend order data
     additional data like country. Any technique collaborative
  3) Modelling: add context inside Rs algo to add context

- Drawbacks = 1) No serendipity as it filters out
  2) Rs no integrity, performance not evaludible
  3) Complex implementation

- Advantages = 1) Computational low
  2) User profile easily createable agr same filtering so a user ki ek profile