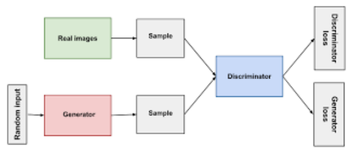
Day 11: Generative Adversarial Networks (GANs)

Introduction to GANs

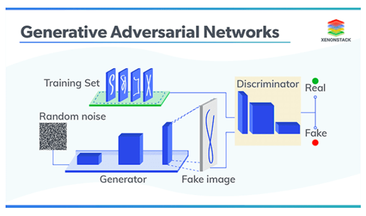
- !
  - We are now at a point in history where DL and Neural Networks can generate new human face from scratch.
  - 
  - This was an AI generated image using Flux and Lora combo.
  - This is achieved by the technique of a GAN model
- **What?**
  - An approach to generative modeling that generates a new set of data based on training data that look like training data
- **Why?**
  - Traditional ML algorithms and Neural Networks can easily be fooled when the data is noisy
    - Large amount of additional meaningless information in it called noise
  - They misclassify images post adding noise to the data
  - Led to a question 🤔
    - "Is it possible to create a neural network that can visualize new patterns similar to the train data?"
  - So GANs were built to create fake results that are similar to the original
- **Types**
  - Deep Convolutional GAN (DCGAN) — Day 12
  - Wasserstein GAN (WGAN) — Day 13
  - Conditional GANs (cGANs) — Day 14
  - An advanced GAN — Day 15
- **Applications**
  - Realistic pictures of people
  - Generate Music by using some clone Voice
  - Animation - Creation of anime characters and characters in video games
  - Image to Image Translation
  - Low resolution to High resolution

GAN Architecture



- **Generator Network**
  - Generates new samples of data that resemble the training data
  - Takes random noise as input and produces data samples
  - Primary Objective: Tries to fool, by creating samples that are so real, the Discriminator cannot tell apart
- **Discriminator Network**
  - Purpose: Distinguish between real data and data generated by the generator
  - Receives both real data and and fake data
  - The output: a probability whether the input data is real of fake
- But why are we doing all this?
- **Adversarial Training Process**
  - Aim is to improve both generator and discriminator - back and forth competition
  - Generator's Goal: Enhance it's ability to create realistic data — Keep getting better at fooling the discriminator
  - Discriminators goal: Improve it's ability to distinguish between real and fake
  - They iteratively improve and both the networks advance by learning from each other's feedback
- **Loss Functions**
  - Generator Loss: Measures how well generator is able to fool the discriminator — Aim: Minimize discriminators ability to detect
  - Discriminator Loss: Measures how well discriminator is differentiating b/w real and fake — Aim: Maximize correct classification of real/fake
- **Random Noise Input**
  - A vector of random values sampled from a normal distribution
  - Purpose: Provide generator with diverse inputs, so that it can produce varied data samples

Working of a GAN



- Step 1: Generator First move
  - G takes a random noise vector as input
  - Using its internal layers and learned patterns, G transforms the noise vector into a new data sample, like a generated image.
- Step 2: Discriminator Turn
  - D receives two kinds of inputs:
    - Real data samples from the training dataset
    - The data samples generated by G in the previous step.
  - Analyze each input and determine whether it's real data or something G cooked up
  - Outputs a probability score between 0 and 1. 1 indicates real, and 0 suggests it's fake
- Step 3: Generator's improvement
  - When D mistakenly labels G's creation as real — G receives a significant positive update
  - Feedback helps G improve its generation process to create more realistic data
- Step 4: Discriminator's Adaptation
  - If D correctly identifies G's fake data (score close to 0) — D is further strengthened in its discrimination abilities
  - Ongoing duel between G and D refines both networks over time

Understand the Loss Function of a GAN

$$E_x[log(D(x))] + E_z[log(1 - D(G(z)))]$$

- `D(x)` is the discriminator's estimate of the probability that real data instance x is real.
- E_x is the expected value over all real data instances.
- `G(z)` is the generator's output when given noise z.
- `D(G(z))` is the discriminator's estimate of the probability that a fake instance is real.
- E_z is the expected value over all random inputs to the generator (in effect, the expected value over all generated fake instances G(z)).

Hands-on: Image generation using GAN