Digital Image Processing – SE3071

Generative Adversarial Network

Lab Activity – 09

Year 3 – Interactive Media **Semester 1, 2023**

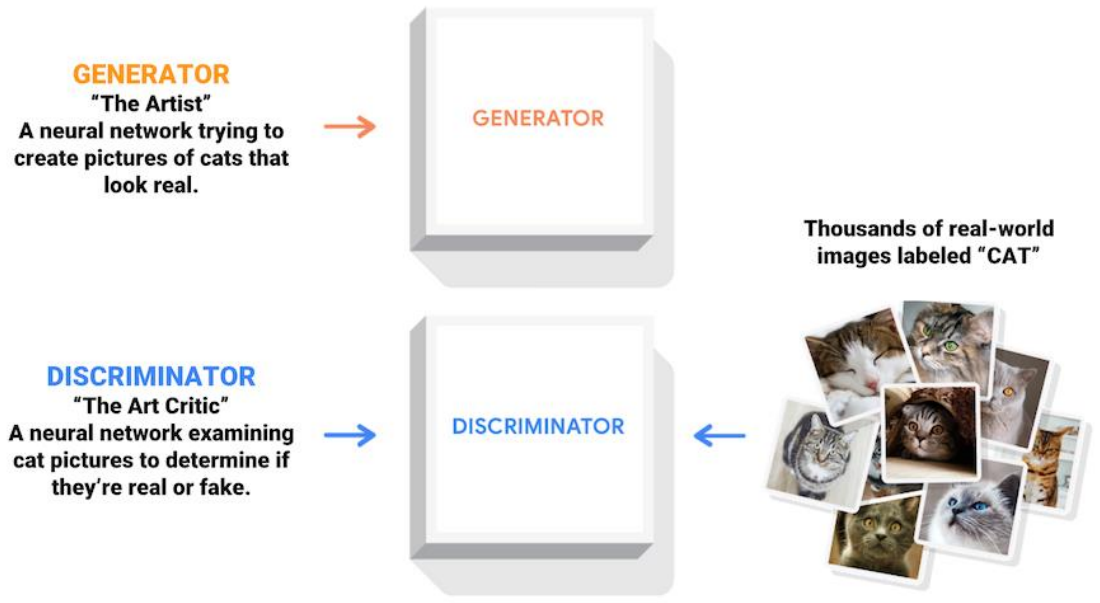
Generative Adversarial Network (GAN)

A Generative Adversarial Network (GAN) is a specific type of neural network designed for the purpose of generating realistic images, making it suitable for a wide range of applications. A GAN consists of two key components: a generator and a discriminator network.

The primary role of the generator network is to take as input a random noise vector and transform it into realistic, high-quality images. It achieves this by learning to generate images that closely resemble those found in the training data. The generator essentially creates synthetic images from random noise.

Discriminator network has a straightforward task: to determine whether a given input image is real or not. It accomplishes this by examining the visual characteristics of the image and making a binary classification decision.

The ultimate objective of training a GAN is to reach a state where the generator becomes highly proficient at producing images that are so convincing that the discriminator is left bewildered and unable to reliably distinguish between real images and those generated by the generator. This adversarial process leads to the continual improvement of the generator's image synthesis capabilities, resulting in the creation of realistic and convincing images.


GENERATOR
"The Artist"
A neural network trying to create pictures of cats that look real.
GENERATOR
Thousands of real-world images labeled "CAT"
DISCRIMINATOR
"The Art Critic"
A neural network examining cat pictures to determine if they're real or fake.
DISCRIMINATOR

Digital Image Processing – SE3071

Generative Adversarial Network

Lab Activity – 09

Year 3 – Interactive Media **Semester 1, 2023**

Part 1

1. Import all necessary libraries
2. Import and Preprocess the Dataset
   a. Load the mnist dataset - **tf.keras.datasets.mnist.load_data()**
   b. Create training images and training labels.
   c. Apply certain transformations.
3. Build Generator Model
   The generator uses **tf.keras.layers.Conv2DTranspose** (upsampling) layers to produce an image from a seed (random noise).
   tf.keras.layers.LeakyReLU activation for each layer, except the output layer.
4. Build Discriminator Model

   The discriminator is a **CNN-based image classifier**.

5. Create Custom Loss Functions and Optimizers

   Discriminator Loss - This method quantifies how well the discriminator is able to distinguish real images from fakes. It compares the discriminator's predictions on real images to an array of 1s, and the discriminator's predictions on fake (generated) images to an array of 0s.
   Generator loss - The generator's loss quantifies how well it was able to trick the discriminator. Intuitively, if the generator is performing well, the discriminator will classify the fake images as real (or 1). Here, compare the discriminators decisions on the generated images to an array of 1s.

6. Create the Custom Training Loop
7. Train the Model - Call the **train()** method defined above to train the generator and discriminator simultaneously.
8. Create a GIF - Use **imageio** to create an animated gif using the images saved during training.

Part 2

With the knowledge gain from Part 1, Use the GAN models (generator & discriminator) to train the model for any dataset.