Deep Convolutional Generative Adversarial Network (DCGAN)

Before diving in DCGAN it is important to have a brief idea about Generative Adversarial Network (GAN). DCGAN is of the many flavors of GANs.

GAN is composed of two main neural networks: \*\*Generator\*\* and \*\*Discriminator\*\*.

Generator’s task is to learn the patterns in the input images and generate new images from noise.

Discriminator’s task is to identify the real images (actual images from dataset) and the fake images (images produced by Generator).

Both these networks act like competitors. The Generator tries to fool the Discriminator and the Discriminator tries to be smart in identifying real and false images.

Training GAN’s is always challenging and requires a lot of processing power. I made use of Google Colab.

\*\*Generator\*\*

The generator takes in a fixed-length input vector (in my case column vector of 100), it is randomly drawn from the Gaussian distribution using NumPy. The amin difference between DGAN and GAN is the use of Transposed Convolutional layers for up-sampling images. The final layer outputs images of shape \*\*64 x 64 x 3). Activation: Leaky Relu, Tanh(for final layer). Output of this network is given as input to the Discriminator along with real images in batches.

\*\*Discriminator\*\*

The discriminator is a simple convolutional neural network for identifying features in an image. Since the discriminator’s task is to differentiate between real and fake images “binary crossentropy” is used as a loss function. I have used Adam optimizer. Activation: Leaky Relu, Sigmoid(for final layer).

The dataset was downloaded from the official Pokémon website.