

# Day 12: Deep Convolutional Generative Adversarial Networks (DCGANs)

## Hands-on

Let's try PyTorch today!

### Introduction

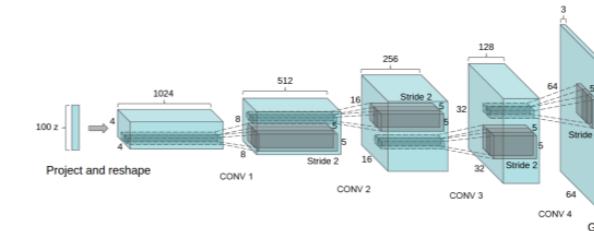
To reduce the problem of mode collapse

When the generator gets biased towards a few outputs and isn't able to produce outputs of every variation from the dataset

Eg: MNIST

Consequently, discriminator also gets optimized towards that particular digits only

DCGANs solve this problem



### Architecture

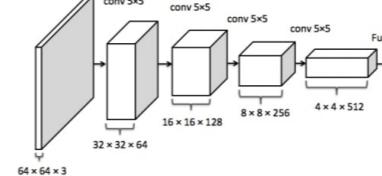
Takes 100 uniform generated values using normal distribution as an input

It changes the dimension to 4x4x1024

Convolution 4 times with a stride of 1/2

Generated output has dimensions of (64, 64, 3)

In the DCGAN paper, Researchers used



Batch Normalization which helps in stabilizing training

ReLU activation function in all layers of the generator, except for the output layers

Determine that the image comes from either a real dataset or a generator

Designed similar to a convolution neural network that performs an image classification task

But, researchers suggested some changes

only strided-convolutions with LeakyReLU as an activation function

The input of the discriminator is a single image from the dataset or generated image

Output is a score that determines whether the image is real or generated