Super Resolution Using the SRGAN

Presented by: Farhan Ajmal Khan

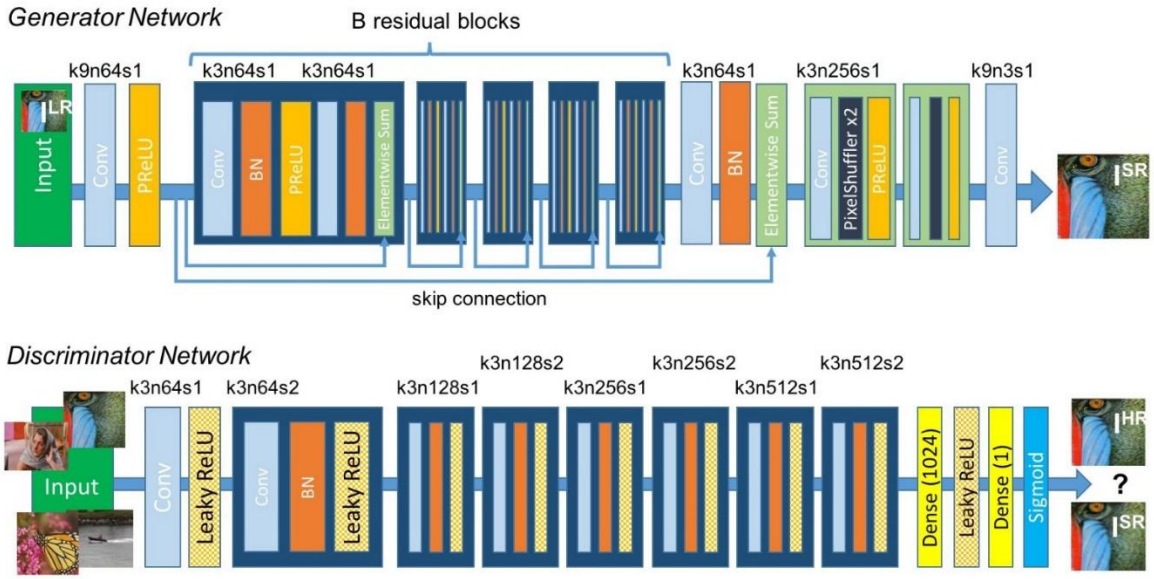
Introduction

Image super resolution can be defined as increasing the size of small images while keeping the drop-in quality to minimum or restoring high resolution images from rich details obtained from low resolution images. This problem is quite complex since there exist multiple solutions for a given low resolution image. This has numerous applications like satellite and aerial image analysis, medical image processing, compressed image/video enhancement etc.

Problem Statement:

To recover or restore high resolution image from low resolution image. There are many forms of image enhancement which includes noise-reduction, up-scaling image and color adjustments. This post will discuss enhancing low resolution images by applying deep network with adversarial network (Generative Adversarial Networks) to produce high resolutions images.The main target is to reconstruct super resolution image or high-resolution image by up-scaling low resolution image such that texture detail in the reconstructed SR images is not lost.

Deep Learning Model:



Training procedure is shown in following steps:

- We process the HR(High Resolution) images to get down-sampled LR(Low Resolution) images. Now we have both HR and LR images for training data set.
- We pass LR images through Generator which up-samples and gives SR(Super Resolution) images.
- We use a discriminator to distinguish the HR images and back-propagate the GAN loss to train the discriminator and the generator.
- Residual blocks: Since deeper networks are more difficult to train. The residual learning framework eases the training of these networks, and enables them to be substantially deeper, leading to improved performance. More about Residual blocks and Deep Residual learning can be found in paper given below. 16 residual blocks are used in Generator.
- PRelu(Parameterized Relu): We are using PRelu in place of Relu or LeakyRelu. It introduces learn-able parameter that makes it possible to adaptively learn the negative part coefficient.
- k3n64s1 this means kernel 3, channels 64 and strides 1.
- Loss Function: This is most important part. As discussed we will be using Perceptual loss. It comprises of Content(Reconstruction) loss and Adversarial loss.

Results:

Original

(178x218)

Generated

(712x872)

Corrected (712x872)

Code execution:

The project is developed in Google Colab and consist of 2 files that has to be executed. First make a folder on google share drive and placed the files in it. Download the celeb dataset from the link below and upload to the google drive with 60% in celeb_HR and 40 in celeb_LR.

https://drive.google.com/drive/folders/0B7EVK8r0v71pTUZsaXdaSnZBZzg

- Open the main file and execute the first code section.
- Using second code section move to the folder where you placed requirements file.
- Then execute the 3rd section, which will install the requirements for environment.
- Then set the paths to data directory where you placed the celeb folders.
- Execute all the sections, VGG19 pretrained model and weights will downloaded itself.