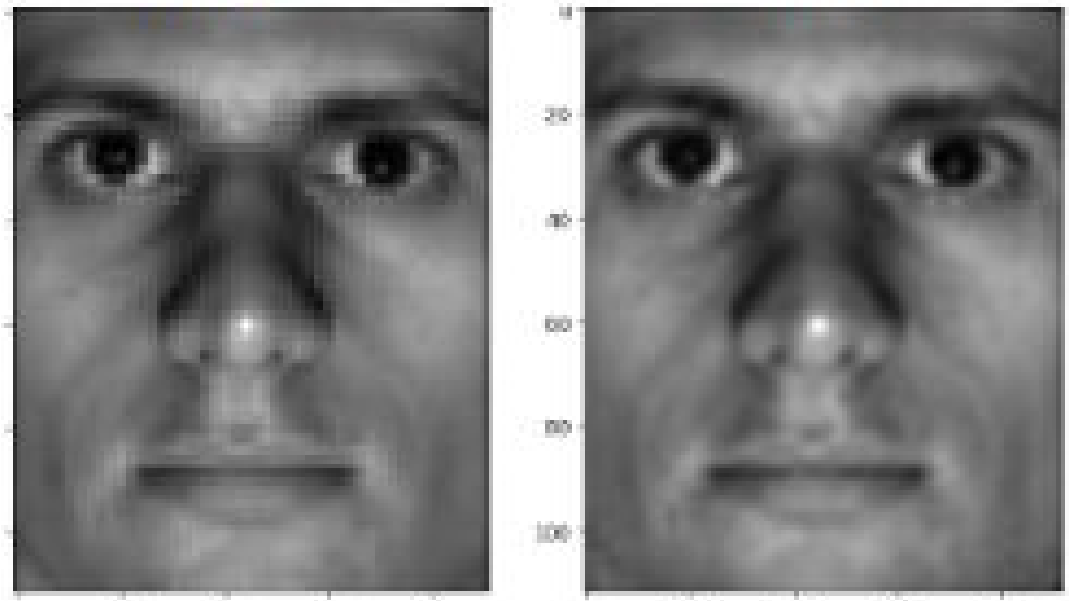
Enhance! A Generative Adversarial Network for Image Upscaling



By:
Saad Jameel (1003365878)
Mohammad Mustafa Arif (1003116736)

1971 Words (Excluding Table of Contents and Bibliography)

Table of Contents

1| Goal and Motivation

The goal of “Enhance!” is to build a neural network that upscales low resolution images of faces to high quality, high resolution images of faces. The problem is an area of significant importance for any application involving data compression and reconstruction. Current methods of upscaling rely on interpolation, a method that extrapolates pixels from surrounding pixels. However, there is fundamentally less information in a downscaled image, so interpolation never reconstructs the original image, and often creates a blurry output.The lack of information in a low resolution image means that if we want to reconstruct the image, we have to “create something out of nothing,” using our knowledge of what the image should look like rather than the knowledge contained within the image itself. We thus propose a machine learning approach to upscaling, one that uses a generative adversarial network that learns what a face should look like, rather than just extrapolating details in the face within the image itself.

2| Description of Overall Software Structure

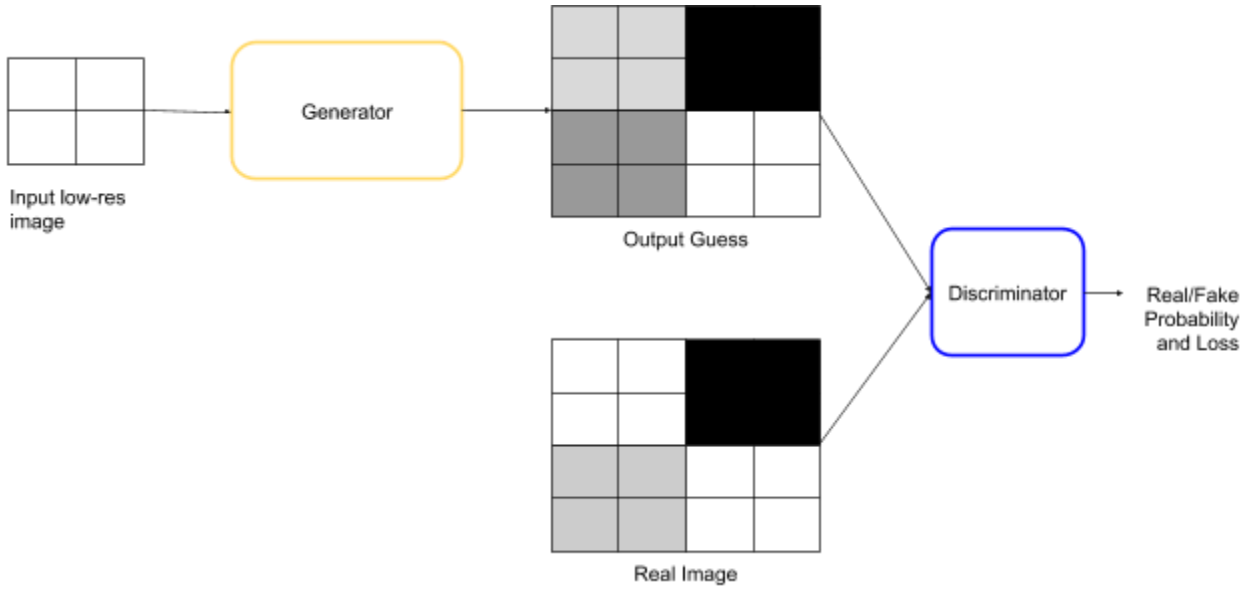


Figure 2.1: A block diagram representing the structure of the GAN

The software consists solely of a Generative Adversarial Network (GAN) to “fill in the dots” as it turns low resolution images (56x46 pixels) into higher resolution images (112x92 pixels).

The network has two components (each of which is a neural network itself): the Generator and the Discriminator (see Figure 2.1). The Generator takes in the low resolution images as the input to produce their higher resolution counterparts, whereas the Discriminator distinguishes a real high resolution image from the one generated by the Generator. The two are trained in a cyclic manner, where the discriminator always tries to identify the images that came from the generator and the generator tries to get the discriminator to predict that these the generated images are real. One caveat in the GAN implementation in this project is the loss function of the Generator. In addition to comparing the discriminator output for a generated image with the “real” state (through BCEloss), the loss also has a secondary component that compares the generated image with the original high resolution image to quantify how far off the two images are (via MSELoss).

3| Sources of Data

The data used in this project came from a dataset called the **Yale Database of Faces**, which contained 2447 closeup portrait images of adult faces. The database is available UCSD website [1]. All the images in the database were grayscale. To preprocess the data,the images were first downscaled to a standardized resolution of 112x92 pixels to make the label images. Then, the images were compressed to a resolution of 56x46 pixels, which were treated as the feature images. To compare to industry standard interpolation, the 56x46 pixel images were upscaled using interpolation to compare to the generated images.

4| Description of the Machine Learning Model

4.1| Discriminator

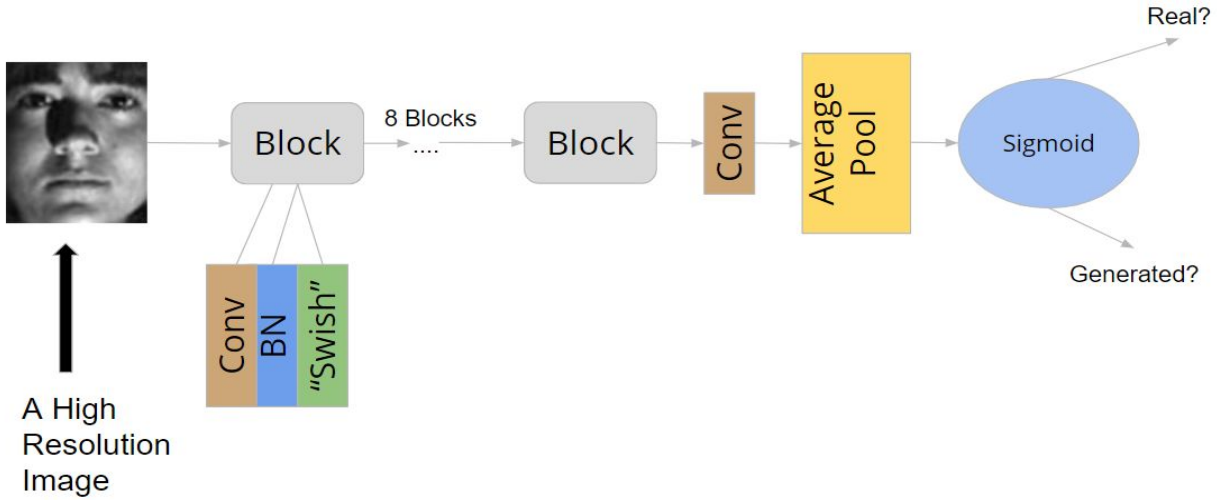


Figure 4.1.1: The internal network within the discriminator

The discriminator has a fairly standard architecture as it is essentially performing binary classification on each input image (between the ‘real’ and ‘generated’ classes). The network consists of 8 convolutional layers, each of which are batch normalized with respect to the entire feature map after each layer. The first layer increases the number of channels, and the second layer has the same number of input to output channels. This pattern is continued for all layers, because the 1 to 1 layers (same number of input and output channels) learn every feature in the feature map. At the end of every layer, there is a swish activation function (see Figure 4.1.2). This function is a continuous approximation to the ReLU function, as shown in Figure 4.1.2. Finally, there is one last convolutional layer, an average pool function, and a sigmoid function to convert the number into a probability of the image being real. The dimensions are kept such that the output is a single number for one image. Due to the complexity of the model and the length of the training, the Adam optimizer was used to train this network. The total loss was a sum of the BCELoss for the fake images, and the BCELoss for the real images.

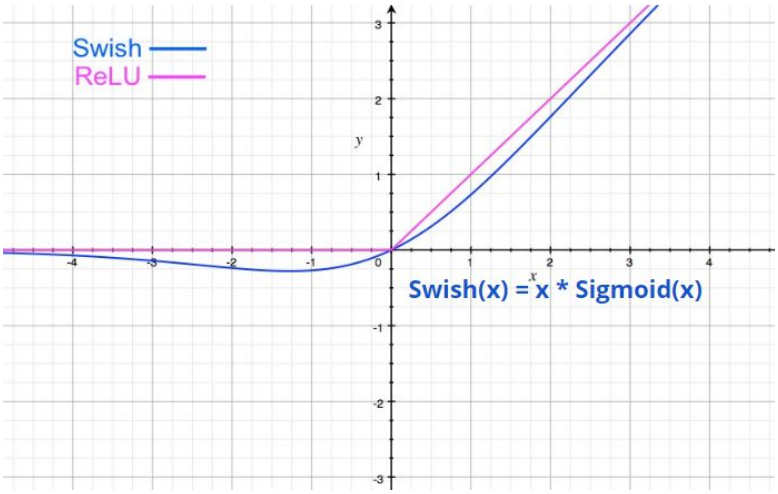


Figure 4.1.2: The “Swish” Activation Function

4.2| Generator

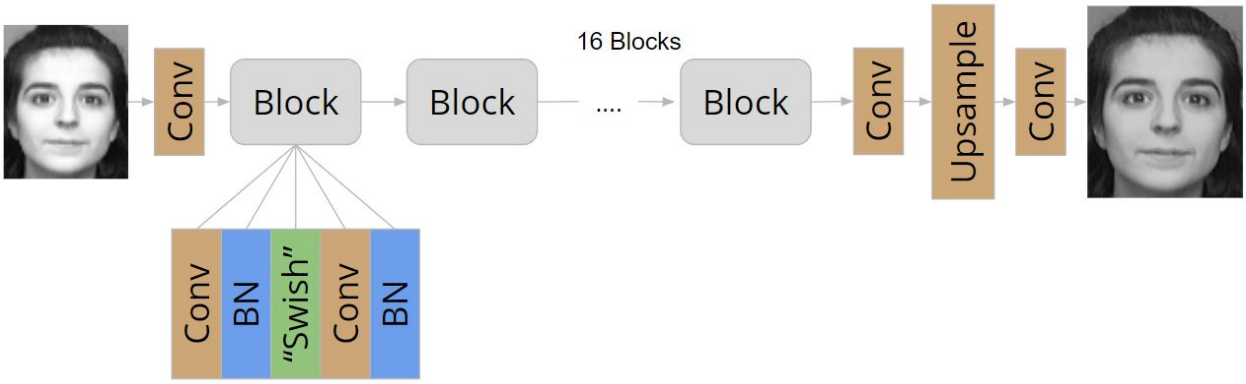


Figure 4.2.1: The internal network within the generator

The generator consists of an initial convolutional layer that blows up the 1 channel, low resolution gray scale image into 20 channels (left on Figure 4.2.1). The 20 channel feature map is then fed into 16 “residual blocks” which all output 20 channels, thus preserving the size of the original image (see Figure 4.2.1). These residual blocks learn the features of the input image, and can be stacked over and over again for more complex images[2]. A convolutional layer and an upsample to increase the size of the image follow at the end to generate the high resolution image.

The activation function used throughout is the swish function (see Figure 4.1.2), which is a continuous approximation to the ReLU function. Swish is computationally more expensive, but it has well defined gradients at an input of 0 along with a small gradient in the negative region. Swish thus prevents neurons from dying and will be

more likely to converge regardless of initialization, and so it works better with larger models [3].

The optimizer used was Adam with (β_1,β_2) = (0.5,0.9) to allow for fast convergence, with only a negligible decrease in accuracy that is generally not seen by the human eye. This was considered necessary due to size and length of training for the model. To quantify the loss from the generator, two separate components were considered (as mentioned above): the content loss and the adversarial loss. The content loss measures how different the generated image is from the real image, and the adversarial loss is a measure of how 'unreal' the discriminator thinks the generated image is. The total loss function is defined as following:

Content Loss = MSELoss [Generated Image, Original Image]

Adversarial Loss = BCELoss [1, Discriminator(Generated Image)]

Total Generator Loss = Content Loss + (1/1000) * Adversarial Loss

It should be noted that defining the loss function in this way was a key step in getting credible results that actually resembled the low resolution input image. This technique was taken from the research paper: Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network [2].

5| Training, Validation and Test

5.1| Training and Validation

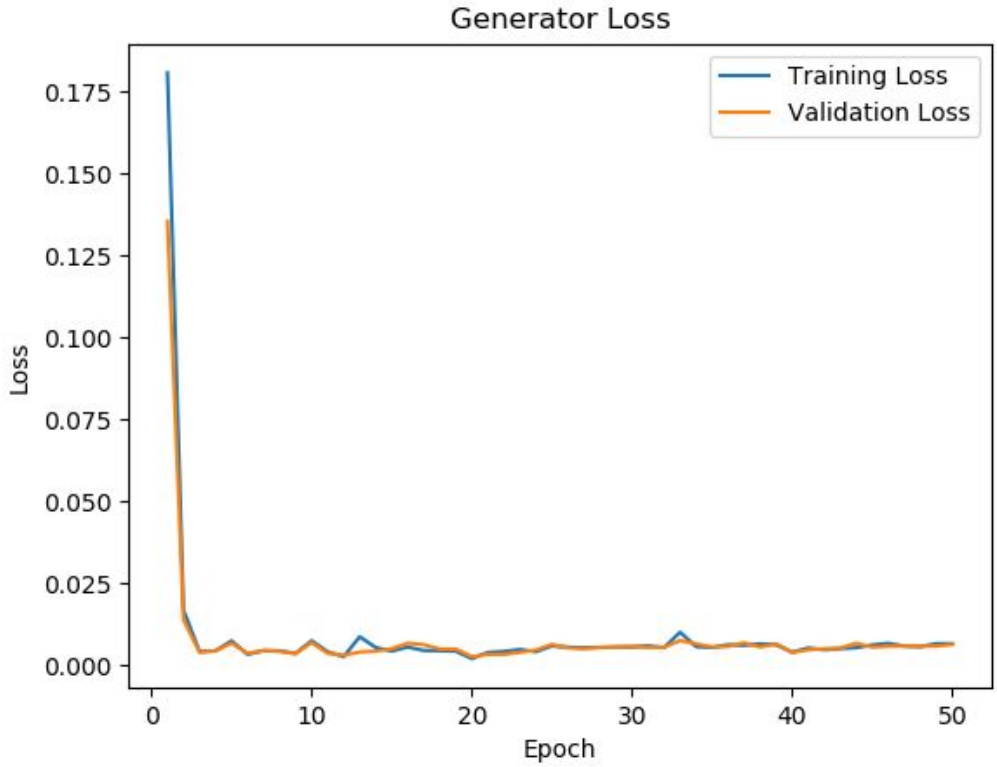


Figure 5.1.1: The training and validation loss of the generator as the model trains

As is evident from figure 5.1, the training and validation loss were approximately the same throughout the entirety of training, and thus the model is fitting extremely well. The optimal hyperparameters were 50 epochs, a batch size of 64 of images, and a learning rate of 0.0002 for both the generator and the discriminator. After the generator produces the fakes, and the discriminator determines whether the fakes and the originals are real, the parameters are updated based on the loss and optimizers described in 4.1 for the discriminator, and 4.2 for the generator.

To evaluate the quality of the images produced, PSNR (Peak Signal to Noise Ratio) was used. The PSNR measures the amount of noise or difference between two signals, in this case the pictures, using Mean Squared Error, and then inverts it (see Figure 5.1.2). Thus when a reconstructed image is compared to the original, higher resolution image, a higher PSNR means a better reconstruction. The PSNR of validation set for the interpolated and generated versions of the image relative to the

original high resolution images are plotted in Figure 5.1.3 as the generator is trained. The generator evidently surpasses interpolation after about 45 epochs.

$$PSNR = 10log_{10}\left(\frac{MAX_I^2}{MSE}\right)$$

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2$$

Figure 5.1.2: The definition of PSNR where I(i,j) are the (i,j)th pixel of one image, and K(i,j) refers to the corresponding pixel of the other image[3].

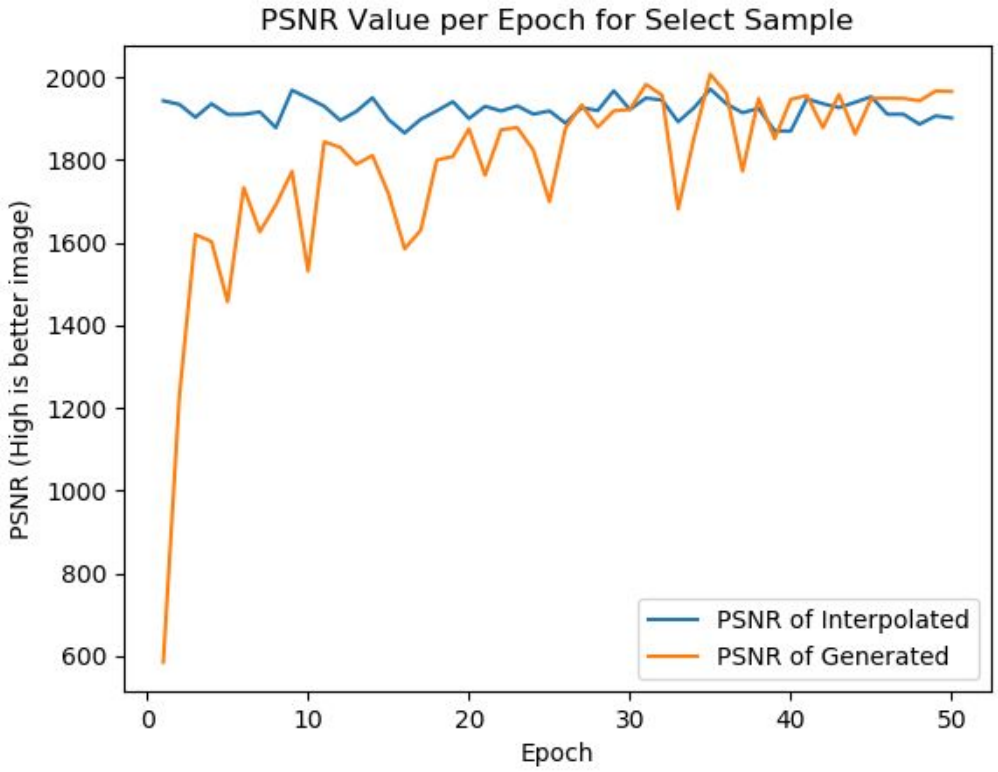


Figure 5.1.3: The PSNR as the generator is trained in comparison to the interpolated image

A select sample of the validation set was taken and images were produced to show the results of the generator (see Figure 5.1.4).

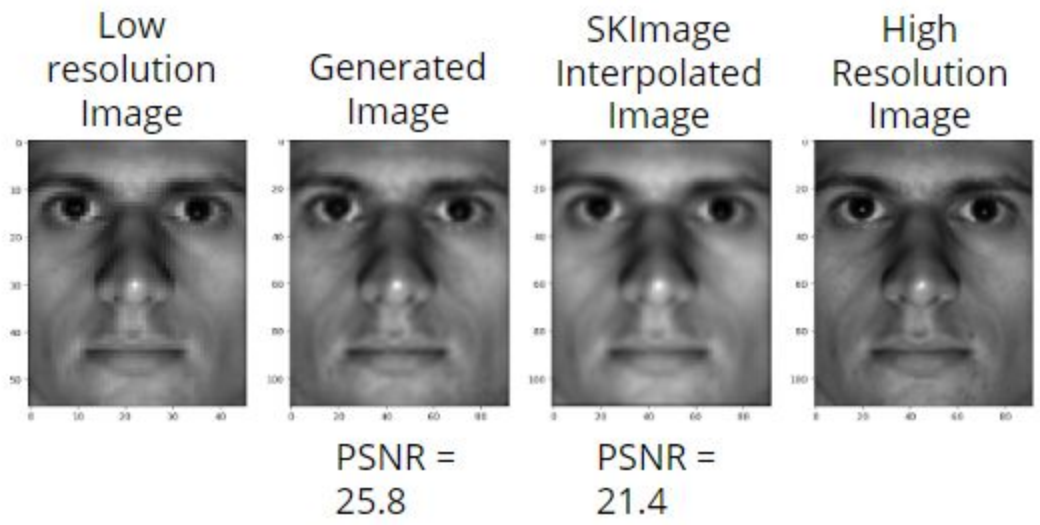


Figure 5.1.4: From left to right, the low resolution image, the generated image, the SKImage interpolated image and the high resolution image, with a PSNR value in reference to the high resolution image.

The discriminator had an accuracy of 100% by the third epoch, with small oscillations at certain epochs, but in general it learned extremely quickly how to distinguish the generated images from the real ones. It thus had an F1 Score, Recall, Precision and Specificity of 1.0 by the end of training.

5.2| Testing

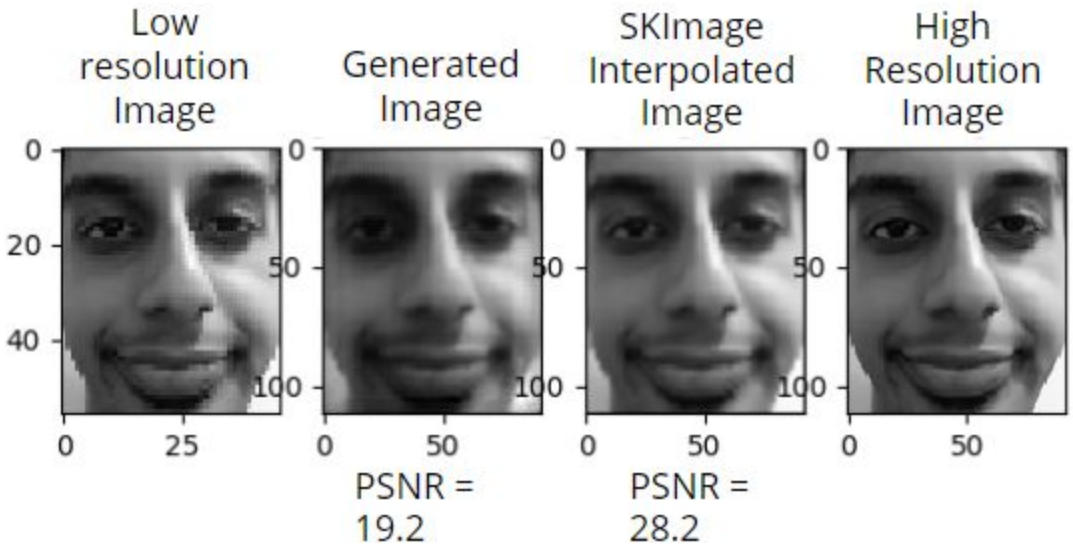


Figure 5.3.1: Visual representation of the upscaling of one of our own images versus the original high resolution image.

The testing set consisted of our custom images. As is evident from the visual representation, the generator works poorly on our own image. The original database consisted of images with similar lighting conditions and angles, and thus the generator learns to generalize only for those specific conditions. Thus the generated image for the testing set is fairly poor when it doesn't match the conditions of the training set, and it's reflected in the PSNR values.

6| Ethical Issues

Since the GAN produces high resolution images from low resolution images, it generates some of its own data which may make the subjects in the output images look significantly different from their actual selves. This is apparent in figure 6.1:

Figure 6.1: A Comparison of GAN Generated Images and Real Images [5]

For some applications, the misclassification that results from these differences can have immense ethical consequences. A good example is the identification criminals/terrorists from poor quality footage and convicting the wrong person due to immense differences between the real and the generated high resolution image.

Another ethical concern tied to this technology is the potential for its abuse, which could result in privacy breach. Some tools such as google maps intentionally pixelate the private details (such as faces and license plates) of the subjects in the footage in order to protect privacy. Upon misuse of the super resolution technology, this privacy may be compromised.

7| Key Learnings - What to do differently in the future

The GAN produced was generally successful and beat traditional upscaling in terms of PSNR, but only for specific controlled conditions. In the future, a large variety of data augmentation methods could help the GAN generalize to less controlled environments and thus produce better results. The model could be further expanded by expanding to coloured images, and from portrait faces to general pictures to reduce bias in the results. The model is already set up in such a way that the dimensions of the output image are always double the input dimensions as there are no linear layers, so expanding to a dataset of images of varying sizes would be trivial. The model itself is quite hefty so determining a way to condense it is most likely necessary for usable model deployment. Finally, adding a GUI to make it an interactive tool for people to actually use to upscale their personal images is absolutely necessary if the model is to be deployed.

New concepts and techniques learnt by working on this project include:

1) The swish function (as described above) for neuron activation. Using this activation function improves convergence and results in lower chance of neuron ‘death’

2) The technique for quantifying generator loss via content loss and adversarial loss (discussed above). This state of the art technique intelligently combines the two components and results in far more credible generator results.

8| Bibliography

[1] "Yale Face Database | vision.ucsd.edu", Vision.ucsd.edu, 2018. [Online]. Available: http://vision.ucsd.edu/content/yale-face-database. [Accessed: 02- Dec- 2018].

[2] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang and W. Shi, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network", *Arxiv.org*, 2018. [Online]. Available: https://arxiv.org/abs/1609.04802. [Accessed: 25- Nov- 2018].

[3] P. Ramachandran, B. Zoph and Q. Le, "Searching for Activation Functions", *Arxiv.org*, 2017. [Online]. Available: https://arxiv.org/abs/1710.05941. [Accessed: 25- Nov- 2018].

[4] "Compute peak signal-to-noise ratio (PSNR) between images - Simulink", *Mathworks.com*, 2018. [Online]. Available: https://www.mathworks.com/help/vision/ref/psnr.html. [Accessed: 25- Nov- 2018].

[5]"ENHANCE!: Upscaling images CSI-style with generative adversarial neural networks", Geoffreylitt.com, 2018. [Online]. Available: https://geoffreylitt.com/2017/06/04/enhance-upscaling-images-with-generative-adversarial-neural-networks.html. [Accessed: 25- Nov- 2018].