MIE1516 Project Final Report

Image Super Resolution using Deep Learning Networks

Name: Meghana Padmanabhan

Student #: 1000905210

Utorid: padmana5

GithubId: MeghanaP

1.0 Introduction

Image Super Resolution (SR), one of the important tasks in image restoration, is the process of "estimating a high resolution image from a low resolution image"[1]. It has many applications in restoring aerial images (such as satellite imagery), long distance facial recognition, medical MRI imaging, old photographs and blurred images and video [2].

The project explored and investigated four different architectures – SRCNN MSE, SRCNN PL, SRGAN MSE, SRGAN PL to analyze the effect of the loss functions: Pixel wise MSE loss (MSE) and Perceptual Loss (PL) on the super resolution images generated by the architectures with respect to the target high resolution images.

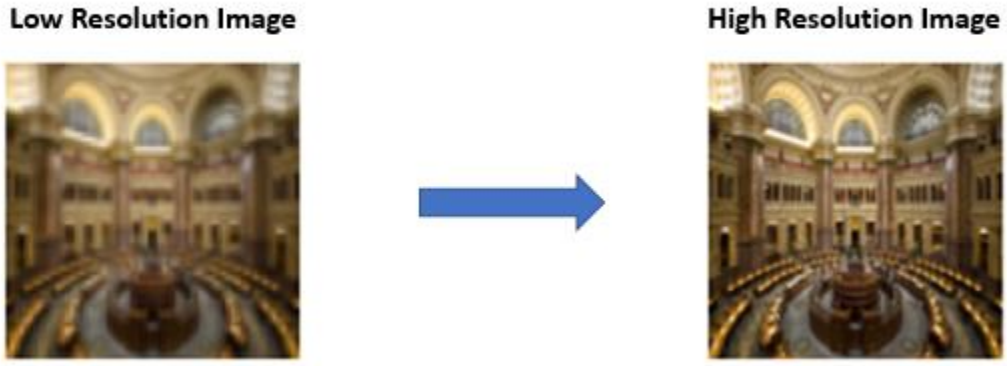

Figure 1: Image Super Resolution Task

2.0 Methodology

In order to generate the super resolution (SR) image, two types of architectures were implemented – Convolutional Neural Networks (CNN) and Generative Adversarial Networks (GAN). Each type of architecture was trained on two different loss functions- MSE and Perceptual Loss. Hence, the resulting SR architectures investigated were – SRCNN MSE, SRCNN PL, SRGAN MSE and SRGAN PL. The following subsections will describe the neural network model representation and loss objective functions.

2.1 Image Representation

The goal is to generate a super resolution image $\boldsymbol{I^{SR}}$ from a low-resolution image input $\boldsymbol{I^{LR}}$so that the super resolution image is close to the high-resolution image ground truth or reference image $\boldsymbol{I^{HR}}$. The low-resolution images were obtained by down sampling the high-resolution images by a factor **r=4** (in our case r=4) with random flipping of pixels in the image. Hence, the dimensions of high-resolution image are $\boldsymbol{I^{LR}}$= W x H x C and $\boldsymbol{I^{SR}}$, $\boldsymbol{I^{HR}}$= r*W x r*H x C [1].

2.2 Super Resolution Image Generator Network Architecture

The SR image generator network is a feed forward convolutional network inspired from the state-of-the-art architecture-SRCNN. The first convolutional layer extracts patches from the low-resolution images into high dimensional feature map vectors.

The second convolutional layer establishes a non-linear mapping the high dimensional vectors to another high dimensional vectors which represents high resolution patches [3]. The third convolutional layer reconstructs the super resolution image to the same size as the input low resolution image, from the patches [3]. The super resolution output is passed through up-sampling blocks to up-sample the high-resolution output to the size of the target high resolution image [1].

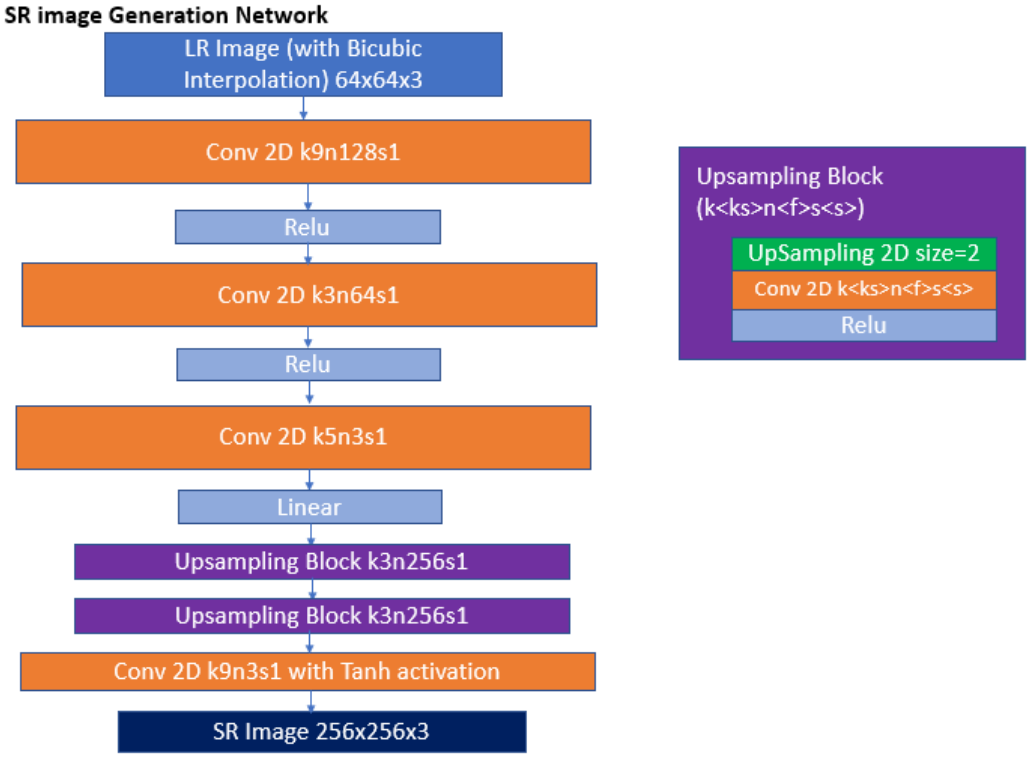

Fig 2: Super Resolution Image Generation Network in CNNs and GANs architectures [1][3]

The network is trained on a generator function **G** parametrized by $\boldsymbol{\theta_G}$ where $\boldsymbol{\theta_G}$ refers to the weights and bias variables trained in each layer of the generator network from layer 1 to **L**. Consider $\boldsymbol{l^{SR}}$ to be the loss function (more details covered in Section 2.4). Hence, for training samples of $\boldsymbol{I_n^{SR}}$ $\boldsymbol{I_n^{LR}}$ for n = 1 to N, the objective function to solve by the generator network is:

$$\widehat{\boldsymbol{\theta_G}} = arg \min_{\boldsymbol{\theta_G}} \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{l^{SR}} (\boldsymbol{G_{\theta_G}}(\boldsymbol{I_n^{LR}}), \boldsymbol{I_n^{HR}}) \qquad \text{(a)[1]}$$

where $\boldsymbol{G_{\theta_G}}(\boldsymbol{I_n^{LR}})$ is the super resolution image generated by the network and objective function aims to obtain the parameters which minimize the loss between the generated SR image $\boldsymbol{G_{\theta_G}}(\boldsymbol{I_n^{LR}})$ and the target HR image $\boldsymbol{I_n^{HR}}$[1].

2.3 Adversarial/Discriminator Network Architecture for GAN based architectures

In traditional CNN architectures for generation of super resolution images such as the network architecture described in section 2.2, the output of the network is a generated super resolution image whose quality can be assessed against the reference high resolution image using metrics highlighted in section 2.5.

In contrast, there is another class of deep learning networks called Generative Adversarial Network (GAN) which is a combined model of two networks – Generator network which is a feed forward CNN network to generate the super resolution image and Discriminator network which is a binary classification network determining whether the super resolution image generated by the generator network is real or fake i.e. how closely does it match the ground truth or reference high-resolution image[1].

The discriminator network implemented in the GAN based architectures inspired from the state of the art SRGAN model is as follows:

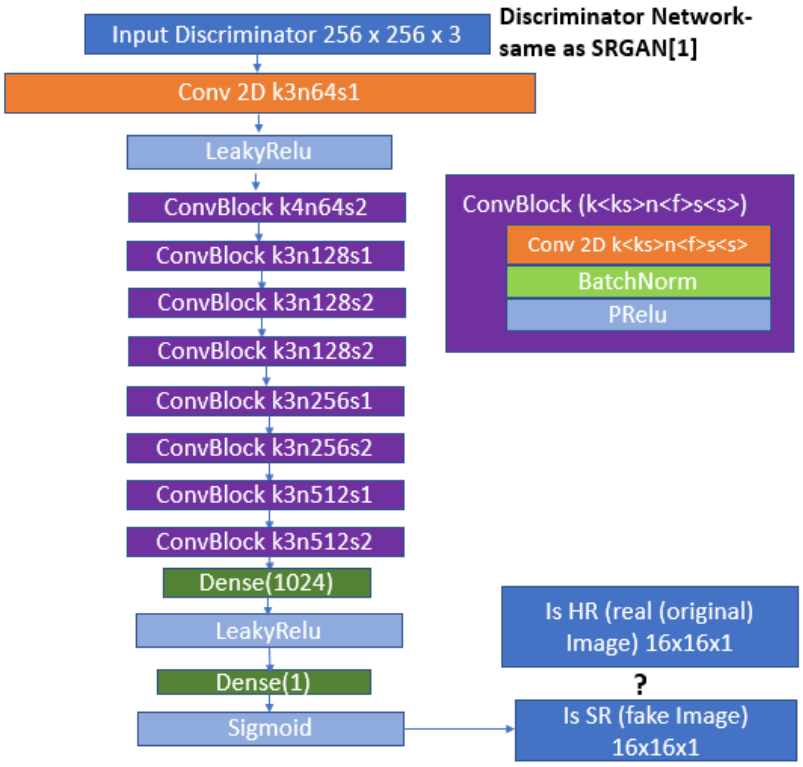

Fig 3: Discriminator Network in GAN architectures [1]

The above network consists of 8 convolutional block layers which increase the filter kernels by size of 2 from 64 to 512 kernels. "Stride convolutional layers" such as the convolutional block layers in the above network, are used to reduce the resolution of image input as the number of filter kernels increase. The 512 feature maps are then passed through two dense layers and sigmoid activation to determine the probability of the sample being classified as fake(or real).

The objective function to solve by the discriminator network with function $D_{\boldsymbol{\theta_D}}$is the traditional adversarial min-max function:

$$\min_{\boldsymbol{\theta_G}} \max_{\boldsymbol{\theta_D}} \mathbb{E}_{\boldsymbol{I^{HR}\sim\ Ptrain(I^{HR})}}[\log D_{\boldsymbol{\theta_D}}(\boldsymbol{I^{HR}})] + \mathbb{E}_{\boldsymbol{I^{LR}\sim\ P_G(I^{LR})}}[\log(1 - D_{\boldsymbol{\theta_D}}(\boldsymbol{G_{\theta_G}}(\boldsymbol{I^{LR}}))] \quad \text{(b)[1]}$$

The intent behind this objective function is to train the generator network to produce super resolution images which are hard for the "differentiable discriminator" to classify a super resolution image from the ground truth [1].

2.4 Loss Functions $\boldsymbol{l^{SR}}$

2.4.1 Content Loss – Pixel wise MSE Loss

This is the most widely used optimization loss function by image super resolution networks. It calculates the pixel wise mean squared error between the ground truth and super resolution image shown in the equation as follows:

$$\boldsymbol{l^{SR}_{MSE}} = \frac{\mathbf{1}}{\boldsymbol{r^2WH}} \sum_{\boldsymbol{w=1}}^{\boldsymbol{rW}} \sum_{\boldsymbol{h=1}}^{\boldsymbol{rH}} (\boldsymbol{I^{HR}_{w,h}} - \boldsymbol{G_{\theta_G}}(\boldsymbol{I^{LR}_{w,h}}))^2 \quad \text{(c)[1]}$$

2.4.2 Content Loss – VGG Feature Map Loss

An alternative to MSE based loss, is a VGG loss which involves extracting the feature map $\boldsymbol{\varphi_{f,g}}$ of a convolutional layer, **g** right before a max-pooling layer, **f** in a pre-trained VGG19 network (Figure 4) and calculating the Euclidean distance between the feature representations of the ground truth and super resolution image as follows:

$$\boldsymbol{l^{SR}_{VGG\backslash f,g,}} = \frac{\mathbf{1}}{\boldsymbol{W_{f,g}H_{f,g}}} \sum_{\boldsymbol{w=1}}^{\boldsymbol{W_{f,g}}} \sum_{\boldsymbol{h=1}}^{\boldsymbol{H_{f,g}}} (\boldsymbol{\varphi_{f,g}}(\boldsymbol{I^{HR}_{w,h}})_{\boldsymbol{f,g}} - \boldsymbol{\varphi_{f,g}}(\boldsymbol{G_{\theta_G}}(\boldsymbol{I^{LR}_{w,h}}))_{\boldsymbol{f,g}})^2 \quad \text{(d)[1]}$$

where $\boldsymbol{W_{f,g}}, \boldsymbol{H_{f,g}}$ are the dimensions of the respective feature maps in VGG19 network.

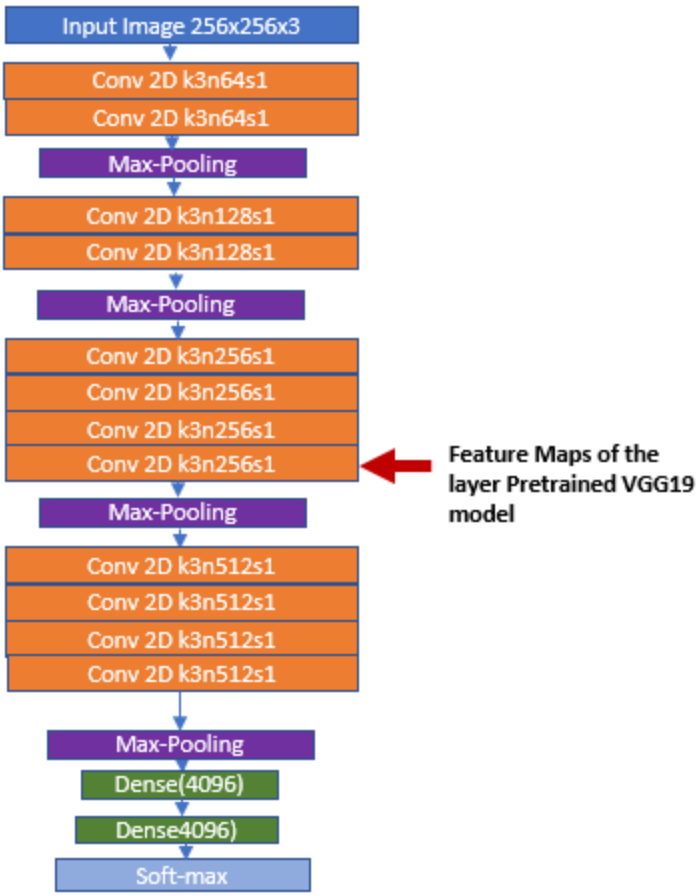

Fig 4: VGG19 Network [4]

2.4.3 Adversarial Loss

The adversarial loss is specific to GAN's discriminator network where we want to encourage the generator to produce super resolution images close to the ground truth such that it "fools" the discriminator network to not classify the super-resolution images as "fake". Hence, we can define an adversarial loss for the generator in terms of the "discriminator network probabilities $D_{\theta_D}(\boldsymbol{G}_{\theta_G}(\boldsymbol{I}^{\boldsymbol{LR}}))$ over all N training samples"[1] as follows:

$$\boldsymbol{l}_{\boldsymbol{GEN}}^{\boldsymbol{SR}} = \sum_{n=1}^{N} -\log(\, D_{\boldsymbol{\theta_D}}(\boldsymbol{G}_{\boldsymbol{\theta_G}}(\boldsymbol{I}^{\boldsymbol{LR}})) \qquad \text{(e)[1]}$$

2.4.4 Perceptual Loss

Perceptual Loss aims at capturing perceptual similarities between the super resolution and high-resolution images. Depending on the type of architecture, perceptual loss has different meanings.

For pure CNN networks, the perceptual loss is the content loss – VGG loss as described in section 2.4.2. It is presented as follows:

$$\boldsymbol{l}^{\boldsymbol{SR}}{}_{\boldsymbol{PL}} = \boldsymbol{l}^{\boldsymbol{SR}}{}_{\boldsymbol{VGG}\backslash \boldsymbol{f},\boldsymbol{g}} \qquad \text{(f)}$$

For GAN networks, the perceptual loss is defined as the weighted sum of content loss (section 2.4.2) and adversarial loss (section 2.4.3). It is presented as follows:

$$\boldsymbol{l^{SR}}_{\boldsymbol{PL}} = \boldsymbol{l^{SR}}_{\boldsymbol{VGG\backslash f,g}} + \mathbf{1e-3}\boldsymbol{l^{SR}_{GEN}} \quad \text{(g)[1]}$$

2.5 Image Quality Metrics

2.5.1 PSNR - Peak Signal to Noise Ratio

Ratio of quality measured between SR (predicted/generated) and HR (ground truth/reference image) images. Higher the value, better is the quality of SR image[5]. It is modelled around MSE loss as follows:

$$PSNR\left(\boldsymbol{I^{HR}}, \boldsymbol{G_{\theta_G}}(\boldsymbol{I^{LR}})\right) = \mathbf{10}\log_{\mathbf{10}}\mathbf{255^2}/\boldsymbol{MSE}(\boldsymbol{I^{HR}}, \boldsymbol{G_{\theta_G}}(\boldsymbol{I^{LR}})) \quad \text{(h)[6]}$$

where $\boldsymbol{MSE}(\boldsymbol{I^{HR}}, \boldsymbol{G_{\theta_G}}(\boldsymbol{I^{LR}}) = \boldsymbol{l^{SR}_{MSE}}$ (from eq (c) in Section 2.4.1)

2.5.2 SSIM- Structural Similarity Index

It assesses luminance, contrast and structure of the SR image (predicted/generated) when compared with HR (ground truth/reference image) [7]. Higher the value, better is the quality of SR image. Range of SSIM value is [-1,1][7]. The following presents the components that make up the SSIM:

Let $\boldsymbol{p} = \boldsymbol{I^{HR}}$, $\boldsymbol{q} = \boldsymbol{G_{\theta_G}}(\boldsymbol{I^{LR}})$

Luminance Comparison Function, $L(\boldsymbol{p},\boldsymbol{q}) = \frac{\mathbf{2}\boldsymbol{\mu_p}\boldsymbol{\mu_q}+\boldsymbol{C_1}}{\boldsymbol{\mu_p}^2+\boldsymbol{\mu_q}^2+\boldsymbol{C_1}}$ (i)[6]

where $\boldsymbol{\mu_p}$, $\boldsymbol{\mu_q}$ are mean luminance of the HR and SR images respectively. The factor is maximal (=1) only when $\boldsymbol{\mu_p} = \boldsymbol{\mu_q}$ [6].

Contrast Comparison Function, $C(\boldsymbol{p},\boldsymbol{q}) = \frac{\mathbf{2}\boldsymbol{\sigma_p}\boldsymbol{\sigma_q}+\boldsymbol{C_2}}{\boldsymbol{\sigma_p}^2+\boldsymbol{\sigma_q}^2+\boldsymbol{C_2}}$ (j)[6]

where $\boldsymbol{\sigma_p}$, $\boldsymbol{\sigma_q}$ standard deviation of HR and SR images respectively used in the measure of contrast. The factor is maximal and equal to 1 only if $\boldsymbol{\sigma_p} = \boldsymbol{\sigma_q}$ [6].

Structure Comparison Function, $S(\boldsymbol{p},\boldsymbol{q}) = \frac{\boldsymbol{\sigma_{pq}}+\boldsymbol{C_3}}{\boldsymbol{\sigma_p}\boldsymbol{\sigma_q}+\boldsymbol{C_3}}$ (k)[6]

where $\boldsymbol{\sigma_{pq}}$ is the covariance between $\boldsymbol{p}$ and $\boldsymbol{q}$. The factor "measures the correlation coefficient between the two images"[6].

$$SSIM(\boldsymbol{p},\boldsymbol{q}) = L(\boldsymbol{p},\boldsymbol{q}) * C(\boldsymbol{p},\boldsymbol{q}) * S(\boldsymbol{p},\boldsymbol{q}) \quad \text{(l)[6]}$$

3.0 Experiments

The following sub-sections will describe the experiments conducted.

3.1 Training Settings

3.1.1 Dataset

DIV2K High Resolution Image Datasets were used in the training. [8][9]
Training Dataset: "DIV2K/DIV2K_train_HR/" which contains 800 images.
Testing/Validation Dataset: "DIV2K/DIV2K_valid_HR/" which contains 100 images.
The high-resolution images were resized to 256x256x3 and low-resolution images were generated by down sampling using bicubic interpolation by a factor of 4 with random flipping/rearranging (noise) of pixels [Appendix A].

3.1.2 Codebase

Please refer to Appendix A for the code reference and details regarding the code files.

3.1.3 Other settings

The following are the common settings used in training the architectures.
Optimizer settings : Adam(learning_rate=0.0002, beta_1=0.5)
Batch Size: 10
Epochs: 201

3.1.4 Runtime Hardware

Google Colab GPU: Nvidia Tesla K80

3.2 Architectures Investigated and their Loss functions

The following subsections will present details of the architectures investigated and their loss functions starting in ascending order of model complexity.

3.2.1 Architecture 1: SR CNN MSE

The architecture consists of image SR generator network describes in section 2.2 run under the conditions described in section 3.1. The loss function for the SR generator network was $\boldsymbol{l}_{\boldsymbol{MSE}}^{\boldsymbol{SR}}$ (from eq (c) in Section 2.4.1).

The following images show the generated super-resolution images at the start of training (0 epochs) and end of training (200 epochs). Please refer to Appendix B for detailed snapshots captured for other intermediate epochs.

Figure 5: SRCNN MSE Generated SR Image after 0 epochs of training

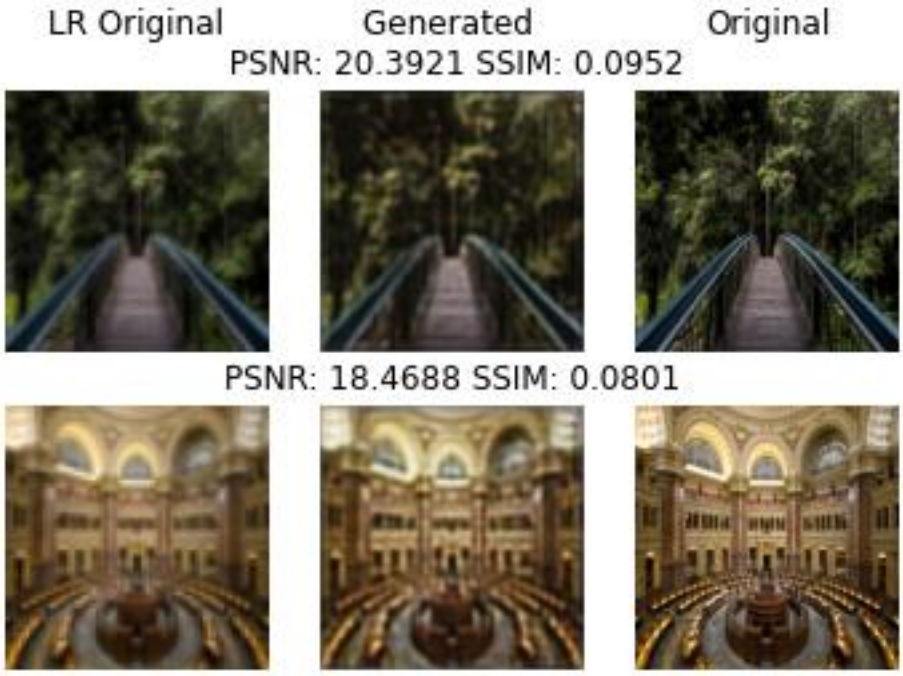

Figure 6: SRCNN MSE Generated SR Image after 200 epochs of training

3.2.2 Architecture 2: SR CNN PL

The architecture consists of SR image generator network described in section 2.2 run under the conditions described in section 3.1. The loss function for the SR generator network was the perceptual loss function of CNN, $\boldsymbol{l^{SR}}_{\boldsymbol{PL}} = \boldsymbol{l^{SR}}_{\boldsymbol{VGG\backslash f,g}}$ (from eq (f) in Section 2.4.4). A pretrained VGG19 network was implemented and the convolutional layer feature map before the third max-pooling layer (see model in section 2.4.2) was used in the loss calculation.

The following images show the generated super-resolution images at the start of training (0 epochs) and end of training (200 epochs). Please refer to Appendix C for detailed snapshots captured for other intermediate epochs.

Figure 8: SRCNN PL Generated SR Image after 0 epochs of training

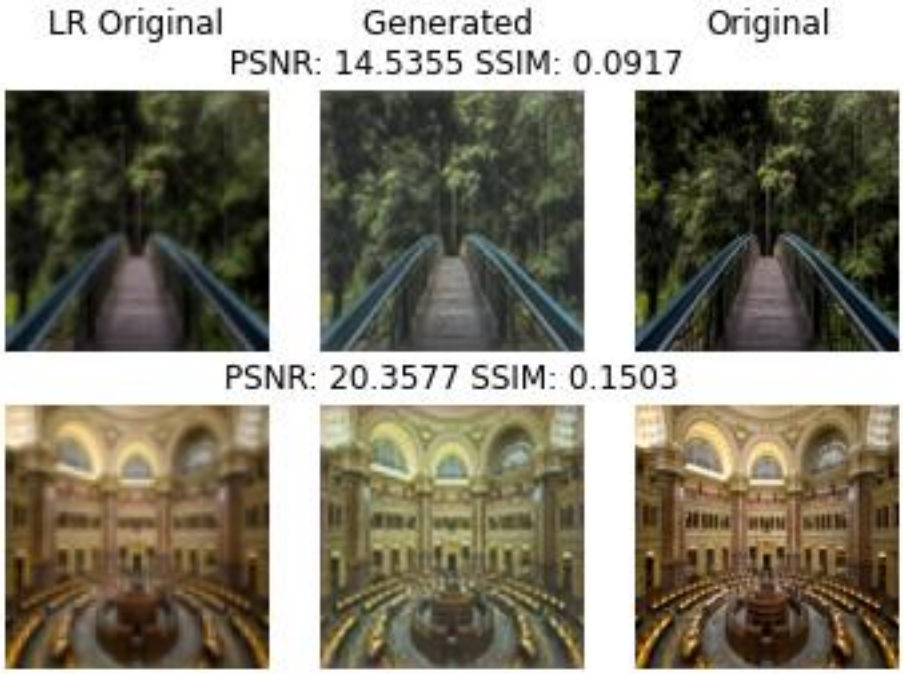

Figure 9: SRCNN PL Generated SR Image after 200 epochs of training

3.2.3 Architecture 3: SR GAN MSE

The architecture advances from the traditional CNN architecture discussed in section 3.2.1 and 3.2.2 to more complex neural network -> GANs. SR GAN MSE consists of SR image generator network described in section 2.2 and a discriminator network described in section 2.3 run under the conditions described in section 3.1.

The loss function for the SR image generator network was a slight modification to the perceptual loss function of GAN (from eq (f) in Section 2.4.4),

$$l^{SR}{}_{GAN\_MSE} = l^{SR}{}_{MSE} + 1e-3 l^{SR}_{GEN} \qquad \text{(m)}$$

The VGG content loss is replaced with pixel wise MSE loss described in eq(c) in Section 2.4.1. The loss function is hence, a weighted combination of content MSE loss and the adversarial loss (due to the addition of the discriminator network.

The following images show the generated super-resolution images at the start of training (0 epochs) and end of training (200 epochs). Please refer to Appendix D for detailed snapshots captured for other intermediate epochs.

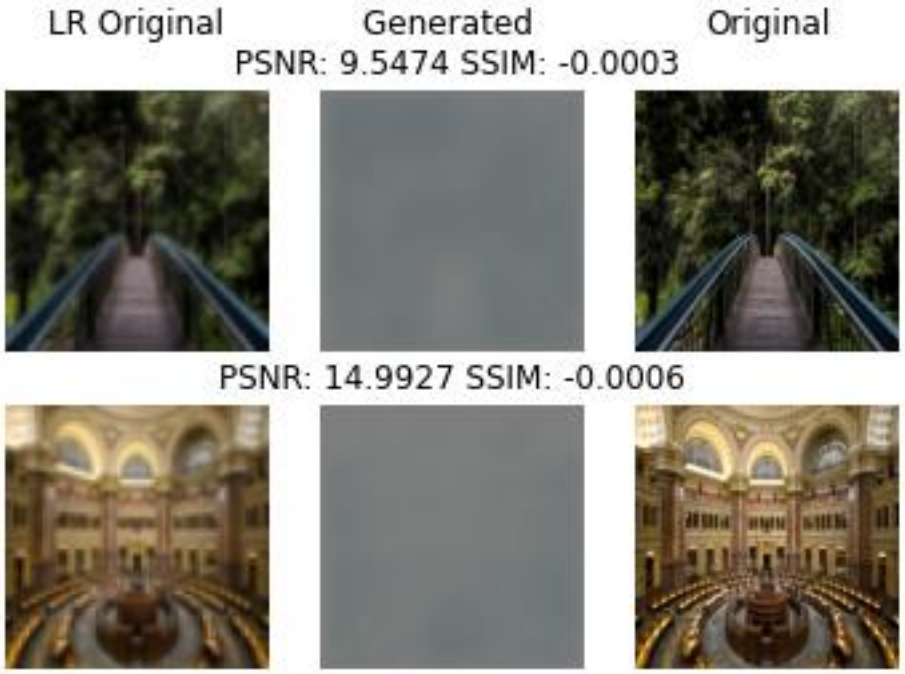

Figure 10: SRGAN MSE Generated SR Image after 0 epochs of training

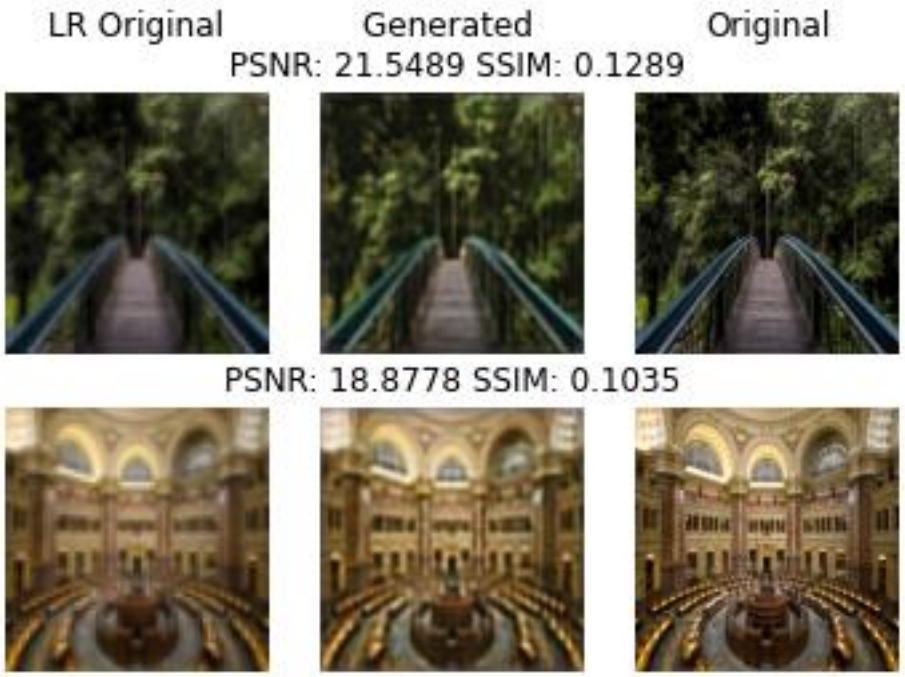

Figure 10: SRGAN MSE Generated SR Image after 200 epochs of training

3.2.4 Architecture 4: SR GAN PL

SR GAN PL consists of SR image generator network described in section 2.2 and a discriminator network described in section 2.3 run under the conditions described in section 3.1.

Similar to SRCNN PL (section 3.2.2), the loss function for the SR generator network was the perceptual loss function of GAN, $\boldsymbol{l^{SR}}_{\boldsymbol{PL}} = \boldsymbol{l^{SR}}_{\boldsymbol{VGG\backslash f,g}} + \mathbf{1e-3}\boldsymbol{l^{SR}_{GEN}}$ (from eq (g) in Section 2.4.4). A pretrained VGG19 network was implemented and the convolutional layer feature map before the third max-pooling layer (see model in section 2.4.2) was used in the loss calculation.

The following images show the generated super-resolution images at the start of training (0 epochs) and end of training (200 epochs). Please refer to Appendix E for detailed snapshots captured for other intermediate epochs.

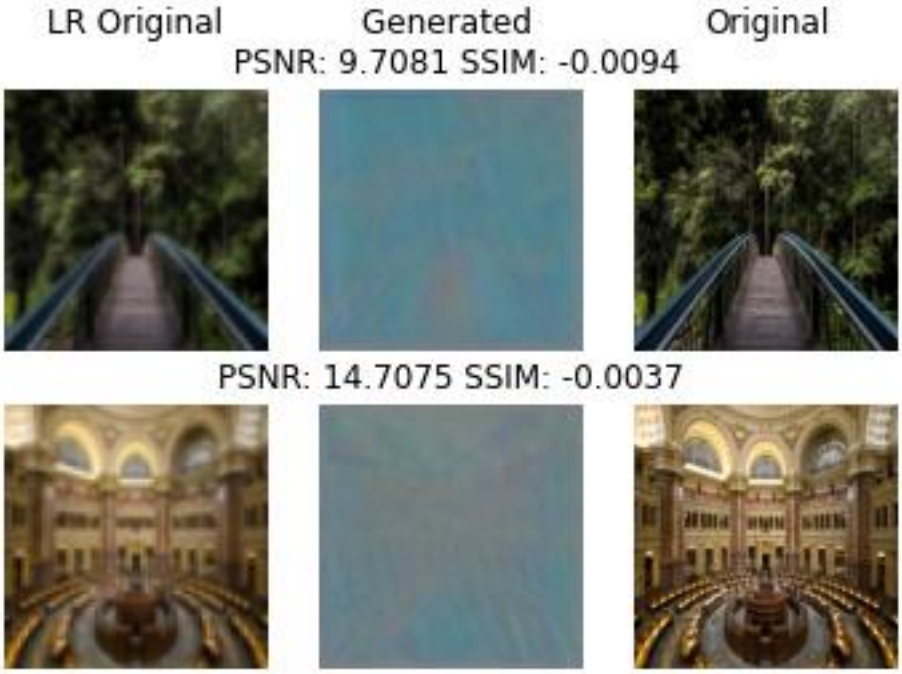

Figure 11: SRGAN PL Generated SR Image after 0 epochs of training

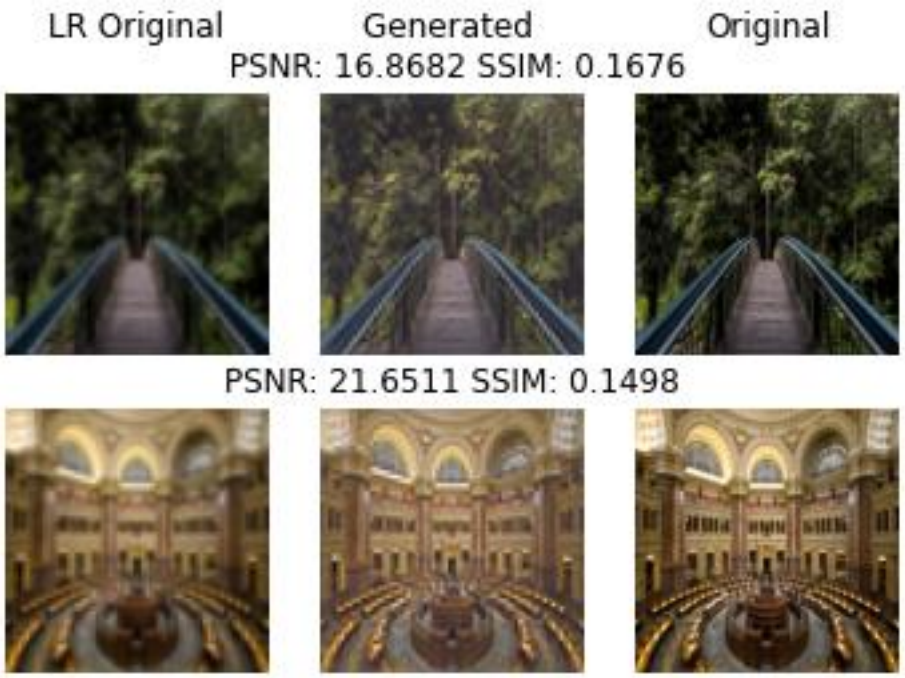

Figure 12: SRGAN PL Generated SR Image after 200 epochs of training

4.0 Results

4.1 Training and Validation Loss Plots - The SR Image Generator Network

The following subsections contains the training and validation loss plots for the SR Image Generator Network for each architecture.

4.1.1 Architecture 1: SR CNN MSE: *Best Train Loss: 0.0302 Valid Loss: 0.0321*

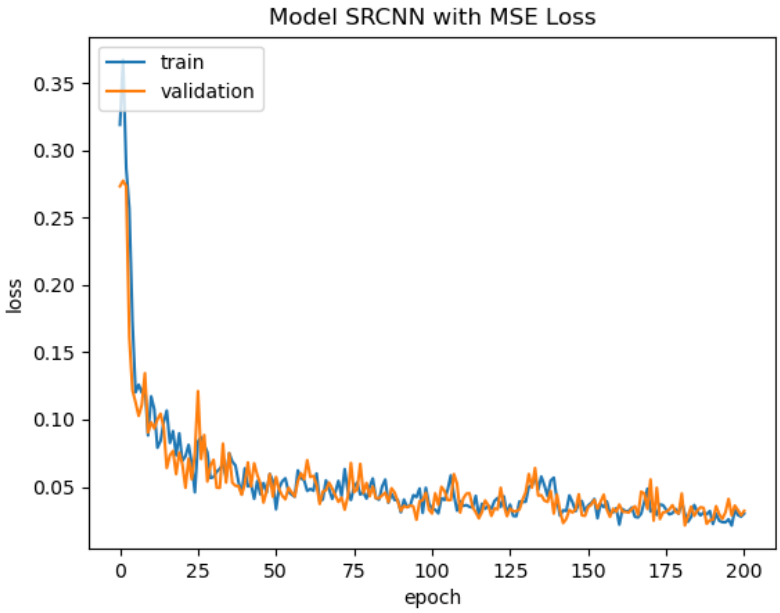

Figure 13: SR CNN MSE Loss Plot after 200 epochs of training

4.1.2 Architecture 2: SR CNN PL: *Best Train Loss: 22.2076 Valid Loss: 22.5765*

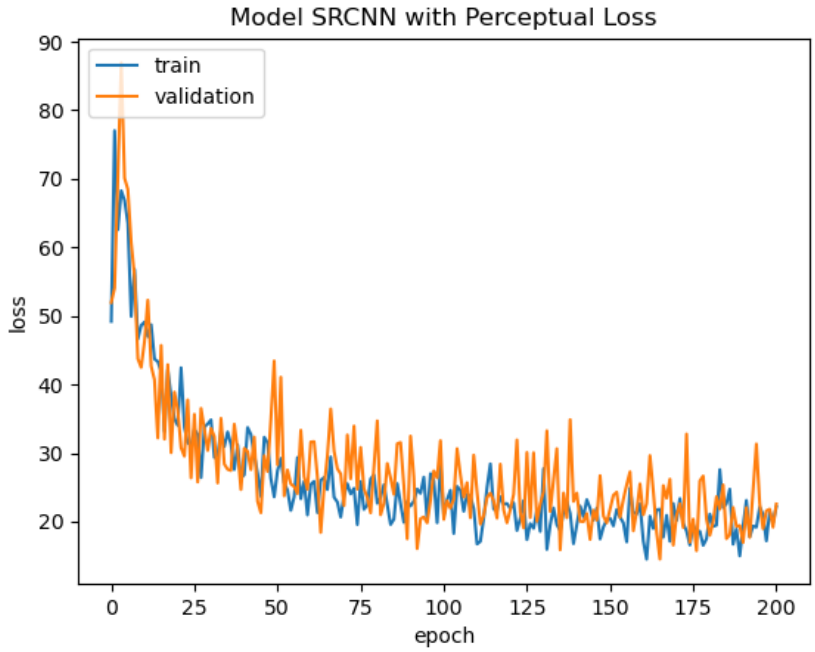

Figure 14: SR CNN PL Loss Plot after 200 epochs of training

4.1.3 Architecture 3: SR GAN MSE: *Best Train Loss: 0.0328 Valid Loss: 0.0221*

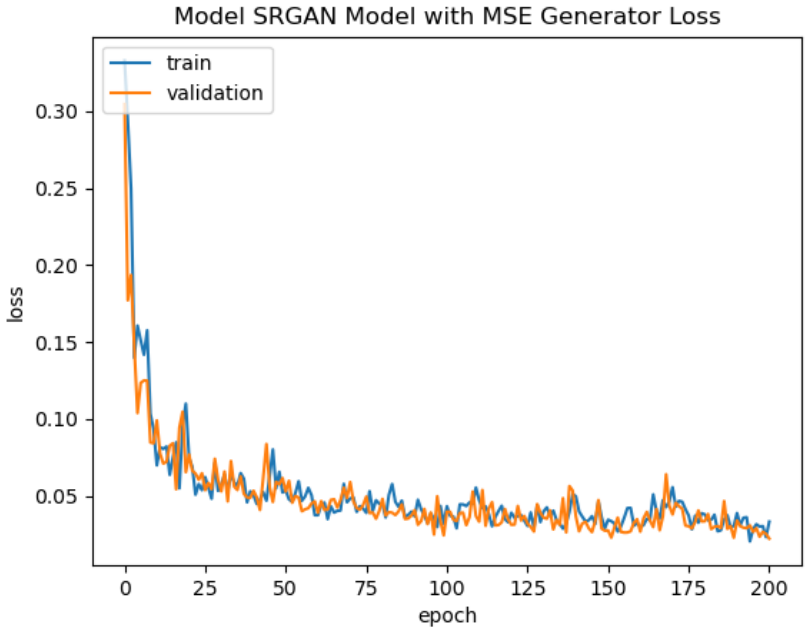

Figure 15: SR GAN MSE Loss Plot after 200 epochs of training

4.1.4 Architecture 4: SR GAN PL: *Best Train Loss: 19.9225 Valid Loss: 31.9291*

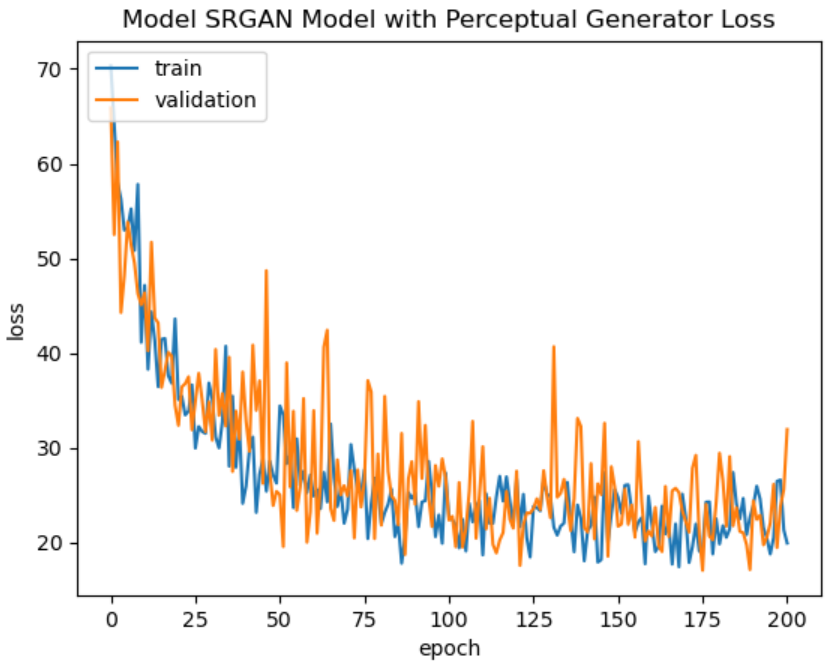

Figure 16: SR GAN PL Loss Plot after 200 epochs of training

4.1.5 Trends in Loss

4.1.5.1 By Loss Function – MSE vs Perceptual Loss (PL)

MSE loss based SR Generator Networks (SR CNN MSE and SR GAN MSE) are considerably lower than the networks which use Perceptual Loss.

This result is evident since Perceptual Loss goes beyond the rudimentary method of comparing pixel similarity between the generated SR image and the ground truth HR image. In the case of SRCNN PL, it utilizes VGG loss component where passing the SR and reference HR images through the feature maps of VGG19 help evaluate the edges, curves, shapes and other high frequency content[1] in the image unlike the pixel wise comparison of the images which MSE loss function performs. It also explains the noise in the loss plots for models optimizing a perceptual loss function than an MSE loss.

Similar trend is noticed in GAN based networks.

4.1.5.2 By Architecture – CNN vs GAN

Both architectures show similar loss trends for MSE and Perceptual Loss respectively. However, the plots in GAN based structures are more noisy since loss values are sensitive to not only the content loss but also the adversarial loss of the discriminator network which is not present in the CNN based models.

4.2 Image Quality Assessment across architectures: PSNR and SSIM

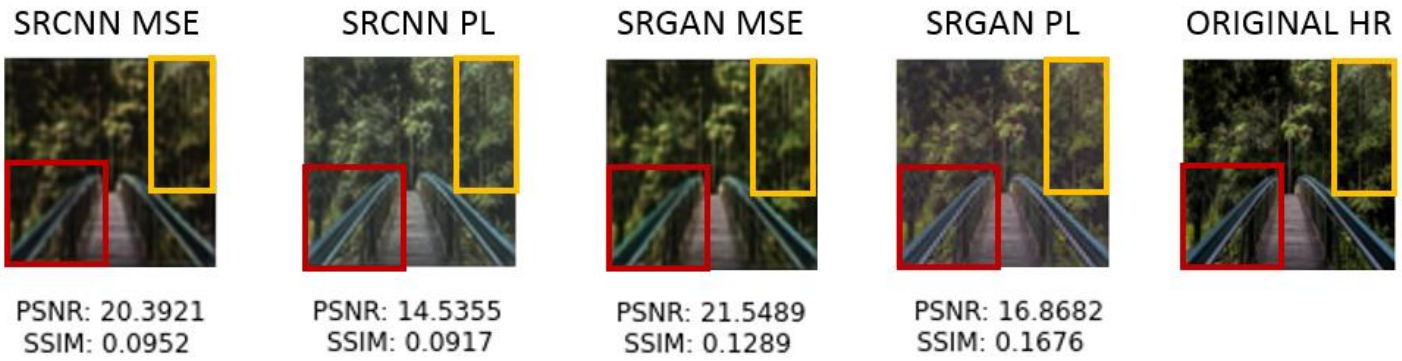

Figure 16: PSNR and SSIM Test Image 1

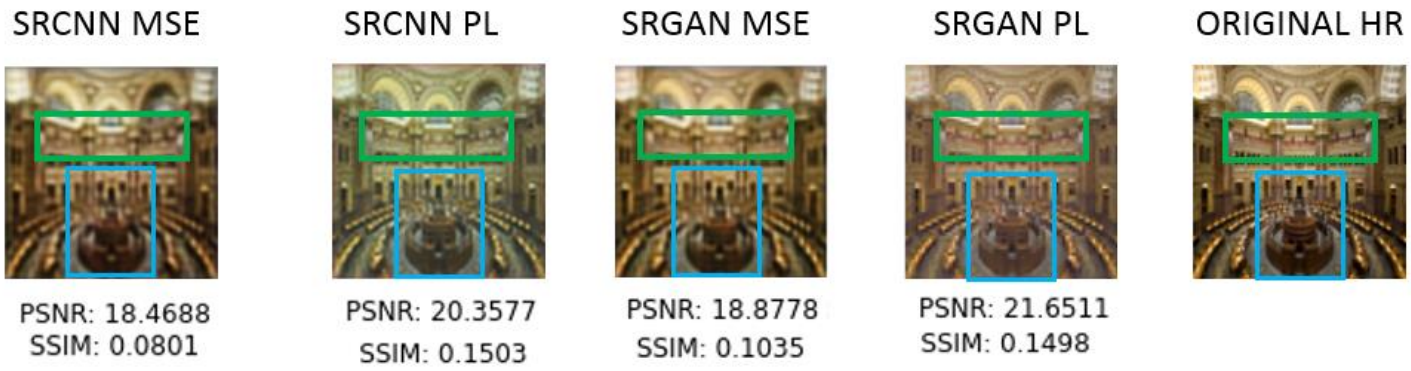

Figure 17: PSNR and SSIM Test Image 2

4.2.1 Quantitative Assessment:

4.2.1.1 PSNR: MSE based models had higher PSNR value but lower SSIM index value. From equation (h) in Section 2.5.2, one can see that PSNR is based of MSE Loss. Lower the pixel wise MSE Loss, higher the PSNR value.

The SSIM index value is low which proves that MSE value falls prey to over-smoothing the super resolution solutions as it compares only pixels and not measure for structural similarity i.e. comparing high frequency content (luminance, contrast, correlation) which SSIM index measures for.

4.2.1.2 SSIM: PL based models displayed higher SSIM values which indicate that structural similarity was established to a certain degree between the SR and original HR images which MSE based models couldn't achieve.

4.2.2 Qualitative Assessment:

4.2.2.1 Test Image 1

In Figure 16, in region highlighted in red, the bridge lines are well defined and close to the original image in the PL models compared to the MSE models which suffer from over smoothing in the generated SR image. Comparing the PL models, the bridge lines look "jagged/pixelated" in SRGAN PL compared to SRCNN PL whose lines were smoother. But the colors and contrast achieved by SRGAN PL's generated SR image is closest to the original HR image.

Similar can be said for the yellow section which highlights the leaves and thin trunks of the trees.

4.2.2.2 Test Image 2

In Figure 17, in region highlighted in green, the red lines under the arcs are well defined and close to the original image in the PL models compared to the MSE models which suffer from over smoothing in the generated SR image. The lines blend into the arcs in the case of the MSE models.

5.0 Conclusion and Recommendations

The project investigated the effect of two loss functions – MSE and Perceptual Loss using four architectures of varying model complexity for the task if Image super resolution. The results obtained do align with the observations made by state of the art image SR models such as the results published in paper *"Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network"*[1].

However, the generated SR images of the four architectures especially the GAN models still do not match with the original HR images as seen by the values of PSNR and SSIM metrics. It doesn't pass the qualitative assessment. This leaves room for more training for the models since they were trained only for 200 epochs and on a small dataset of 800 images. From the trends of the generated SR images during training given in Appendixes B,C,D,E, training the networks for more epochs has potential for improvement.

Assessment of the image quality of the generated SR image with respect to the original/reference HR image was done using PSNR and SSIM metrics. However, these metrics need to be accompanied with additional assessment by humans (subjective assessment of the images).

Technical challenges were met in training the models on Google Colab since these models take a long time to execute and would reach the free max GPU utilization capacity for the day thus preventing longer runs of the models.

In the proposal, another question regarding the inclusion of residual blocks was suggested[1]. Due to the technical challenges mentioned above, it couldn't be investigated in the scope of the project deadline. However, from literature, it has seen to have potential of improving the results obtained for the image super resolution task.

6.0 References

[1] C. Ledig, L. Theis, F. Huszar, J. Caballero, A.Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, W. Shi.(May 2016). Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network .Presentation at CVPR. [Online]. Available: https://arxiv.org/pdf/1609.04802.pdf.[Accessed April 13, 2020]

[2] A.Misra.(2018, Oct.) The End To All Blurry Pictures [Online].Available: https://towardsdatascience.com/the-end-to-all-blurry-pictures-f27e49f23588.[Accessed March 3, 2020]

[3] C. Dong, C.Change Loy, K. He, X.Tang.(July 2015). Image Super-Resolution Using DeepConvolutional Networks.[Online].Available: https://arxiv.org/pdf/1501.00092.pdf .[Accessed April 13, 2020]

[4] Han, Seung & Park, Gyeong & Lim, Woohyung & Kim, Myoung & Na, Jung-Im & Park, Ilwoo & Chang, Sung. (2018). Deep neural networks show an equivalent and often superior performance to dermatologists in onychomycosis diagnosis: Automatic construction of onychomycosis datasets by region-based convolutional deep neural network. PLOS ONE. 13. e0191493. 10.1371/journal.pone.0191493.

[5] The MathWorks, Inc.(2020) PSNR [Online].Available: https://www.mathworks.com/help/vision/ref/psnr.html.[Accessed April 13, 2020]

[6] Horé, Alain & Ziou, Djemel. (2010). Image quality metrics: PSNR vs. SSIM. 2366-2369. 10.1109/ICPR.2010.579.

[7] The MathWorks, Inc.(2020) ssim [Online].Available:https://www.mathworks.com/help/images/ref/ssim.html.[Accessed April 13, 2020]

[8] E. Agustsson, R. Timofte.(July 2017). NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study.Presented at The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops.[Online]. Available: http://people.ee.ethz.ch/~timofter/publications/Agustsson-CVPRW-2017.pdf.[Accessed April 13, 2020]

[9] R. Timofte et al., "NTIRE 2018 Challenge on Single Image Super-Resolution: Methods and Results," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Salt Lake City, UT, 2018, pp. 965-96511.

7.0 Appendices

7.1 Appendix A: Code Reference and File Description

The code was inspired from and built upon:

```
@misc{eriklindernoren,
    author       = {Erik Linder-Norén},
    title        = {{dust: Keras-GAN}},
    month        = aug,
    year         = 2019,
    publisher    = {Github},
    journal = {GitHub repository},
    howpublished = {\url{https://github.com/eriklindernoren/Keras-GAN}}
    }
```

My github repo: https://github.com/MeghanaP/ImageSuperResolutionUsingCNNGAN

It contains folder for each model with results and notebook on which its was trained. It also contains the commented notebooks for each model in the root folder of the repo.

7.2 Appendix B: SR CNN MSE Generated SR Images through epochs

Figure B1: SRCNN MSE Generated SR Image after 0 epochs of training

Figure B2: SRCNN MSE Generated SR Image after 50 epochs of training

Figure B3: SRCNN MSE Generated SR Image after 100 epochs of training

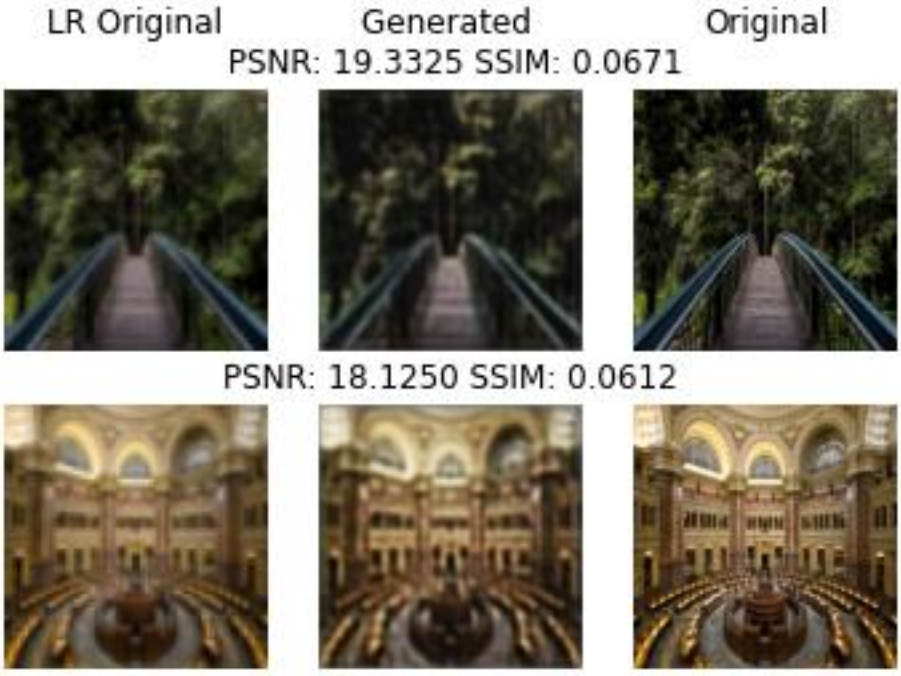

Figure B4: SRCNN MSE Generated SR Image after 150 epochs of training

Figure B5: SRCNN MSE Generated SR Image after 200 epochs of training

7.3 Appendix C: SR CNN PL Generated SR Images through epochs

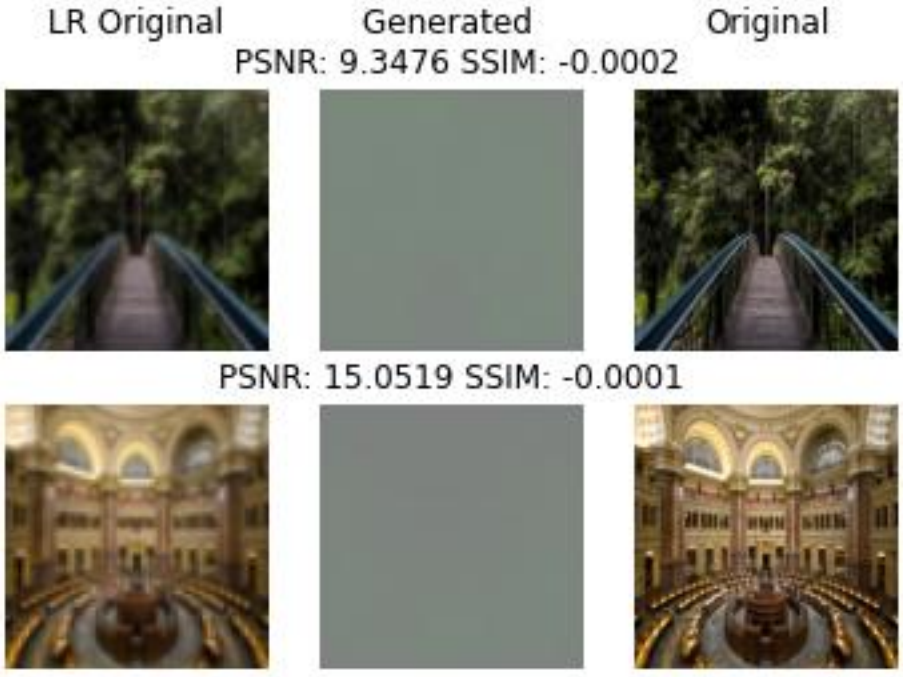

Figure C1: SRCNN PL Generated SR Image after 0 epochs of training

Figure C2: SRCNN PL Generated SR Image after 50 epochs of training

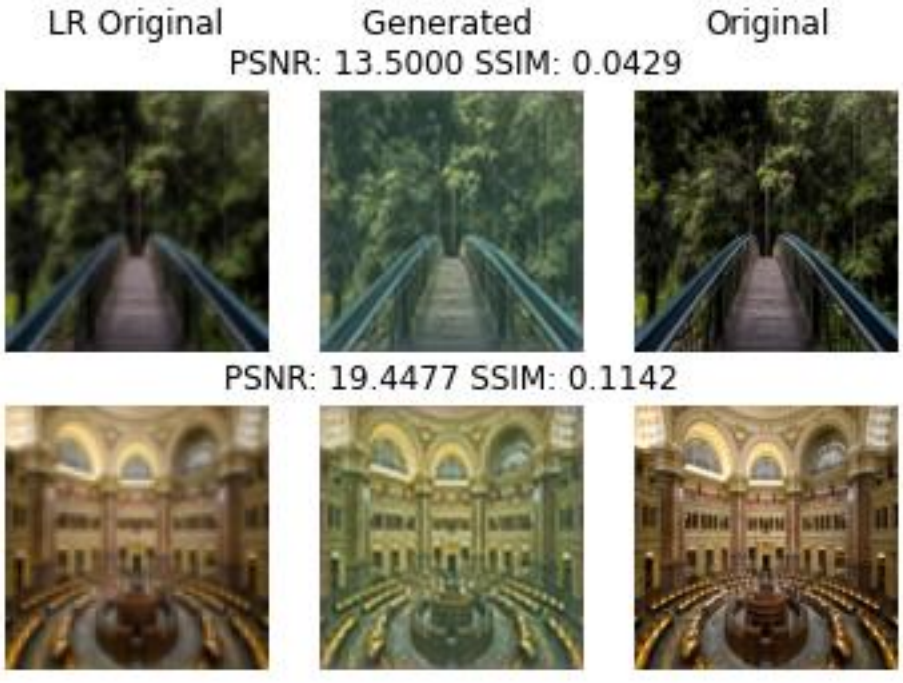

Figure C3: SRCNN PL Generated SR Image after 100 epochs of training

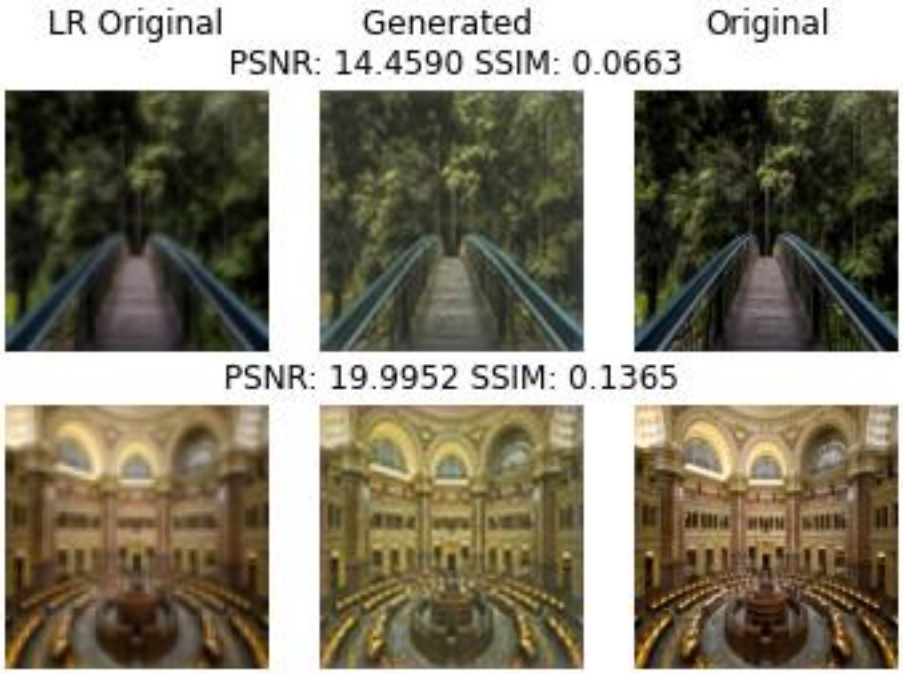

Figure C4: SRCNN PL Generated SR Image after 150 epochs of training

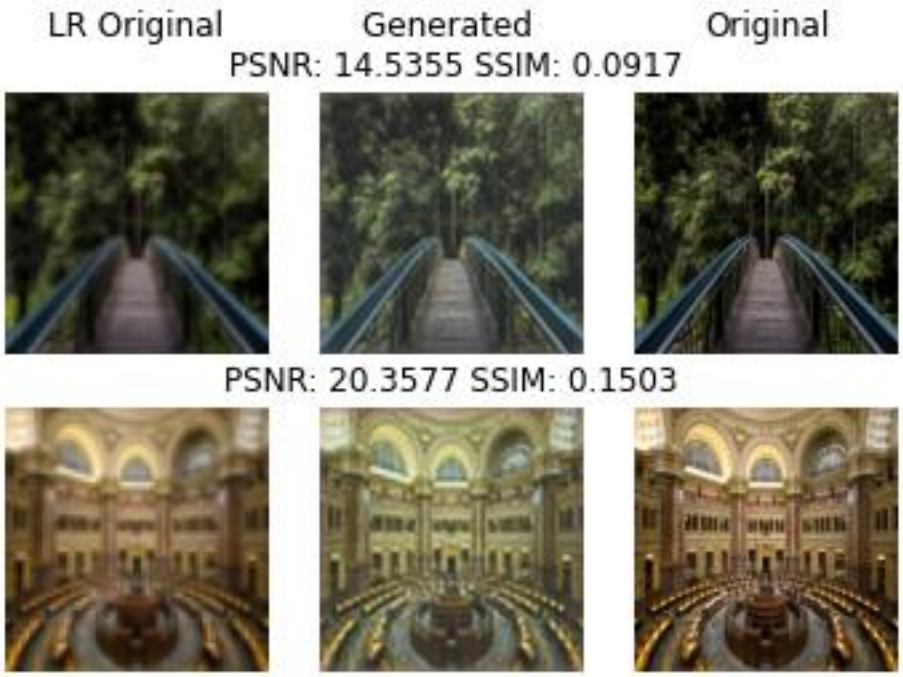

Figure C5: SRCNN PL Generated SR Image after 200 epochs of training

7.4 Appendix D: SR GAN MSE Generated SR Images through epochs

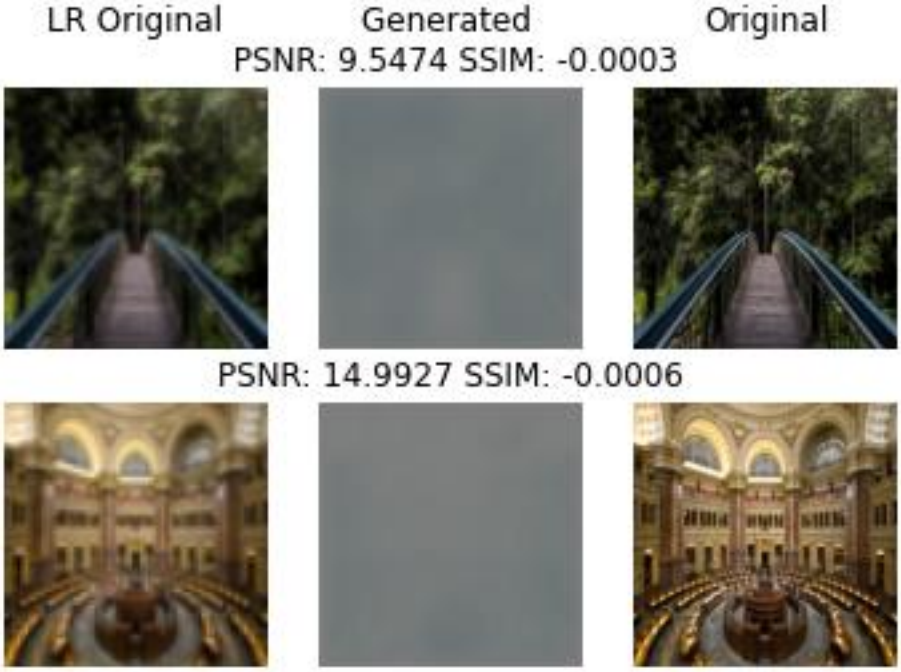

Figure D1: SR GAN MSE Generated SR Image after 0 epochs of training

Figure D2: SR GAN MSE Generated SR Image after 50 epochs of training

Figure D3: SR GAN MSE Generated SR Image after 100 epochs of training

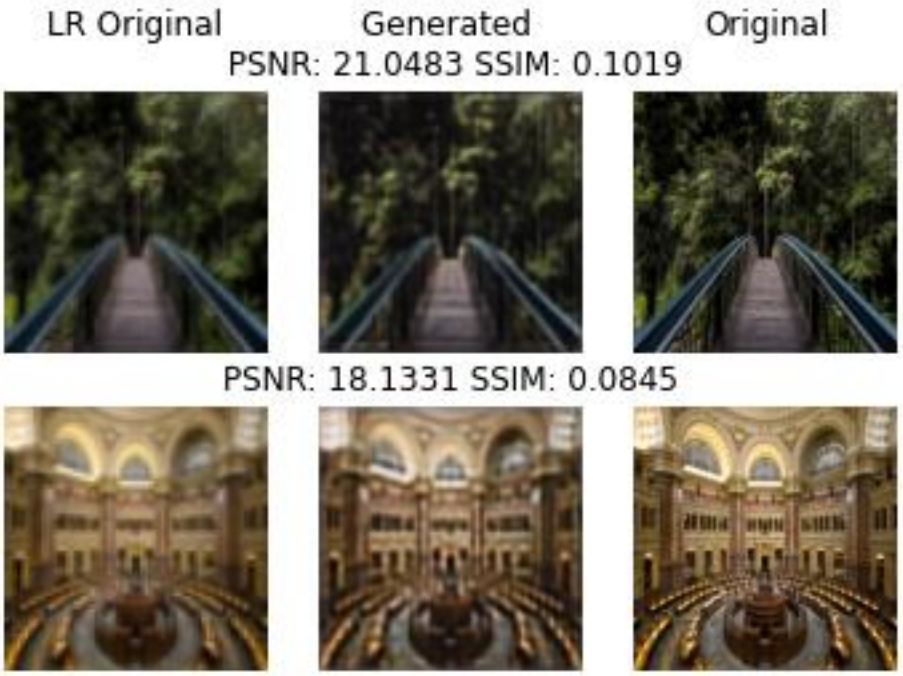

Figure D4: SR GAN MSE Generated SR Image after 150 epochs of training

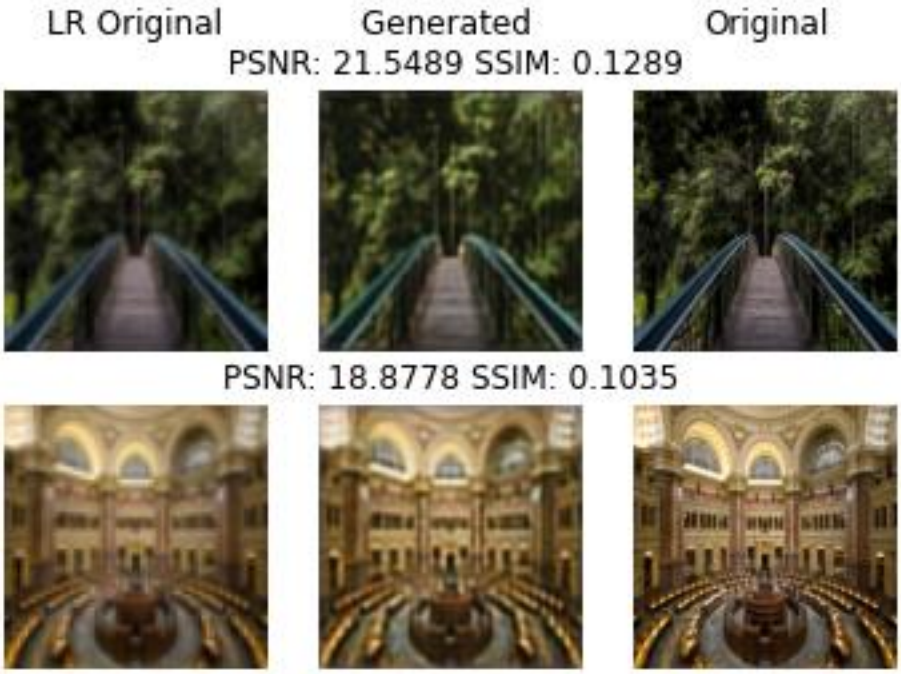

Figure D5: SR GAN MSE Generated SR Image after 200 epochs of training

7.5 Appendix E: SR GAN PL Generated SR Images through epochs

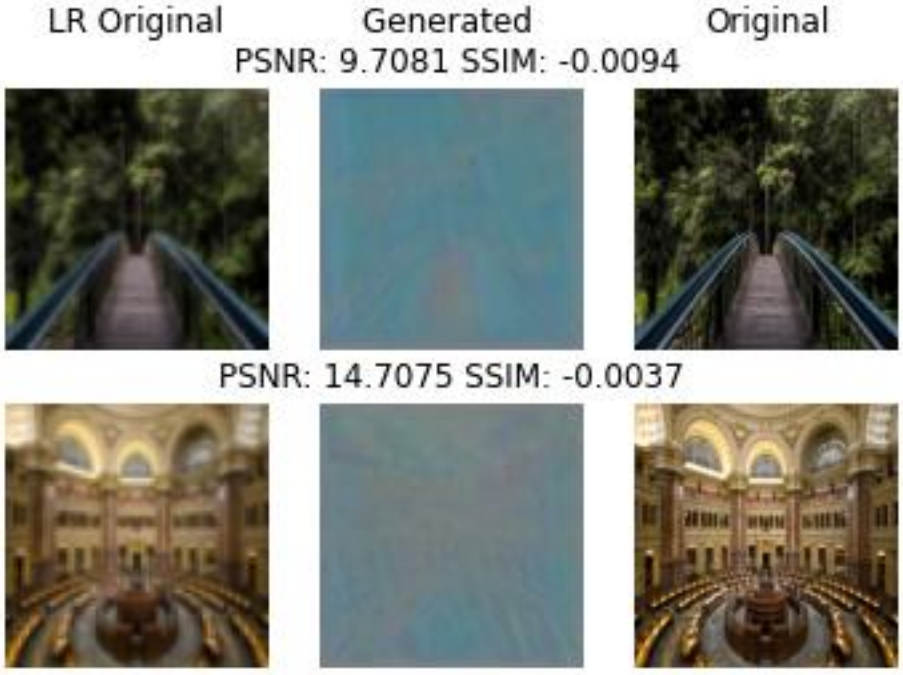

Figure E1: SR GAN PL Generated SR Image after 0 epochs of training

Figure E2: SR GAN PL Generated SR Image after 50 epochs of training

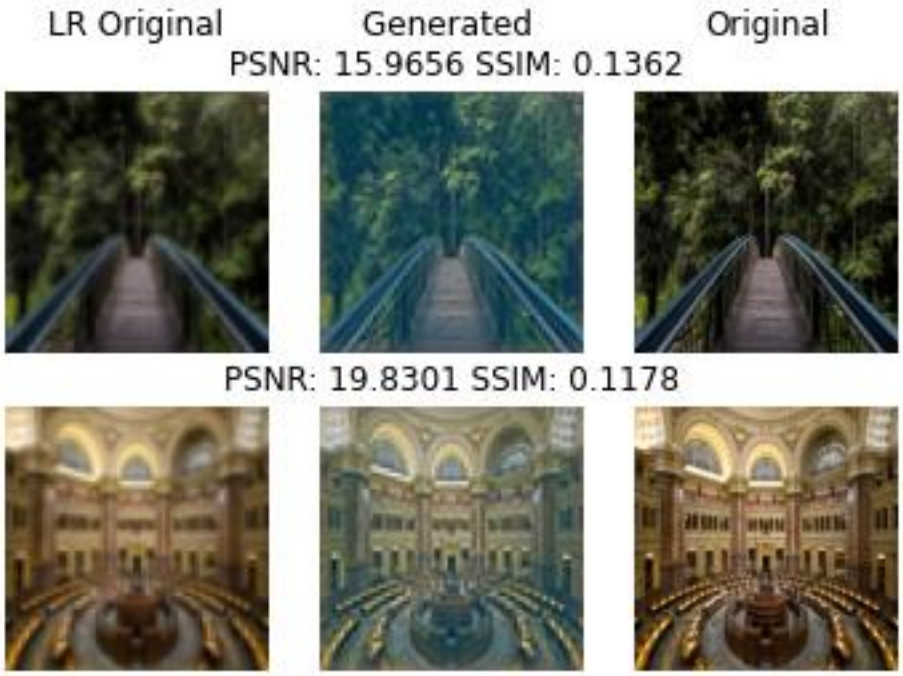

Figure E3: SR GAN PL Generated SR Image after 100 epochs of training

Figure E4: SR GAN PL Generated SR Image after 150 epochs of training

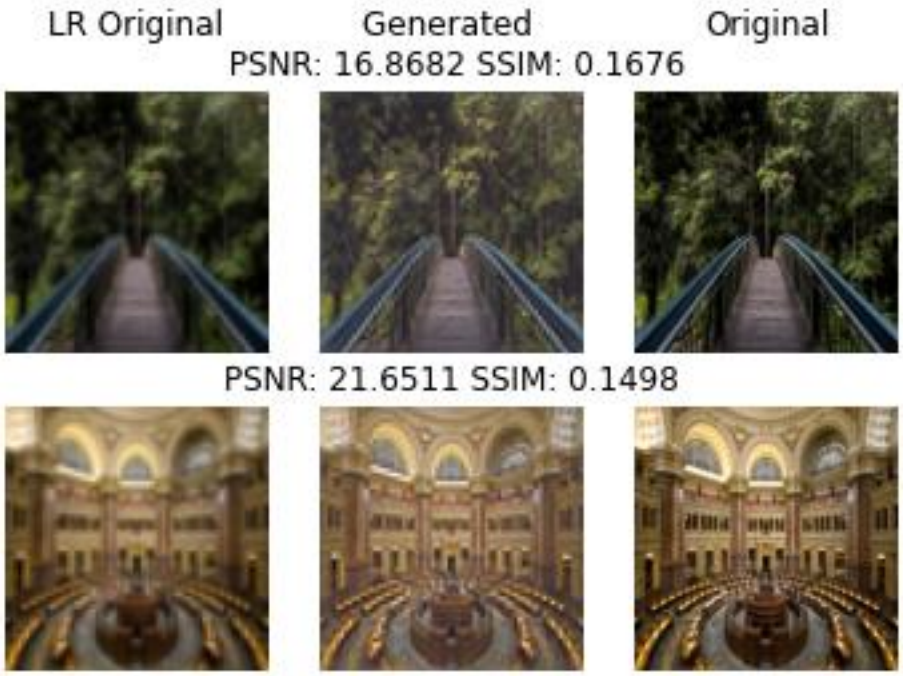

Figure E5: SR GAN PL Generated SR Image after 200 epochs of training

8.0