Assignment 2

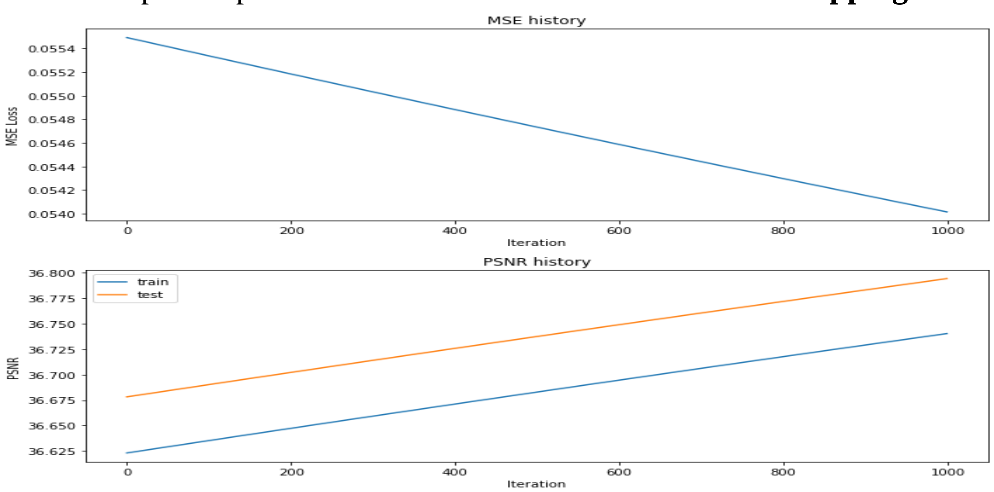
Name(s): Chiranjeevi Konduru, Rishabh Agrawal
NetID(s): konduru4, ra26

In each the following parts, you should insert the following:

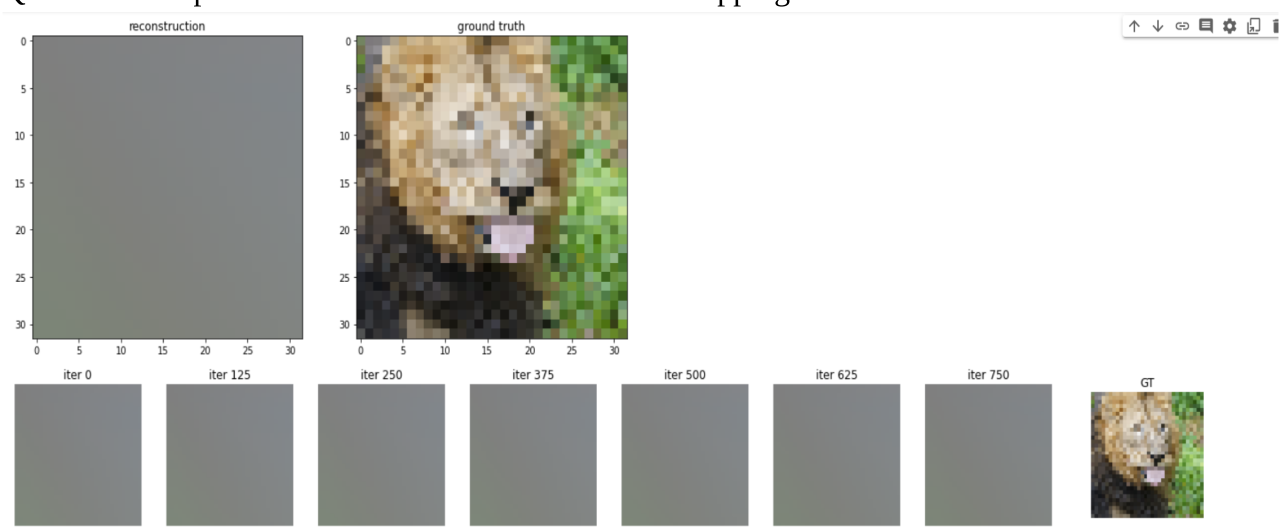
- Train/test loss plots
- Qualitative outputs for GT, No encoding, Basic Positional Encoding, and Fourier Feature Encoding at three different scales

Part 1: Low resolution example – SGD

The below plots represent the **Low Resolution – SGD None Mapping** train/test loss plots

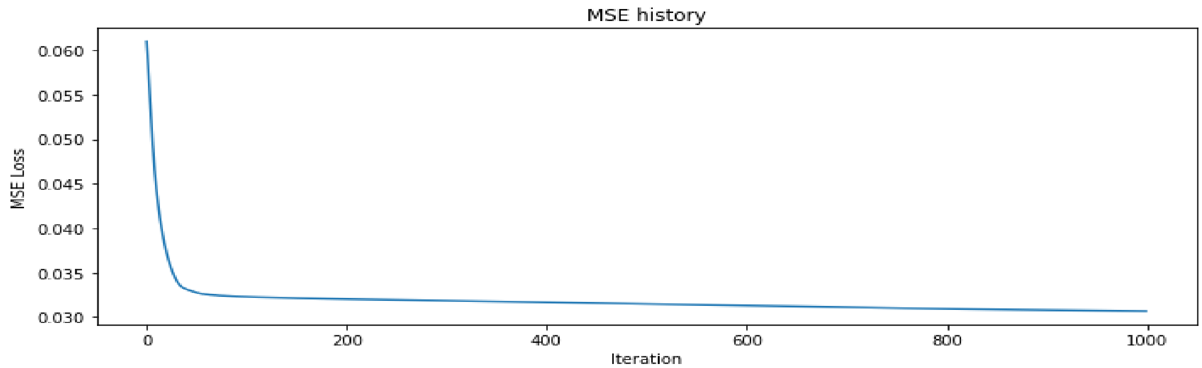


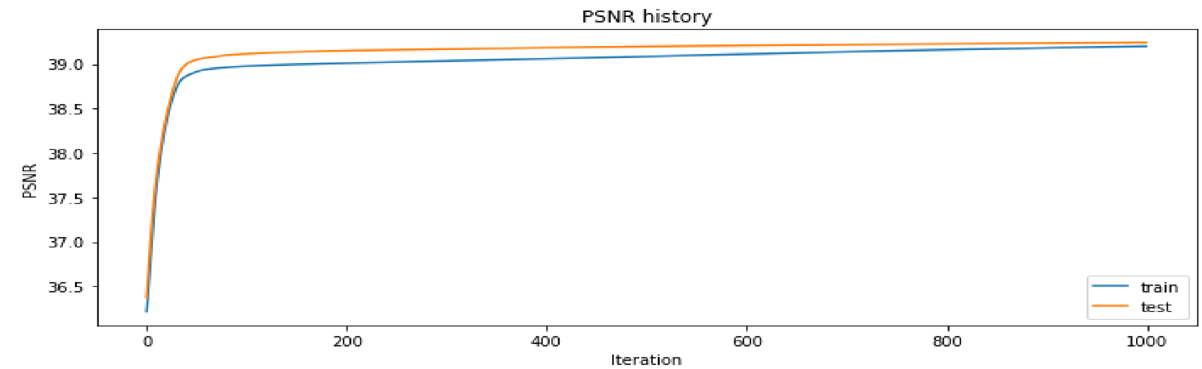
Qualitative output for Low Resolution – SGD None Mapping



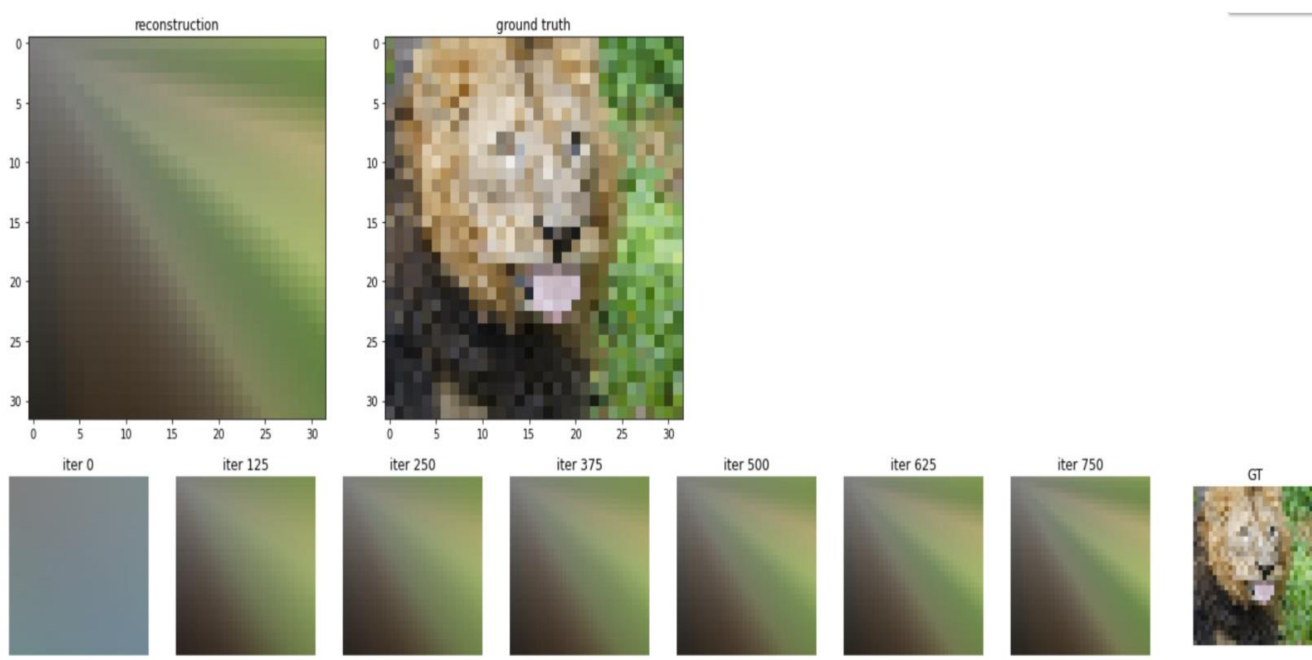
Part 2: Low resolution example – Adam

The below plots represents the **Low Resolution – Adam None Mapping** train/test loss plots

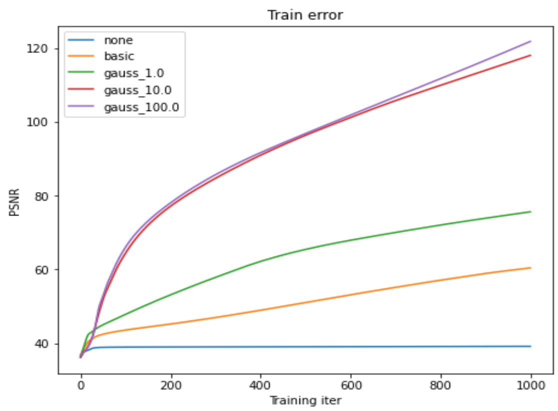


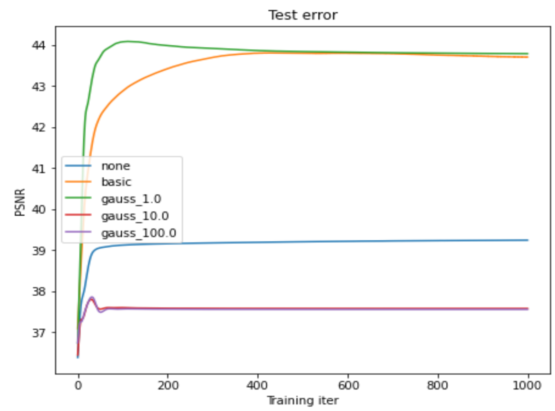


Qualitative output for Low Resolution – Adam None Mapping

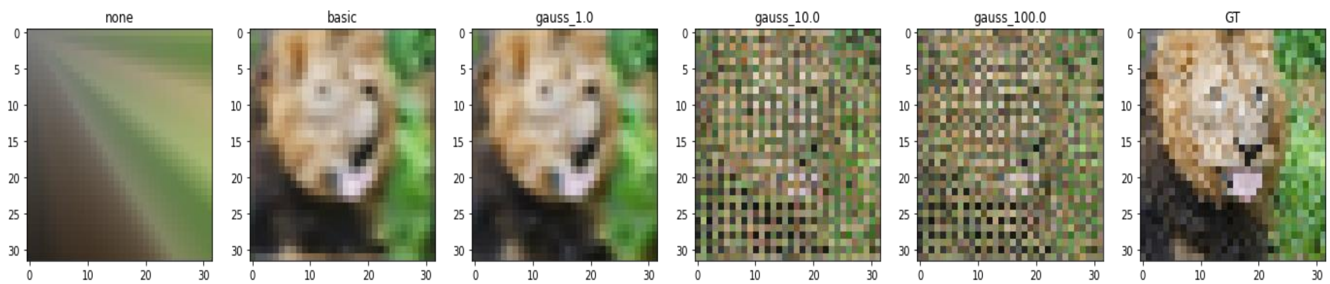


Part 3: Low resolution example (All mappings and Adam Optimizer)
The below plots represent the Low Resolution – All Mappings (none, basic, gauss1.0, gauss_10.0, gauss_100.0) with Adam optmizer and it's respective train/test loss plots

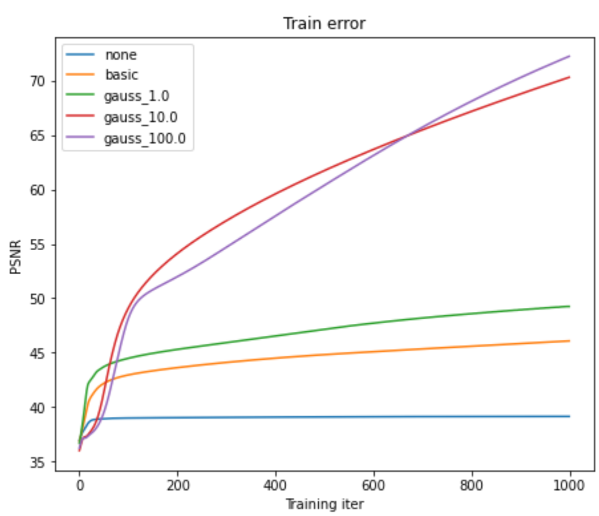


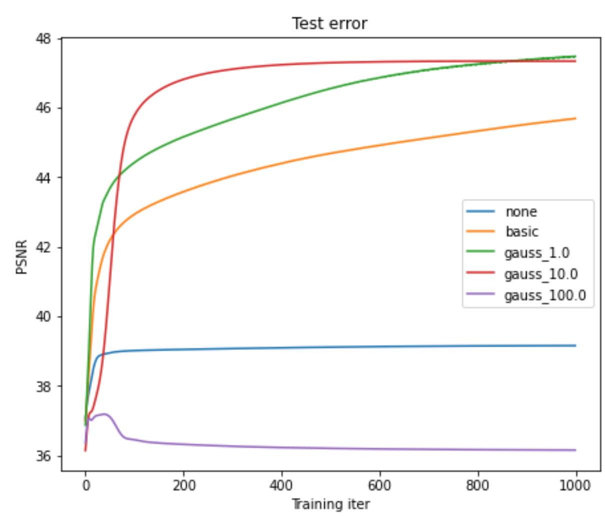


Qualitative outputs

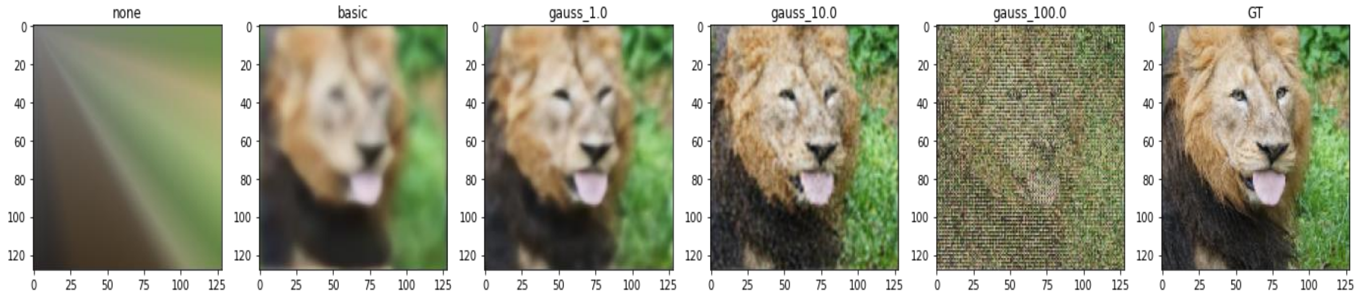


Part 4: High resolution example
The below plots represent the High Resolution – All Mappings (none, basic, gauss_1.0, gauss_10.0, gauss_100.0) with Adam optmizer and it's respective train/test loss plots





Qualitative outputs



Part 5: High resolution (image of your choice)

(For this part, you can select an image of your choosing and show the performance of your model with the best hyperparameter settings and mapping functions from Part 3. You do not need to show results for all of the mapping functions.)

<u>Best Hyperparameters:</u>

Mapping = gauss_10.0

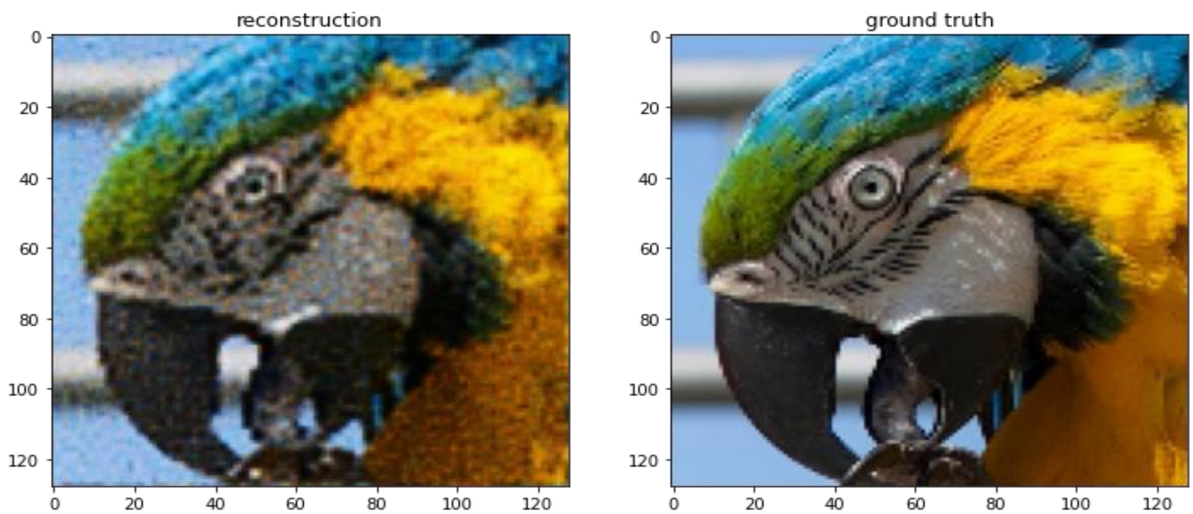
Optimizer = Adam

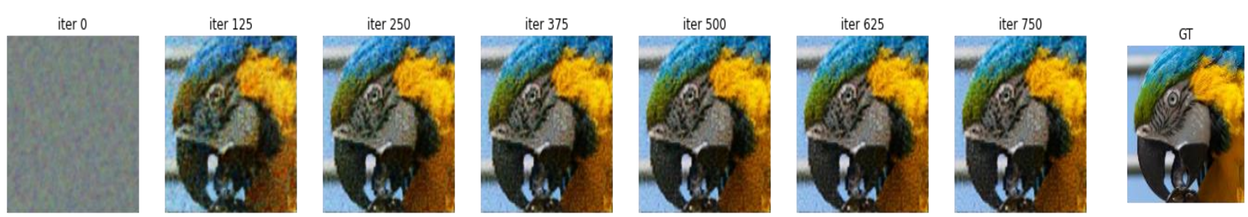
Learning rate = $1e^{-4}$

Num_layers = 5

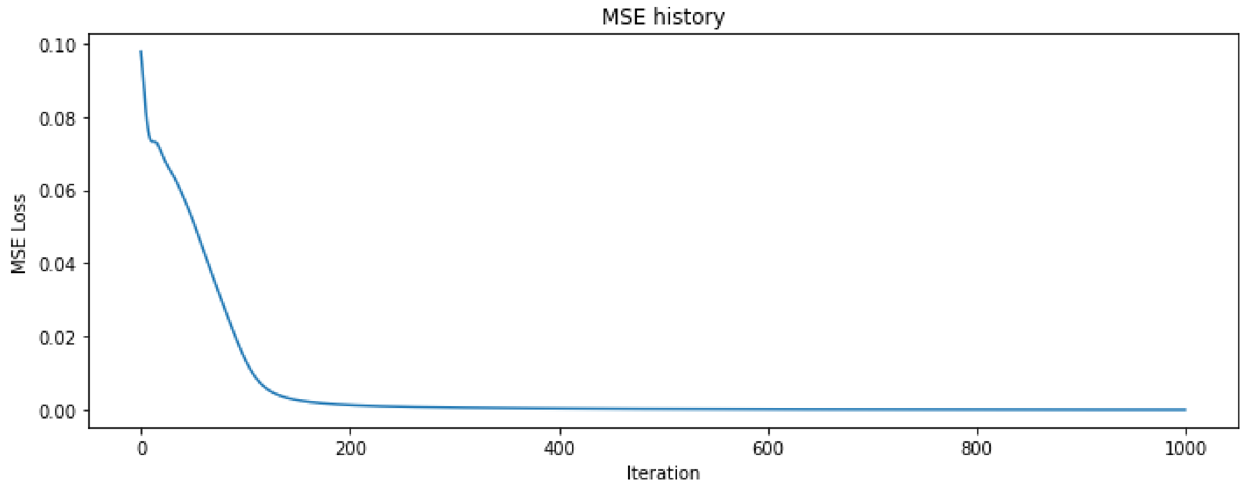
Epochs = 1000

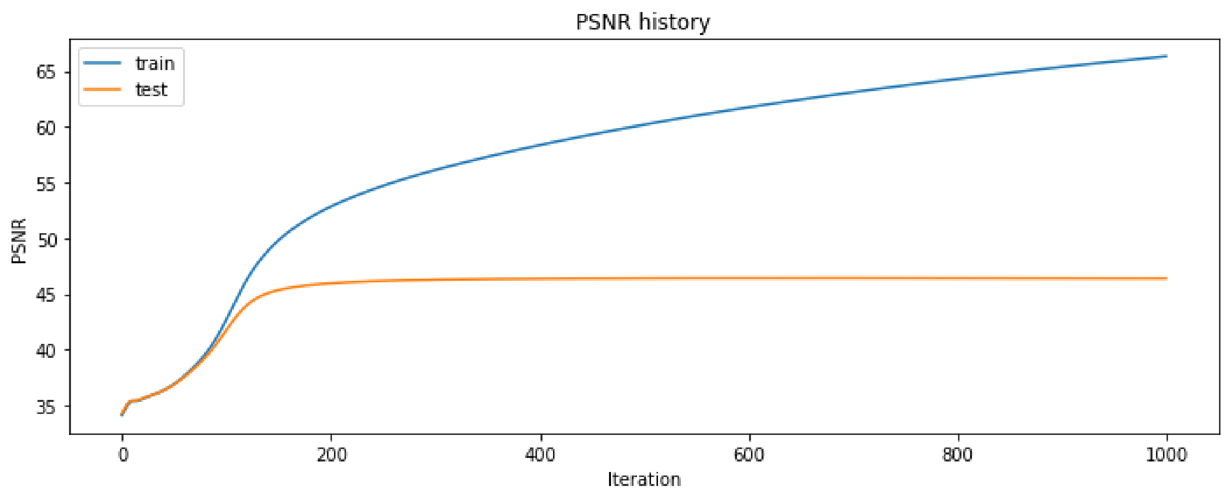
Qualitative Output





MSE Train/test Loss plot





Part 6: Discussion

Briefly describe the hyperparameter settings you tried and any interesting implementation choices you made.

Hyperparameters Used:

Mapping = None, basic, gauss_1.0, gauss_10.0, gauss_100.0

Optimizer = Adam

Learning rate = 1e^{-4}

Num_layers = 5

Epochs = 1000

Hidden_size = 256

We got the best results with gauss_10.0, it produced an almost similar image to that of Ground Truth. We've tried increasing the num_layers and changing the learning rate multiple time, the results of them are discussed in Extra Credit Section. We particularly used Adam optimizer because it was computationally efficient as it converged in less iterations than SGD optimization. This is because Adam optimizer uses Momentum technique to help accelerate the convergence process by accumulating the past gradients to determine the direction. We can be observe that from MSE Loss plot in part 1 and part 2.

How did the performance of SGD and Adam compare?

The Adam was computationally Efficient compared SGD, hence Adam performed well. Firstly, it required fewer iterations to converge. Adam optimizer also adjusts the learning rate adaptively for each of the parameters, this is one of the reasons why it converges faster without overshooting. Further, it is less sensitive to the noise in gradients, as it averages out the gradients, which will help to smooth out the noise. The psnr plot also conveys that Adam optimizer reaches psnr of 39 in less than 100 epochs compared to SGD which takes 1000 epochs but still gets to 36.8 psnr only. So we used Adam for both Low resolution and High Resolution (All mappings) images while training the model.

How did the different choices for coordinate mappings functions compare?

With no mapping (None), we barely got the image with SGD and Adam optimizers. When we performed basic mapping, Adam optimizer for High resolution image started to give us outline. When these coordinates (High Resolution Image) were mapped with gaussian values, the image for **gauss_10.0** gave us the accurate image to that of ground truth. However, for low resolution images, they have less variance in intensity, which would make a Gaussian map less useful. Hence, the results are poor for Low-resolution images. Also, higher gaussian mappings (gauss_100) the variance will be high which resulted in improper image compared to gauss_10 mapping.

What insights did you gain from your own image example (Part 5)?

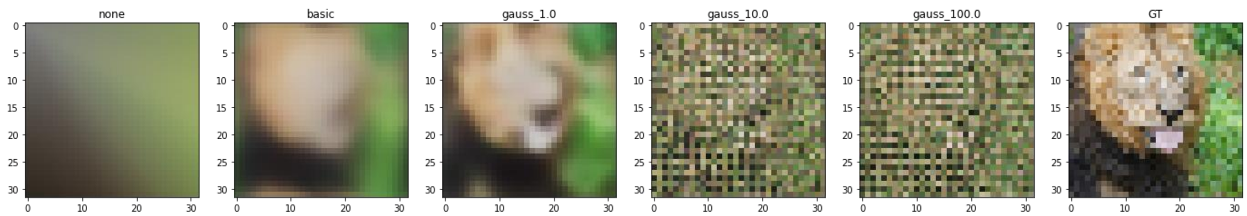
After tuning the model with the hyperparameters (mentioned in part 5) on the image, we were able to obtain the better result. However, we gave the image size as 128(Same as that of our High Res, given in part 4), if we could've increased the image to size = 256, the output would've been better. Also the loss was saturated after 200 epochs (Check MSE Loss train/test plot in part 5) and there was no significant difference in the image produced. So we could've got the same result with fewer epochs. And we think that since image size was same to that of image size given for High resolution, the hyper parameters used for this own image was also same and we got best results possible.

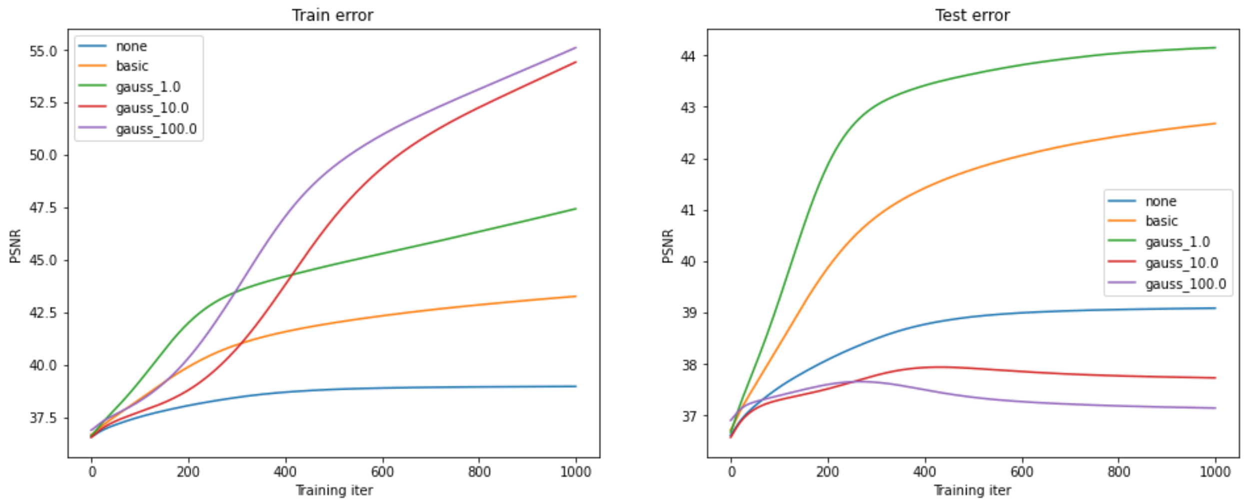
Extra credit: Comparison and discussion of Model performance with these new Hyperparameters

Learning rate: $1e^{-5}$
Num_layers: 7

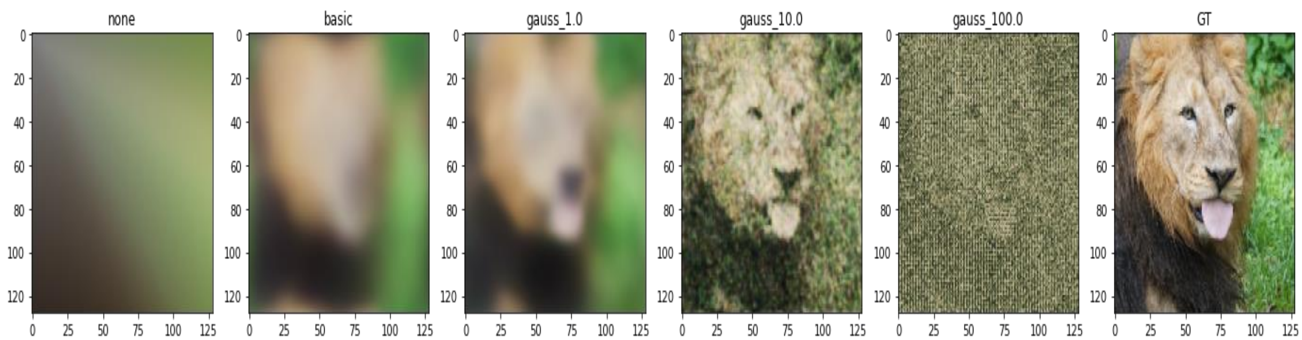
As we can observe that psnr range for Low – resolution image is 55 (max) here. But with normal hyperparameters given in **part 3,** the psnr value is 120 approx., the higher the psnr value, the better would be the image generation. One of the main cause here is as Layers increases, it will tend to overfit. Also the from Test Error plot we can observe that graph saturates under 100 epochs in Part 3, but here we can observe it takes longer to reach the same level of psnr. Hence, it is computationally expensive, but result is similar.

Low resolution – All Mappings (Adam Optimizer) with new hyperparameters mentioned above in Extra credit section





High resolution – All Mappings (Adam Optimizer) with new hyperparameters mentioned above in Extra credit section



We can see that at Gauss_10, the image is not as that of Ground truth with these hyperparameter settings. This because as the layer size increases the model tend to overfit, which is the result we got for gauss_10.0 image. The plot below, also highlights that the model was unable to attain a higher psnr value in lesser epochs (part 4 plot). It is taking longer to reach the saturation value than with the parameter used in part 4. The computational time for running with these parameters is significantly higher (2 hr 30 mins) than compared to our part 4 parameters, which gave better results in half the time (1 hr 10 mins). Hence these parameter values are not efficient(computationally) and also tend to overfit.