ELEC5306 Project2 Section 1.4 Report

Group 4

Section 1.4 Tune hyper-parameters

In this section, we firstly set up three networks with different sizes of convolution and deconvolution layers. Then, in each type of network, choose different learning rates (1e-3, 1e-4, 1e-5) to train the model. The maximum epoch is 30 so that we could also figure out the result on the validation dataset during this training period.

1. The tables below record the result for each curve during training.

<p.s.> The best results for using each learning rate are highlighted in yellow.

1) Loss curve:

Learning rate = 1e-3

Epoch / (N, M)	10	20	30
(32, 48)	3.64	2.20	1.64
(64, 96)	5.34	2.23	1.62
(128, 192)	10.47	25.92	4.86

Learning rate = 1e-4

Epoch / (N, M)	10	20	30
(32, 48)	9.98	5.36	3.73
(64, 96)	8.89	3.77	2.49
(128, 192)	6.87	3.41	2.24

Learning rate = 1e-5

Epoch / (N, M)	10	20	30
(32, 48)	88.97	35.45	19.11
(64, 96)	42.87	20.94	13.89
(128, 192)	40.31	19.54	12.79

2) MSE curve:

Learning rate = 1e-3

Epoch / (N, M)	10	20	30
(32, 48)	0.0143	0.0086	0.00645
(64, 96)	0.0209	0.0087	0.00635
(128, 192)	0.0411	0.1017	0.0191

Learning rate = 1e-4

Epoch / (N, M)	10	20	30
(32, 48)	0.039	0.021	0.015
(64, 96)	0.035	0.015	0.010
(128, 192)	0.027	0.013	0.008

Learning rate = 1e-5

Epoch / (N, M)	10	20	30
(32, 48)	0.349	0.139	0.075
(64, 96)	0.168	0.082	0.054
(128, 192)	0.158	0.076	0.050

3) PSNR curve:

Learning rate = 1e-3

Epoch / (N, M)	10	20	30
(32, 48)	18.478	20.657	21.953
(64, 96)	16.793	20.625	22.010
(128, 192)	13.89	9.94	17.21

Learning rate = 1e-4

Epoch / (N, M)	10	20	30
(32, 48)	14.113	16.816	18.411
(64, 96)	14.615	18.372	20.169
(128, 192)	15.729	18.779	20.594

Learning rate = 1e-5

Epoch / (N, M)	10	20	30
(32, 48)	4.587	8.593	11.294
(64, 96)	7.804	10.903	12.680
(128, 192)	8.062	11.192	13.029

4) SSIM curve:

Learning rate = 1e-3

Epoch / (N, M)	10	20	30
(32, 48)	0.321	0.506	0.580
(64, 96)	0.299	0.507	0.529
(128, 192)	0.191	0.221	0.358

Learning rate = 1e-4

Epoch / (N, M)	10	20	30
(32, 48)	0.127	0.215	0.287
(64, 96)	0.130	0.294	0.390
(128, 192)	0.162	0.325	0.453

Learning rate = 1e-5

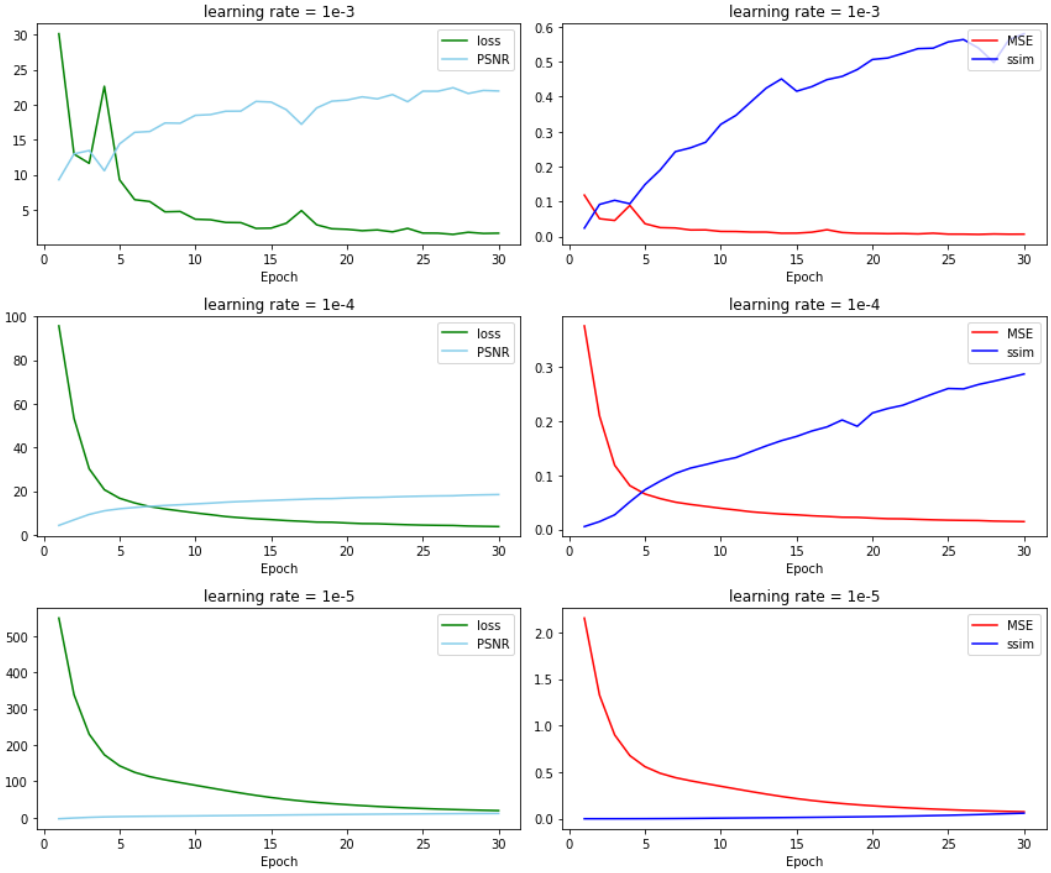
Epoch / (N, M)	10	20	30
(32, 48)	0.007	0.023	0.060
(64, 96)	0.0223	0.0609	0.0885
(128, 192)	0.0337	0.0715	0.1047

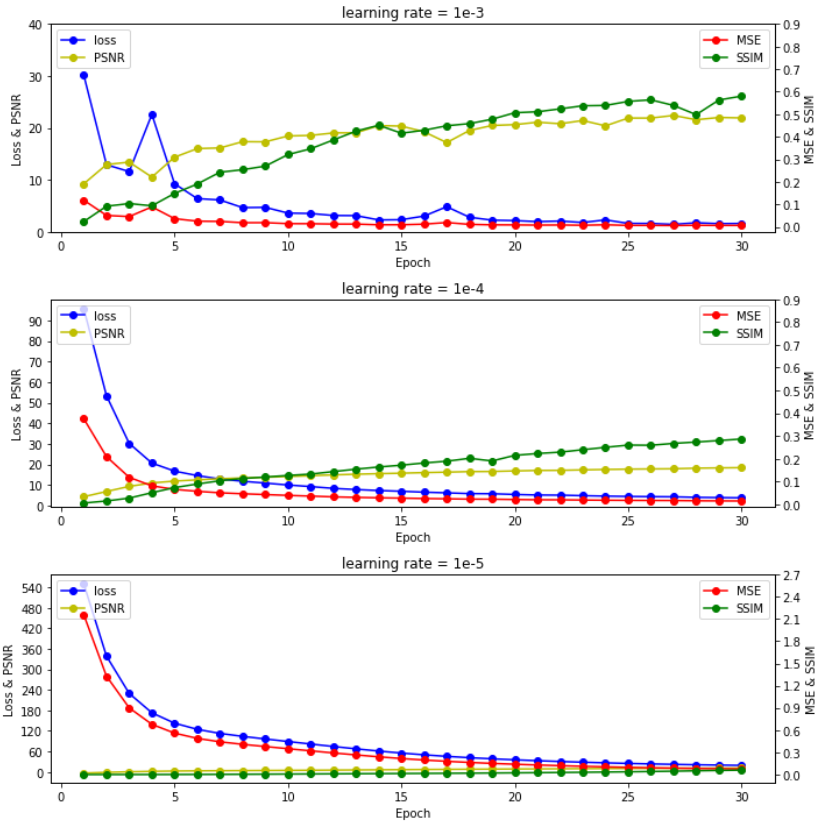
2. The results are also been plotted in different plots

In this section, we use two different ways to plot the results for each model. In the first figure, we combine Loss and PSNR as a sub-plot and we combine MSE and SSIM as the other sub-plot.
In the second figure, we simply combine all the results into it. However, we used two y-axes. The left y-axis represents the Loss and PSNR, the right y-axis represents the MSE and SSIM.

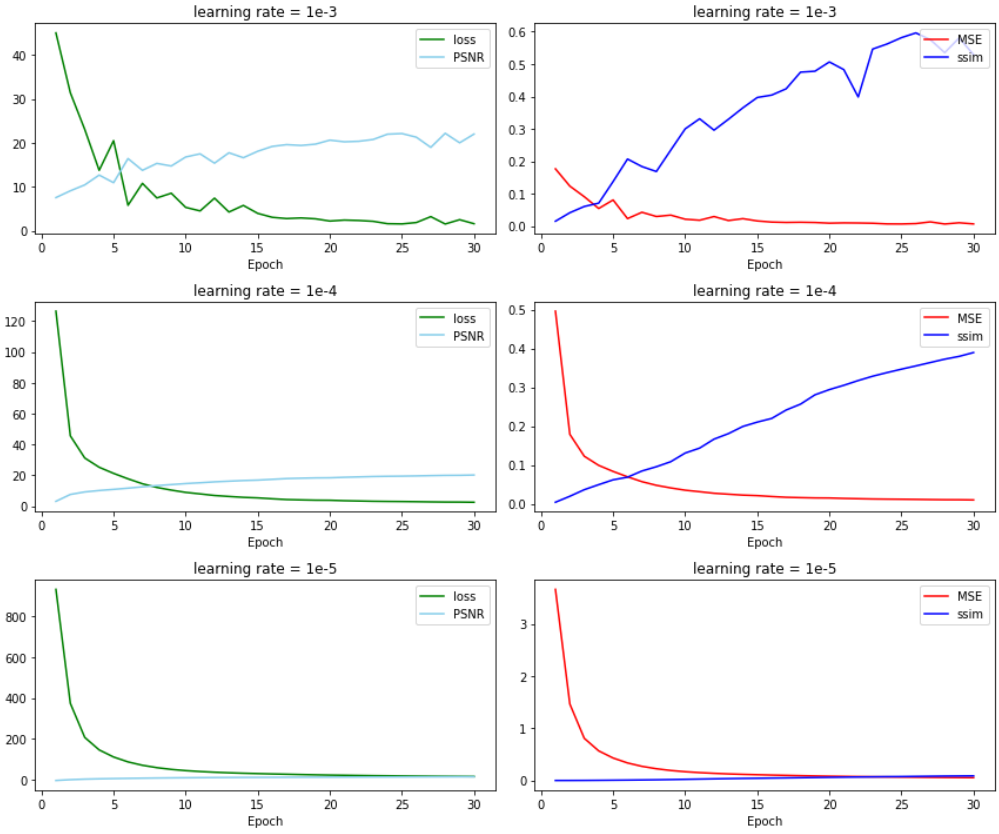
We plot the result in this way because we want to visualize the results as more obvious and easier to understand.

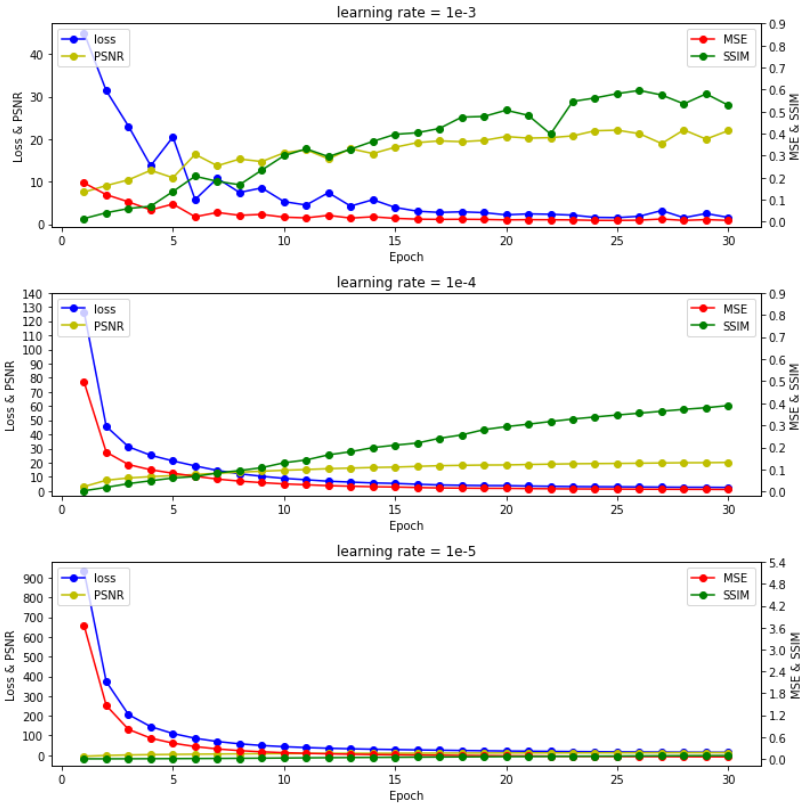
(N, M) = (32, 48):


learning rate = 1e-3
loss
PSNR
Epoch
learning rate = 1e-3
MSE
ssim
Epoch
learning rate = 1e-4
loss
PSNR
Epoch
learning rate = 1e-4
MSE
ssim
Epoch
learning rate = 1e-5
loss
PSNR
Epoch
learning rate = 1e-5
MSE
ssim
Epoch

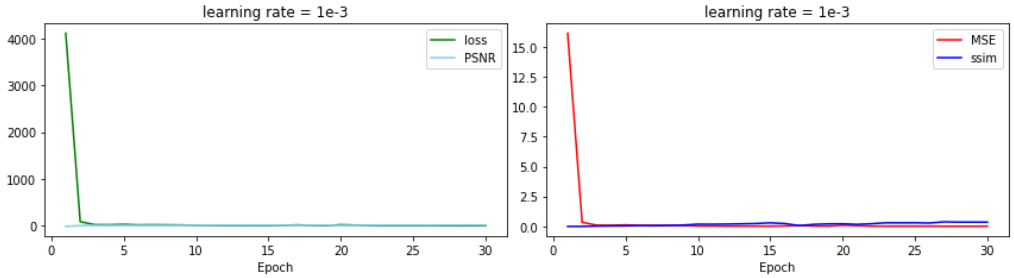

learning rate = 1e-3
loss
PSNR
MSE
SSIM
Loss & PSNR
MSE & SSIM
Epoch
learning rate = 1e-4
loss
PSNR
MSE
SSIM
Loss & PSNR
MSE & SSIM
Epoch
learning rate = 1e-5
loss
PSNR
MSE
SSIM
Loss & PSNR
MSE & SSIM
Epoch

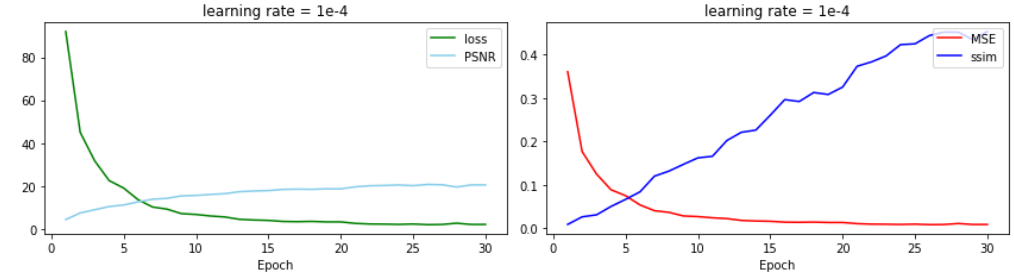
(N, M) = (64, 96):

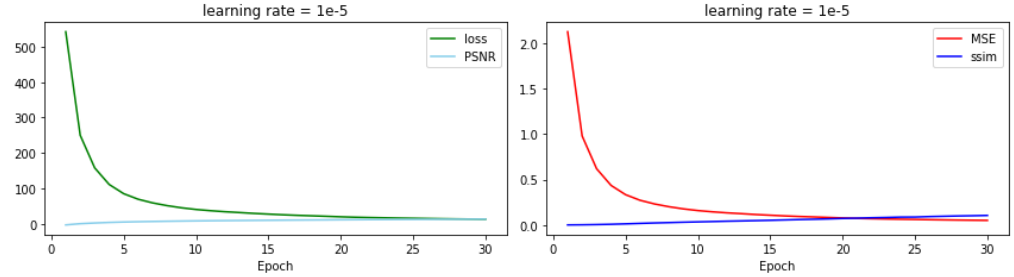

learning rate = 1e-3
loss
PSNR
Epoch
learning rate = 1e-3
MSE
ssim
Epoch
learning rate = 1e-4
loss
PSNR
Epoch
learning rate = 1e-4
MSE
ssim
Epoch
learning rate = 1e-5
loss
PSNR
Epoch
learning rate = 1e-5
MSE
ssim
Epoch

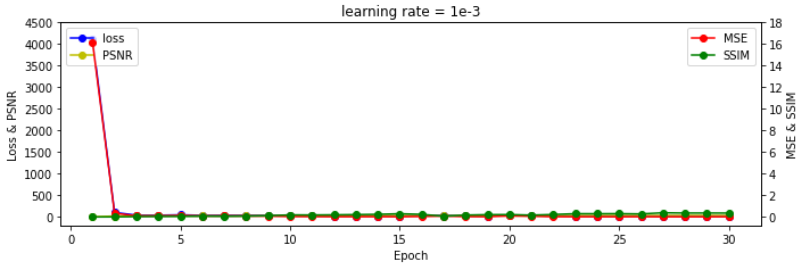

learning rate = 1e-3
loss
PSNR
MSE
SSIM
Loss & PSNR
MSE & SSIM
Epoch
learning rate = 1e-4
loss
PSNR
MSE
SSIM
Loss & PSNR
MSE & SSIM
Epoch
learning rate = 1e-5
loss
PSNR
MSE
SSIM
Loss & PSNR
MSE & SSIM
Epoch

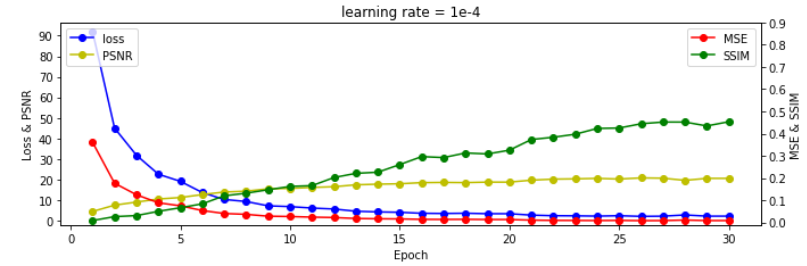
(N, M) = (128, 192):

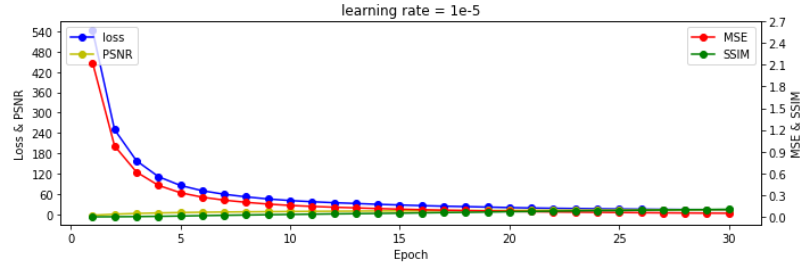

learning rate = 1e-3
loss
PSNR
Epoch
learning rate = 1e-3
MSE
ssim
Epoch


learning rate = 1e-4
loss
PSNR
Epoch
learning rate = 1e-4
MSE
ssim
Epoch


learning rate = 1e-5
loss
PSNR
Epoch
learning rate = 1e-5
MSE
ssim
Epoch


learning rate = 1e-3
loss
PSNR
MSE
SSIM
Loss & PSNR
MSE & SSIM
Epoch


learning rate = 1e-4
loss
PSNR
MSE
SSIM
Loss & PSNR
MSE & SSIM
Epoch


learning rate = 1e-5
loss
PSNR
MSE
SSIM
Loss & PSNR
MSE & SSIM
Epoch

3. Results Analysis

Aim:

1) Minimize the Loss and the MSE (Mean Square Error)
2) Maximize the PSNR and SSIM.

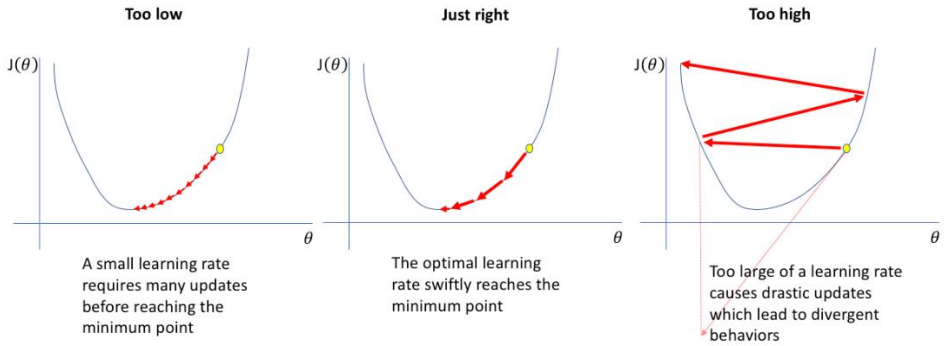
Based on the results above, we could find that:

1) When we increase the number of epochs, the performance will be better (the loss and MSE will be smaller, PSNR and SSIM will be larger).

It is because the number of epochs corresponds to the number of times that we train the network with a whole dataset. The more times we train the network, the more proper weights will be assigned to the network after forward and backward processes.

2) In the same network (same kernel size), the higher the learning rate, the better the performance (the loss and MSE will be smaller, PSNR and SSIM will be larger) will be for each epoch.

The reason is also straightforward. With a small learning rate, it means we will spend more time on finding the global minima of the loss function (figure below), and it could also be easily stuck at the local minima, so the loss will be decreased slower than the others. However, if the learning rate is too large, the network will be hard to converge



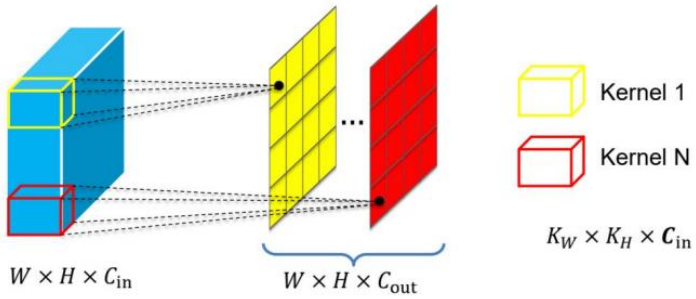
3) Similarly with 2), we could figure out that the initial loss or MSE loss will be higher if we use a smaller learning rate.

It is because, during the first forward process, the model has not “learned too much”, which means it has not updated the weights very well after backward propagation and the optimizer also has not moved too much to the minima in the loss function.

4) When the learning rate is small, the network which has larger convolution and deconvolution channel numbers (kernel numbers) usually will have a better result. For example, when the learning rate is 1e-4 or 1e-5, the network which (N, M) = (128, 192) has better performance (the loss and MSE will be smaller, PSNR and SSIM will be larger).

The channel number represents how many kernels we used in each layer. And we have a fixed kernel size which is 5*5 in our network, which means we have the same receptive field for each convolutional layer. Besides, as our input images have 3 channels, so the first convolution layer must have an input channel number of 3.

The image below claims how the feature maps are generated using N kernels.



In this case, if more kernels are used, more respective fields will be generated and more computation is needed.

Some classical networks such as VGG choose to use several smaller size convolution kernels instead of using one larger size kernel to have the same receptive field. It is because we could save the number of parameters so that we could further save the computation needed. However, as the kernel size in our model is the same (5*5), we can simply ignore this effect.

In this case, we could claim that, **if the learning rate is small, more kernels will help the network learn more features and have a better performance.**

5) An interesting discover

As for (N, M) = (128, 192) and learning rate = 1e-3, we find that the loss and MSE loss in the first epoch is extremely higher (loss is about `4109.36`) than in the others. And it drops down directly and returns to normal as we assumed after the first epoch. However, the final loss is still larger than the others. We trained this network twice on different computers and the results are exactly the same.

But it could be considered as acceptable because after 30 epochs, the loss reduced to only around 4 step by step, which means this hyper-parameter could train the model successfully although it is not the best solution.