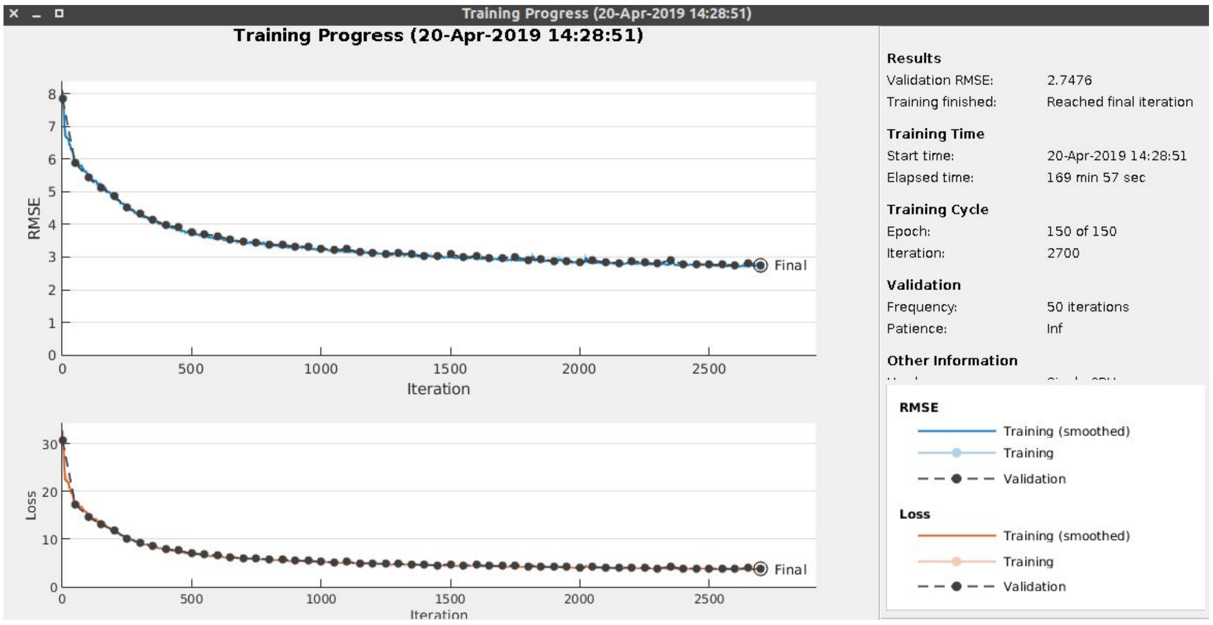
Q1) C)

The architecture of our neural network is inspired from VGG and U Net. We have sticked to the basic encoder decoder structure , but played with the number of convolutional layers before applying the max pool operations to go to a lower resolution. The following paper https://arxiv.org/pdf/1409.1556.pdf proo localization and classification accuracies of deep networks can in general be increased by increasing the depth of CNNs and going to further smaller fields. In our case, we are already on a receptive field of 4*4 and it didn't seem correct to us to go further down to a 2*2 receptive field by applying one more MaxPool2D layer. So we just borrowed number of CNNs before each maxPool from VGG net for the encoder part.

Hence in our architecture we use 2CNNs before first maxPool operation, 2 before 2nd Max Pool and 3 before 3rd Max Pool operation, same as in VGG Net.
In the decoder part, I added CNNs with each upsampling CNN also, to keep the decoder structure similar to the encoder. Didn't make it exact same because the network was getting to big in that case and hence the training time.

Overall Architecture ENCODER
CNN - ReLU - CNN - RELU - MAXPOOL - CNN - ReLU - CNN - RELU - MAXPOOL - CNN - ReLU - CNN - RELU - CNN - RELU - MAXPOOL -

Overall Architecture DECODER
UPSAMPLE_CONV - RELU - CONV - RELU - UPSAMPLE_CONV - RELU - CONV - RELU - UPSAMPLE_CONV - RELU - CONV - RELU - CONV - CLIPPED RELU - REGRESSION



Training Progress (20-Apr-2019 14:28:51)

Results

Validation RMSE:	2.7476
Training finished:	Reached final iteration

Training Time

Start time:	20-Apr-2019 14:28:51
Elapsed time:	169 min 57 sec

Training Cycle

Epoch:	150 of 150
Iteration:	2700

Validation

Frequency:	50 iterations
Patience:	Inf

Other Information

Qn d) 2)

We can observe that the filters in the initial layers of CNNs, specially the conv11 and conv12 are like the thick lines oriented along different directions. Hence the shallow layer filters are capturing the low level features. Whereas as we go deep into the netwrok, filters seems to capture more high level features like textures which are hard to interpreet from the image.

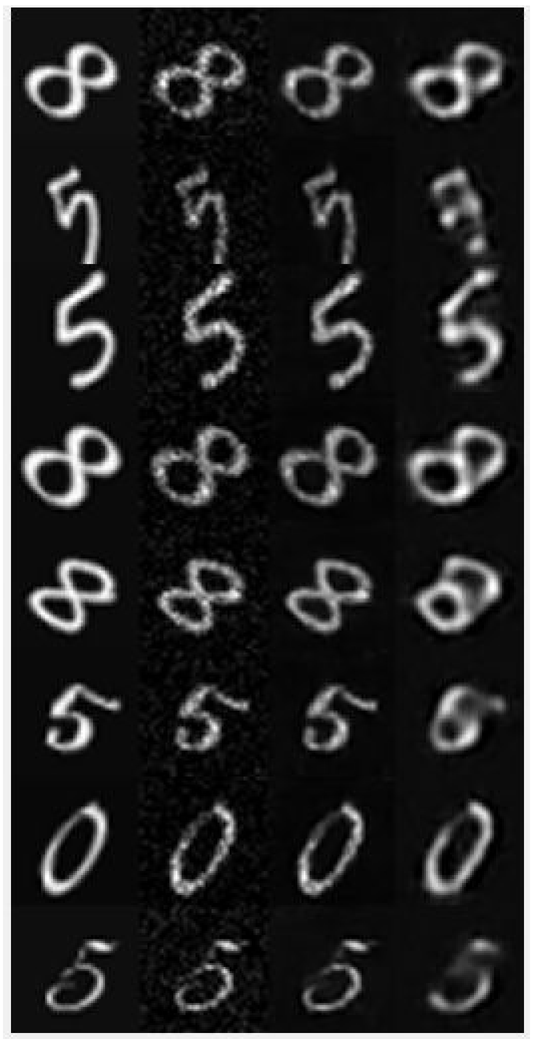
Qnd) 3)

As we go deeper into the network, more and more fine feature representaion of the image is done, ie high level information is encoded into the deeper activations. The shallow layers of encoder and the last layers of decoder have a high resolution representaiton of the input image.

Qne)

Results on 8 images, the order in the montage is

Original Non corrupted image - Corrupted Image - Pretrained Network Result - My Trained Network

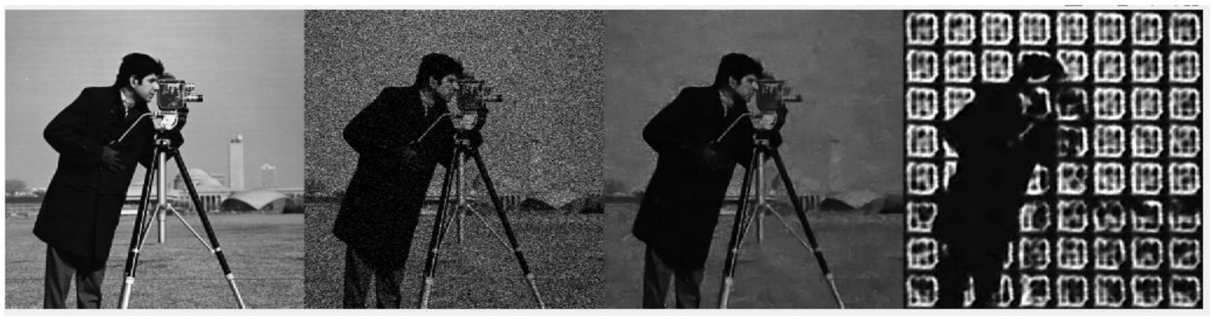


```
pretrained_rmse_list =

  1×8 cell array

    {[0.3693]}    {[0.4255]}    {[0.2960]}    {[0.4236]}    {[0.2966]}    {[0.3012]}    {[0.2617]}
{[0.2359]}


my_rmse_list =

  1×8 cell array
```

```
    {[0.2027]}    {[0.3653]}    {[0.3181]}    {[0.2077]}    {[0.2854]}    {[0.2561]}    {[0.2100]}
{[0.2876]}


pretrained_ssim_list =

  1×8 cell array

    {[0.9973]}    {[0.9983]}    {[0.9988]}    {[0.9962]}    {[0.9984]}    {[0.9991]}    {[0.9994]}
{[0.9998]}


my_ssim_list =

  1×8 cell array

    {[0.9993]}    {[0.9992]}    {[0.9992]}    {[0.9991]}    {[0.9992]}    {[0.9996]}    {[0.9997]}
{[0.9997]}
```

Quite different to the expected, the RMSE of pretrained network is more although the visual reconstruction quality of pretrained network is better

CAMERAMAN
Original Non corrupted image - Corrupted Image - Pretrained Network Result - My Trained Network



I dont know the reason for those smal small artifacts of 32 *32 patches in the background. Definitely the pretrained network works better