First of all, we try to construct a CNN network by ourselves. There are four steps of the CNN part, which are data processing, the construction of network, training model, and testing model.

Firstly, for data processing, we need to change the dimension of input data. The initial input and target data shape is (1, 640, 372, 2). So in data processing, firstly we compute absolute value to get a real image and the shape is (1, 640, 372) now. Secondly use unsqueeze function to raise dimension to (1, 1, 640, 372) because the upsample function in model require a 4 dimensions input, Finally use T.center\_crop to crop the images to the central 320x320 pixel region(1, 1, 320, 320).

Then for network model, it contains two convolution layers, two pooling layers, upsample and a 1\*1 convolution layer and ReLU activation function. The kernel of first and second convolution layer are both 5\*5，the stride and padding are both 1 and 2 separately, and the out\_channels is 16 and 32 respectively. And the two pooling layers are both 2\*2 of maxpooling. For upsample, the scale is 4 and mode is bilinear and each layer contains a ReLU. First of all, the input data size is (1, 1, 320, 320),after the first convolution, I extracted 16 features, the size change to (1, 16, 320, 320), because (320+2\*2-5)/1+1=320. Then passing the maxpooling, the size reduce to (1, 16, 160, 160), the same with last layer, the size continue to change to (1, 32, 80, 80) after the second convolution and maxpooling. Then we need to use upsample to inprove size to (1, 32, 320, 320) and at last change to (1, 1, 320, 320) by 1\*1 convolution layer.

Turn to training model, I used the dataloader function from pytorch. Because the size of input and target data is same, so the loss function we can choose could be l1 or l2(Mse), firstly i choose l1 and sdg for optimizer. I set learning rate 0.001, after data processing like before and begin to train. because the limit of time and the gpu of my laptop, I just trained 2 epochs and save the model. However, I observe the loss has a remarkable fluctuation and keep unchanged in the end.

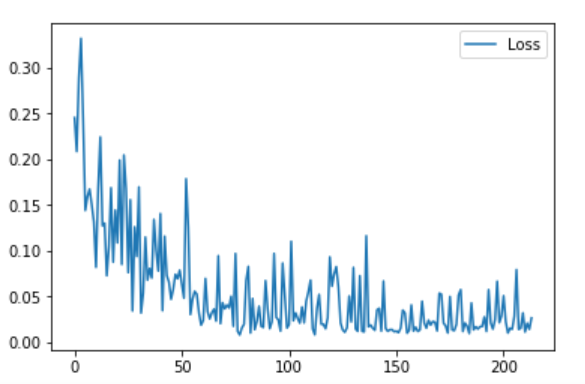
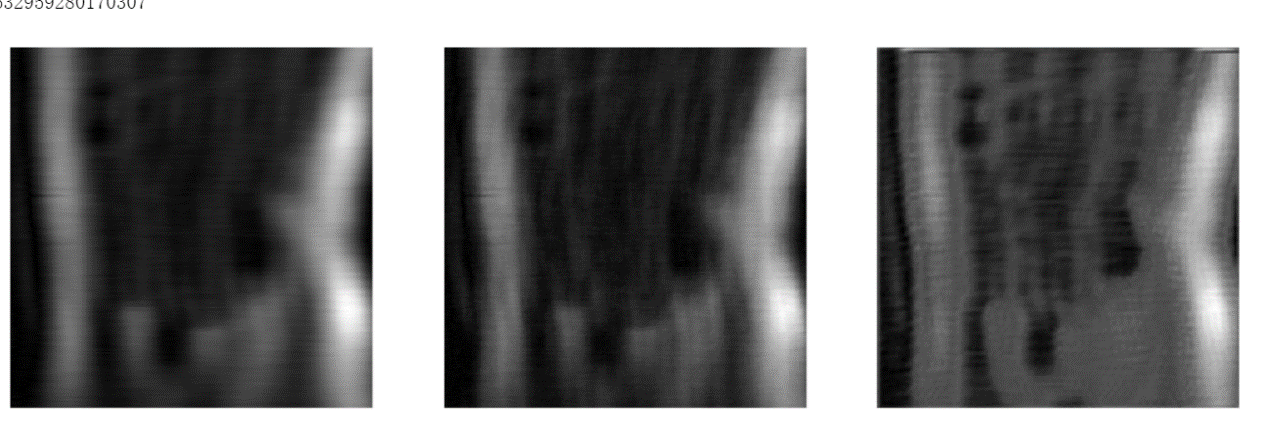


Fig.1. The reduction of loss curve

By testing model, the performance of my model is not good. I input the testing data, set AF = 8 and use ssim function test, the average ssim is just 0.45 and the image is below. So i try to modify the network and adjust hyperparameter to optimize the model. Firstly, i change the loss function to mse and reduce learning rate to 0.0001. Besides them, I also add a convolution layer to extract 64 features, and add 2 1\*1 convolution to reduce size progressively. After training 2 epochs and testing, I find the performance is still not good, but I do not have enough to continue to adjust model, and we decide to use unet model.

Fig.2. the left is the image with undersampling rate 8, the center is the target image and the right is the image after inputting in model.

For dataloader, firstly use load\_data\_path function to load all file names, paths and slices.

Next getting the dataset by MRIDataset function, finally use DataLoader function to get a

data iterator which can iterate each set of data. I would mainly introduce the get\_epoch\_batch function in MRIDataset. It could random select a few slices from each volume. Firstly it loads the data from file and transform to tensor, the apply random mask by MaskFunc function. At last after undersampling and normalizing data, we got the input data.