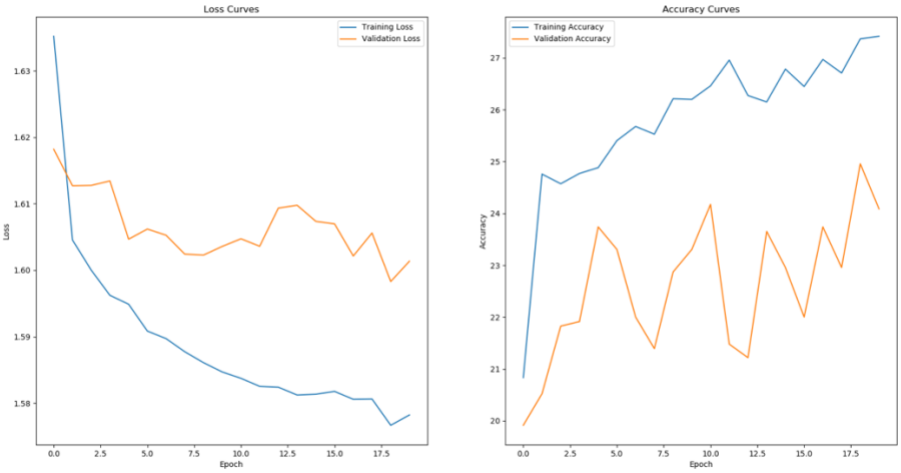
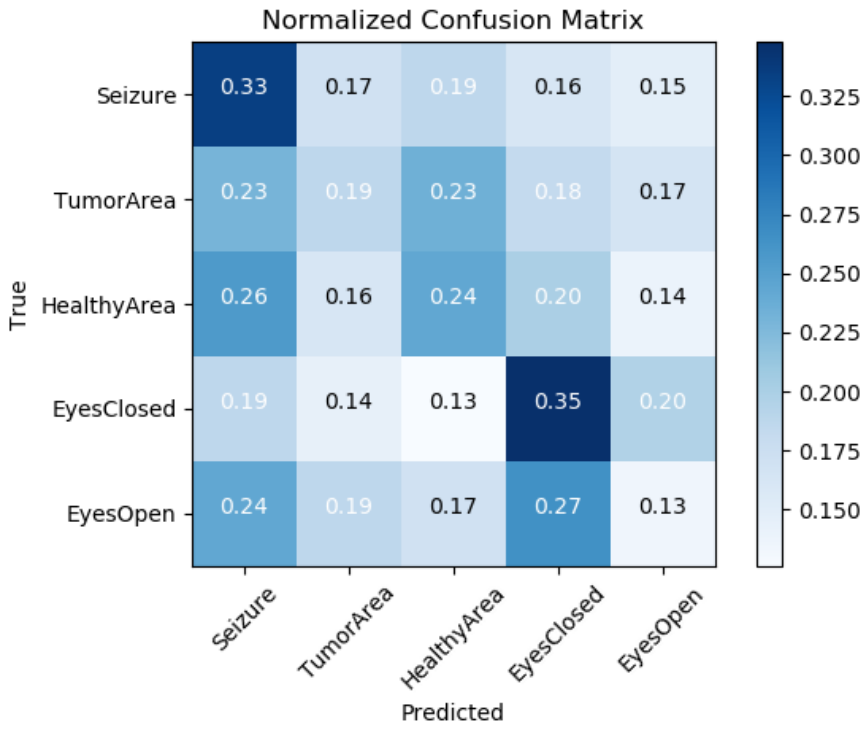
1.2b Assumption based on current model: Trainable parameters: (178*16+16*5) 2949 in total because there are (16+5) 21 neurons total without inputs layer in MLP model.

Then total computation will be sum of each hidden layer: 178*16*3 + 16*5*3 + 16*3 = 8832

1.2c
Number of epochs = 20

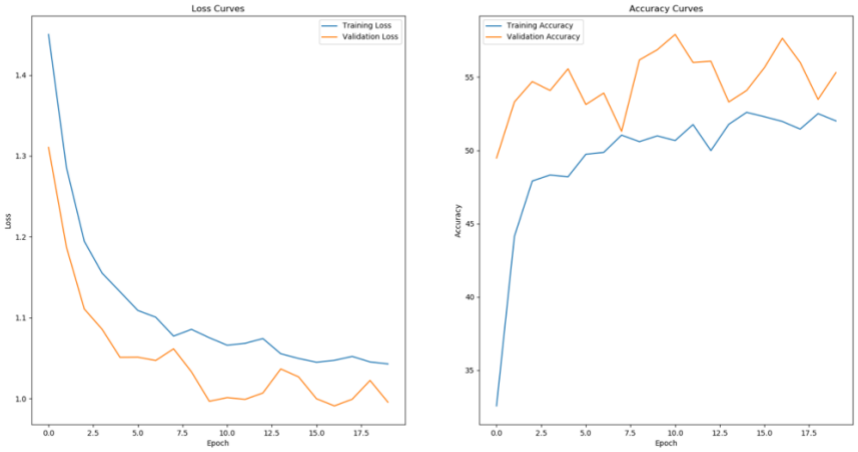


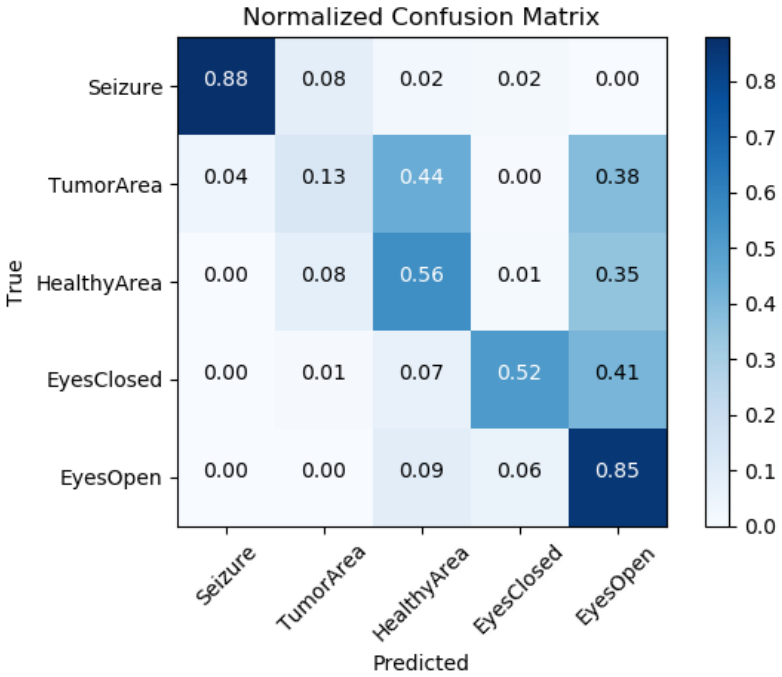

1.2d
Number of epochs = 20



1.2e

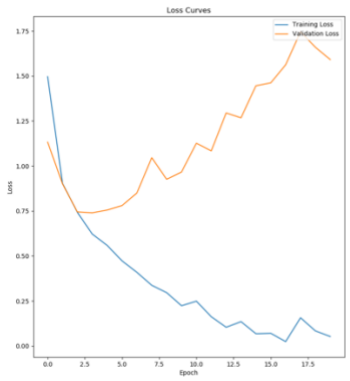
First idea is that I have tried to change hidden units from 16-30-50. Hidden units could help carry info. So generally, more hidden units were added, more accuracy model became. For more hidden units, it will have overfitting problem. In order to avoid overfitting problem, I added dropout =0.5. Then added ReLu function with first batch 178. The good thing for ReLu function is that it does not activate all neurons at once, which will make the whole network sparse and speed up the running(the fastest model in three). The below plots, the improved model performance improved much because of the reasons I mentioned above.



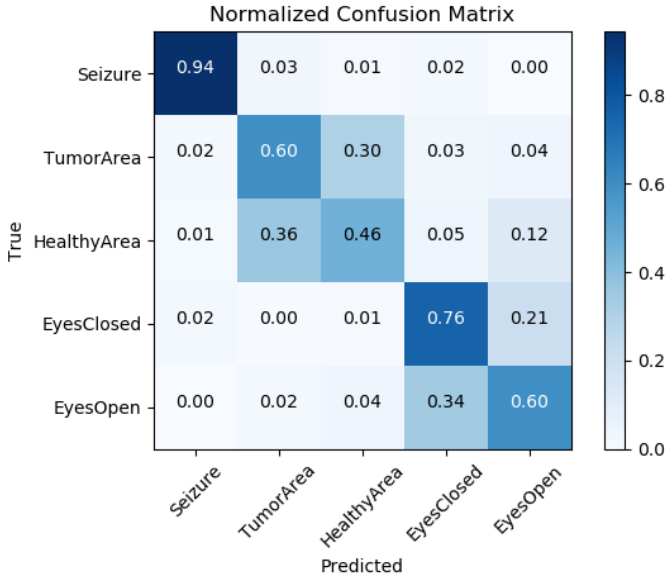



1.3b
First convolutional layer = (5*1+1)*6=36, which kernel size = 5, strade = 1, filter =6
Second convolutional layer = (5*6+1) * 16 = 496
Then first fully connected layer = 128*45*16+128 = 92288, which hidden units = 128, input = 720
Then first fully connected layer = 5*128+5 = 645
So, total parameters of trainable for CNN model is 36+496+92288+645 = 93465

1.3c
Number of epochs = 20

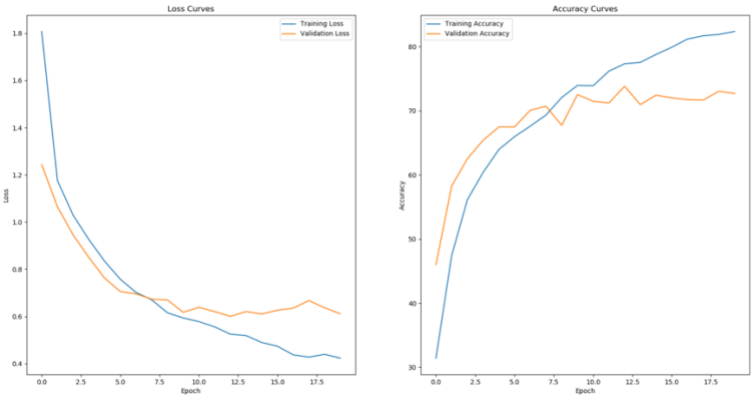


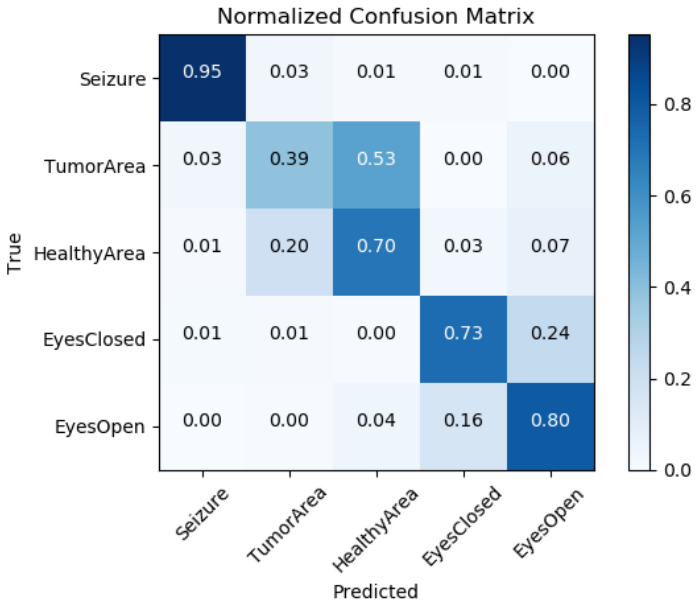





1.3d
For this model, I saw the previous one was doing pretty good. So, here I only added dropout for this model and did not change others. One reason is that a little bit overfitting may impact this model performance. And result is under what I expected. Model parameters was good and dropout could only help a little. I have tried 0.1, 0.2, 0.3 0.4 dropout value. I leave 0.2 plots below, which do not make compared to previous model.

Number of epochs = 20



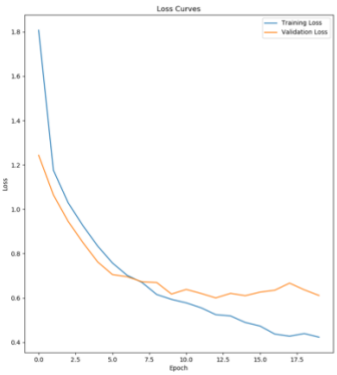



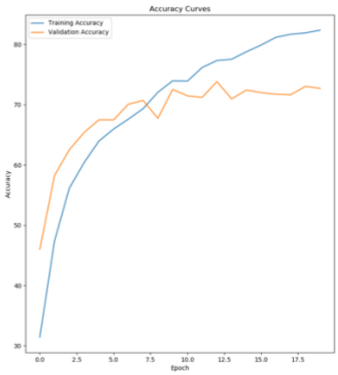
1.4b

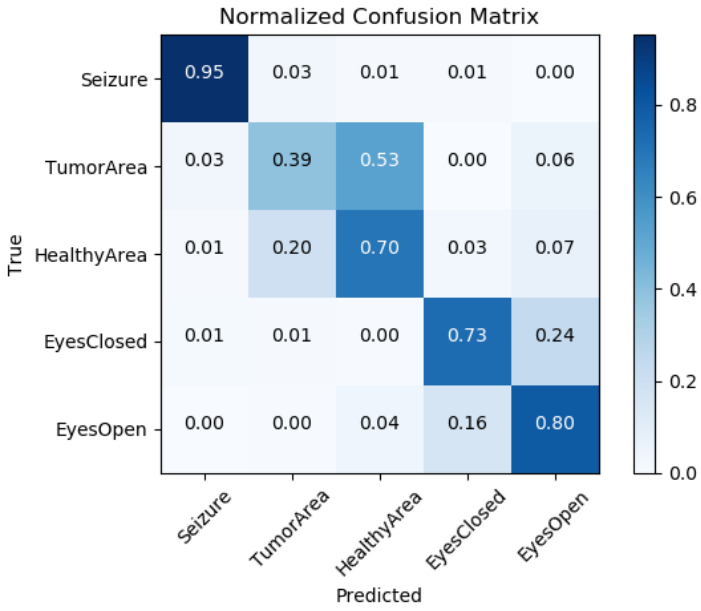
GRU model is much easy to calculate the total trainable parameters, which is 3*16*(16+1+1) = 864

1.4c

Number of epochs = 20



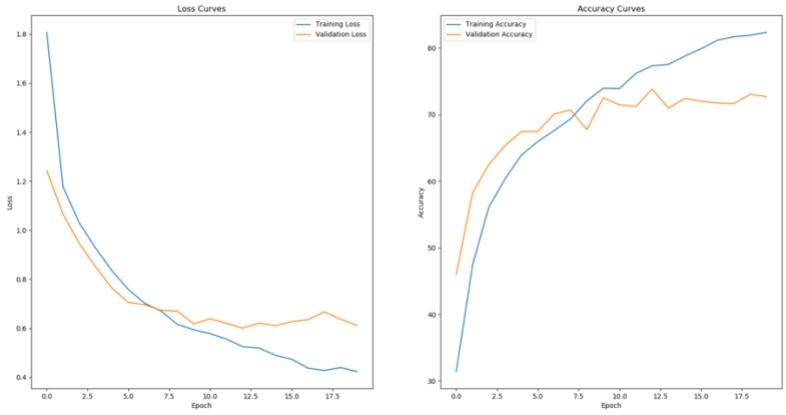



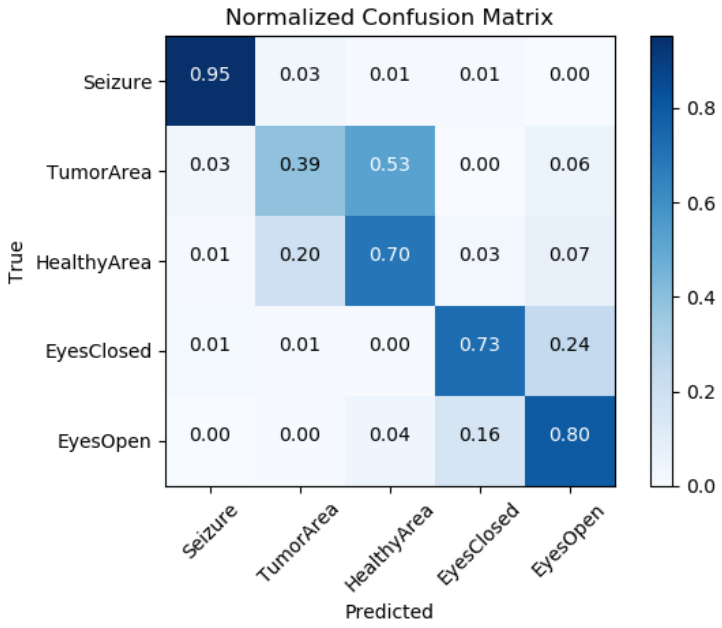




1.4d

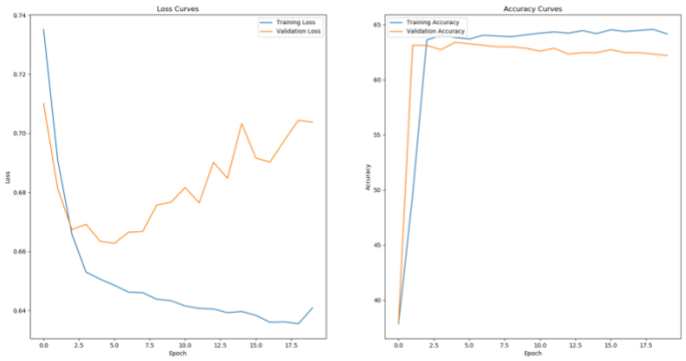
I did not change hidden size and number of layers because RNN model tends to overfit the data and increasing hidden size and number of layers will have a negative impact. So, I added a dropout 0.3 and it works positively in the model. And since RNN was the slowest model, ReLu function become my first and best choice here. Compared to previous model, it did improve a little but overall it is not signification.

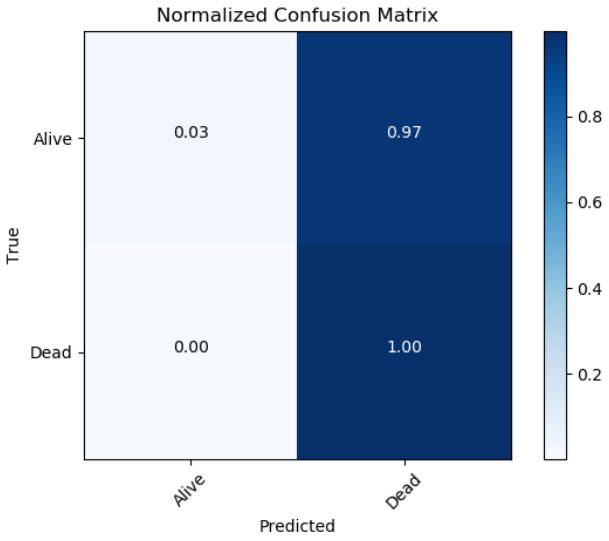
Number of epochs = 20



2.3a
Number of epochs = 20




The learning curve shows that epoch size will be greater to select close to 1 and from loss function and accuracy function plot shows that model was not going well. It could clearly explain in normalized confusion matrix because it fails to predict alive patient 97%. It has 100% accuracy predict dead patient, which could not make sense. It is even worth than random guess.

2.4b
I kept epochs to 20 because it will be better to compare the two different model. And since RNN is quite expensive and tends to overfit. Larger epochs will hurt the model for sure. But I still tried 128 hidden units because I did not want to lose info at the very beginning. And I added three GRU layers, which could make model go deep and learn more. Finally, in order to overcome overfitting problem, I added a 0.5 dropout. Compared to previous mode, it has tremendously improvement as the reason I showed above. The result is showing below.

But for further improvement, I think I would do sth in loss function. I would give a little bit weight to the right prediction of alive patient, which could make alive prediction right a little important than dead patient. I think it will help model in the performance.

Number of epochs = 20