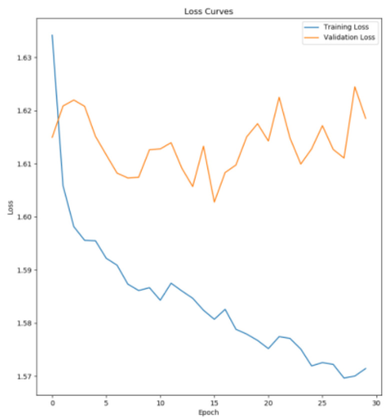
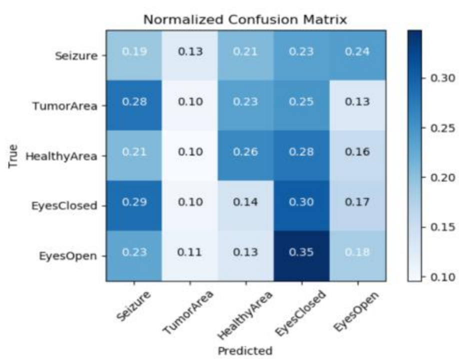
1.2 b)

- There are a total of 2949 trainable parameters, that is MLP has 16+5 = 21 neurons without including the inputs, (178*16) + (16*5) = 2949.
- The total computation is 8832, that is since linear layer applies three operations per data point that should make (178*16*3 = 8544) + (16*5*3 = 240) + sigmoid operation (16*3 = 48). Hence 8544 + 240 + 48 = 8832.

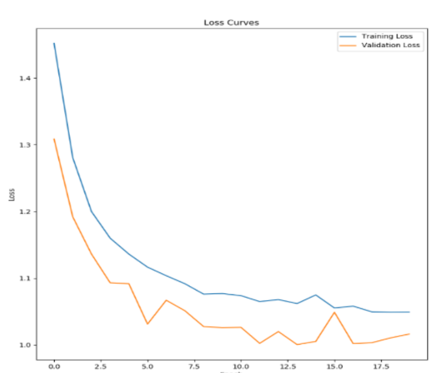
1.2 c) Number of epochs = 30

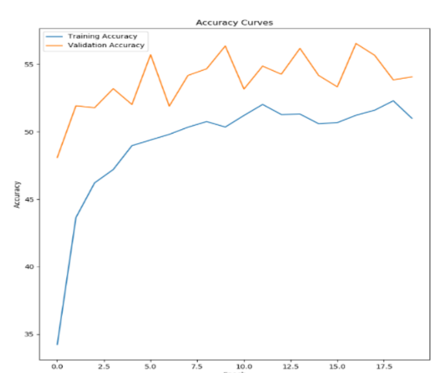


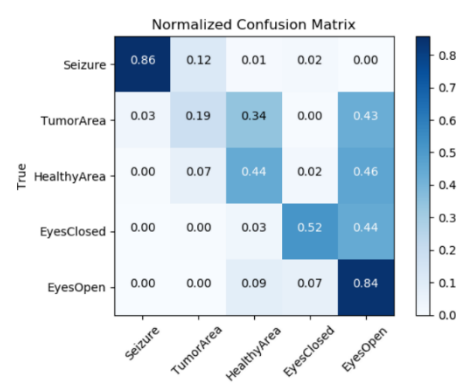
1.2 d) Number of epochs = 30



1.2 e) I changed the hidden units = 50, dropout = 0.5, BatchNorm1d = 178, activation function = relu, epoch = 20. From the below plots, my model's performance improved significantly mainly because of relu activation function and dropout of 0.5. I experimented with different dropout values and found that anything beyond or below .5 did not have an impact on performance (and hence chose .5). Relu function does not activate all the neurons at the same time making the network sparse and hence makes the computation efficient. Increasing the number of hidden units had less impact. This makes sense because increasing model complexity causes overfitting issues which could have an impact on the performance.   Another thing I noticed was MLP is the least expensive model in terms of computation time.



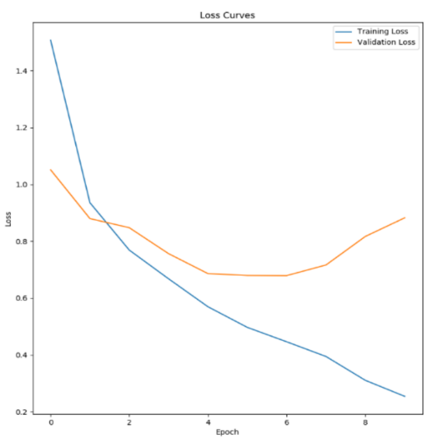


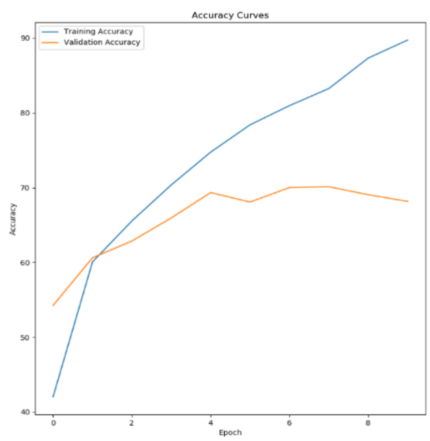


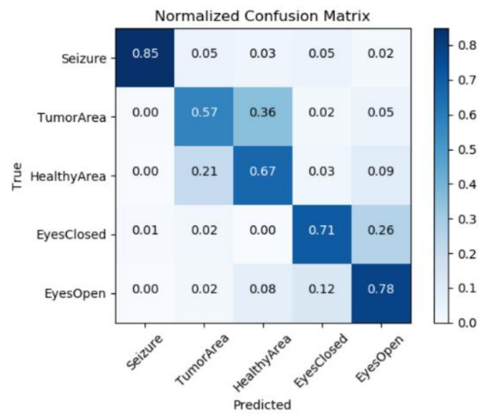
1.3 b)

- The convolutional layer has [(Kernal size * stride) + 1] * filter. So with k = 5, s = 1, and F = 6, [(5 * 1) + 6] = 36.
- For convolutional layer 2 it will be [(5 * 6) + 1] *16 = 496
- The fully connected layer 1 has [(Hidden Units * Input) + Hidden Units], where hidden units = 128 and inputs = 45*16, so [128*(45*16)+128] = 92,288
- For fully connected layer 2, [(5*128)+5] = 645
- The number of parameters of the CNN is 93, 465, that is adding everything 36 +496+92,288+645.

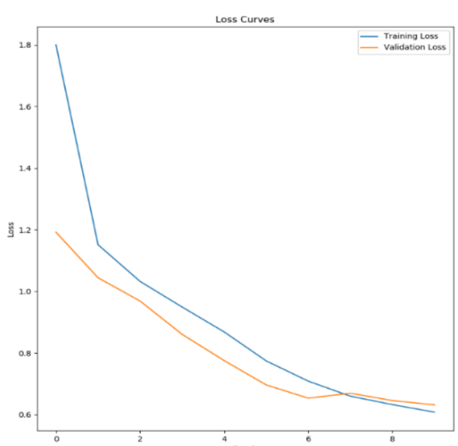
1.3 c) Number of epochs = 10

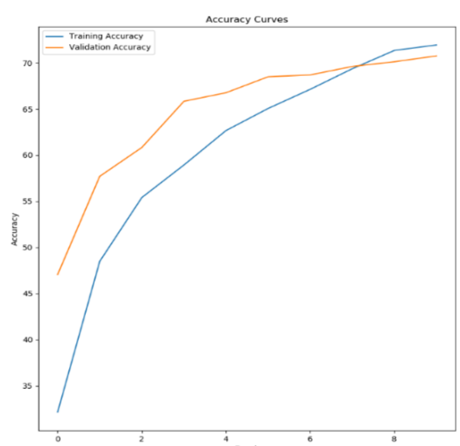


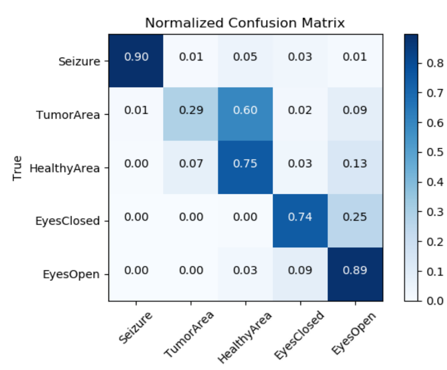




1.3 d) I only changed the dropout = 0.2 and kept everything else the same. I did not change anything else because I wanted to see the impact of only dropout on the performance. I experimented with different dropout values and found that going beyond 0.2 decreased performance significantly. Going below 0.2 did not improve performance much and hence 0.2 seems to be the ideal dropout value for this CNN model. From the plots below, my performance has improved positively just based on my ideal dropout value of 0.2.



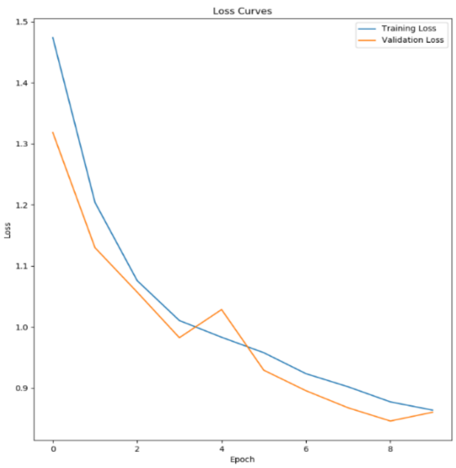


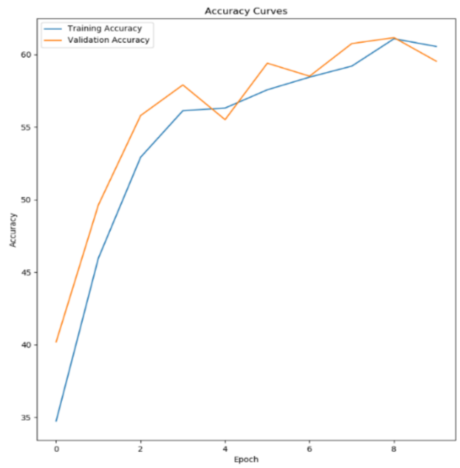


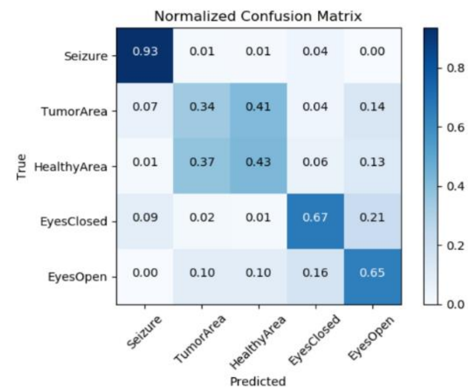
1.4 b)

- RNN has G*[H(H+I)+h] where H is hidden units, G is GRU and I is input. So, 3*[16(16+1)+16] = 864. So the trainable parameters is 864.

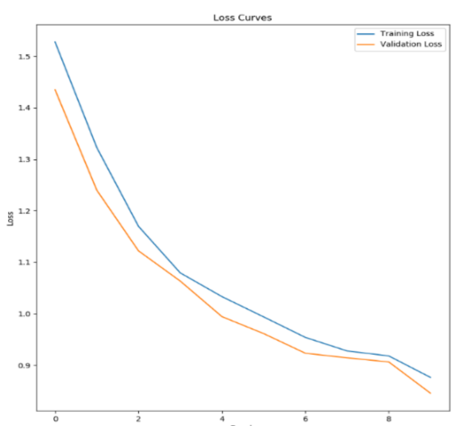
1.4 c) Number of epochs = 10

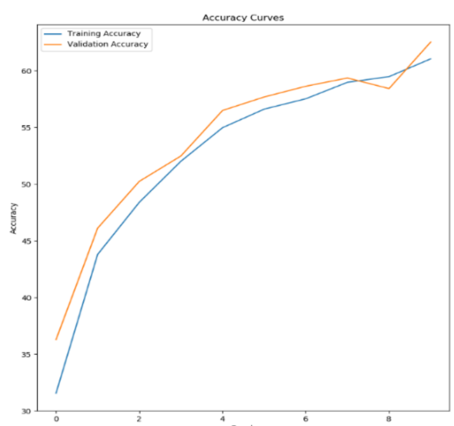


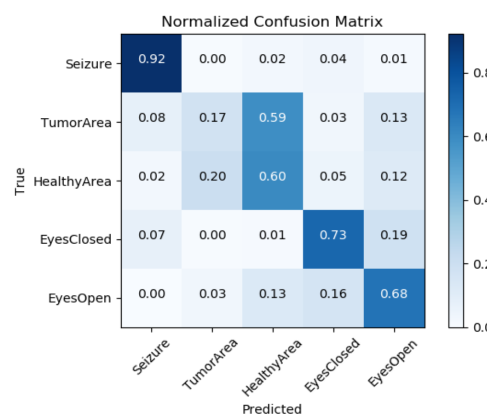




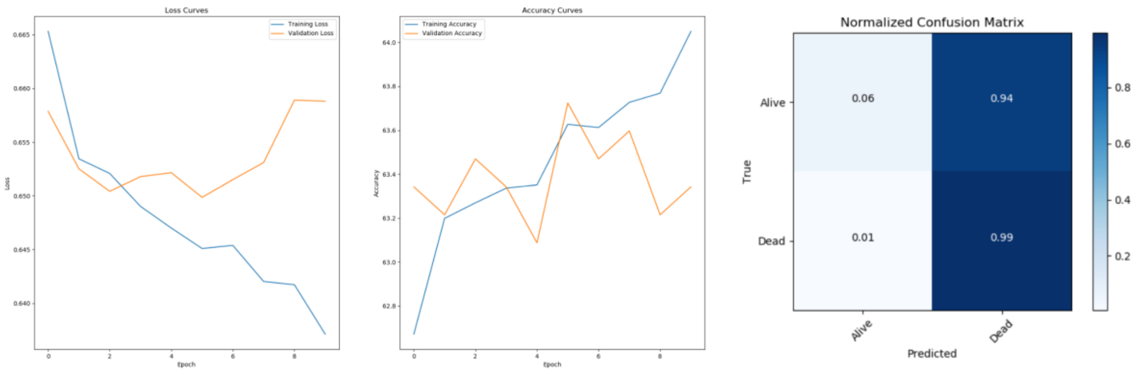
1.4 d) I kept the number of epochs = 10 and did not increase it because RNN takes a very long time to run. I tried increasing the number of hidden size and number of layers and found that it impacted my model's performacne negatively and hence I stuck with hidden_size = 16 and num_layers = 1. I tried increasing the droput value to 0.3 and found that it has a very slight postive imact on the model's performance. I experiemnted with different activation functions. With sigmoid and tanah my performance did not improve significantly but with Relu performance did improve slightly. From the below plots, although the performance improved postively overall, the improvement was not very significantly greater as seen in other models.







2.3 a) Number of epochs = 10



2.3 b) I kept the number of epochs to 10 since RNN is computationally expensive. I increased the hidden size to 128, dropout values to 0.5, and included three GRU layers. Based on the below graphs, this has increased the performance tremendously as compared to the previous model.