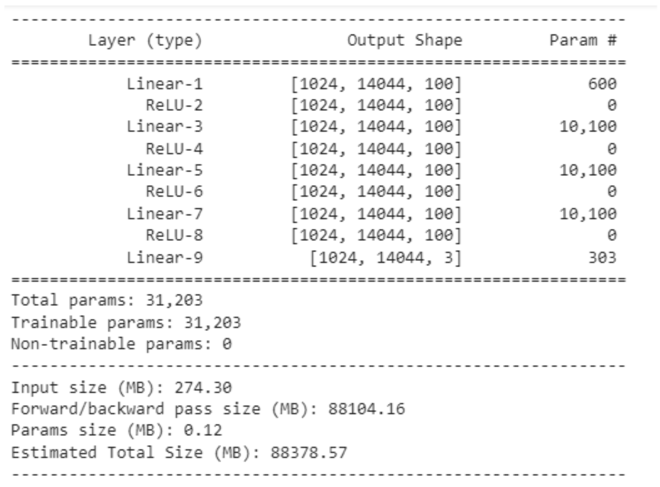
Computational MR imaging **Name:** *Navaneeth N*
Laboratory 9: ML in MRI and neural network architecture design
25/01/2023

1. **1. Diffusion Data Classification:**

   1.1 First we try to design a neural network tasked with classifying labels of brain tissue pixel by pixel based on quantitative diffusion parameters. In the data set we have 3 class labels as Thalamus , Corpus callosum (CC) and Cortical white matter (Cortical WM). The features selected in total are 5, namely, T1 weighted anatomical Image, FA (Fractional Anisotropy) , MD (Mean Diffusivity) , AD (Azial Diffusivity) and RD (Radial Diffusivity). This represents our feature vector 'x'. We choose the training to testing ratio as 0.8.

   1.2 The model of choice is a 3-layer fully connected neural network. Each fully connected layer has 100 elements followed by a ReLU activation function. The last layer is a linear layer that gives us the classification label as an output of the network. Figure 1. summarizes the network structure.



```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Linear-1        [1024, 14044, 100]             600
              ReLU-2        [1024, 14044, 100]               0
            Linear-3        [1024, 14044, 100]          10,100
              ReLU-4        [1024, 14044, 100]               0
            Linear-5        [1024, 14044, 100]          10,100
              ReLU-6        [1024, 14044, 100]               0
            Linear-7        [1024, 14044, 100]          10,100
              ReLU-8        [1024, 14044, 100]               0
            Linear-9          [1024, 14044, 3]             303
================================================================
Total params: 31,203
Trainable params: 31,203
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 274.30
Forward/backward pass size (MB): 88104.16
Params size (MB): 0.12
Estimated Total Size (MB): 88378.57
----------------------------------------------------------------
```

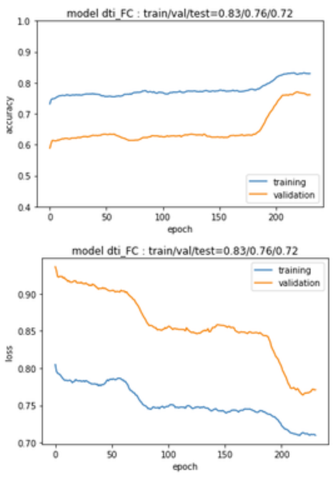
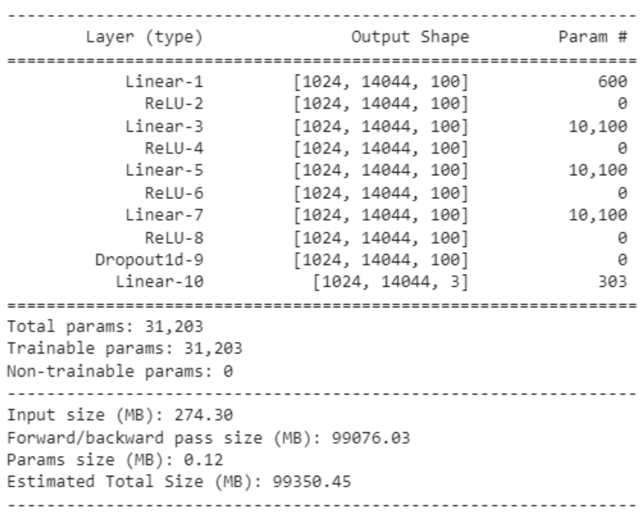


Figure 1: 3 layer FC dti classifier structure.

   1.3 The classifier's output accuracy and loss after 250 epochs is plotted below. We can see that the classifier performance is fairly average on the training, validation, and testing data.

   1.4 An experiment to improve the classification accuracy is attempted by introducing a dropout layer which ignores the contribution of the weights from 25% of elements in the network. This is aimed at making the model better at generalization and to cope with overfitting issues. However, we see that in terms of performance, the dropout does not improve the network.



```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Linear-1        [1024, 14044, 100]             600
              ReLU-2        [1024, 14044, 100]               0
            Linear-3        [1024, 14044, 100]          10,100
              ReLU-4        [1024, 14044, 100]               0
            Linear-5        [1024, 14044, 100]          10,100
              ReLU-6        [1024, 14044, 100]               0
            Linear-7        [1024, 14044, 100]          10,100
              ReLU-8        [1024, 14044, 100]               0
         Dropout1d-9        [1024, 14044, 100]               0
           Linear-10          [1024, 14044, 3]             303
================================================================
Total params: 31,203
Trainable params: 31,203
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 274.30
Forward/backward pass size (MB): 99076.03
Params size (MB): 0.12
Estimated Total Size (MB): 99350.45
----------------------------------------------------------------
```

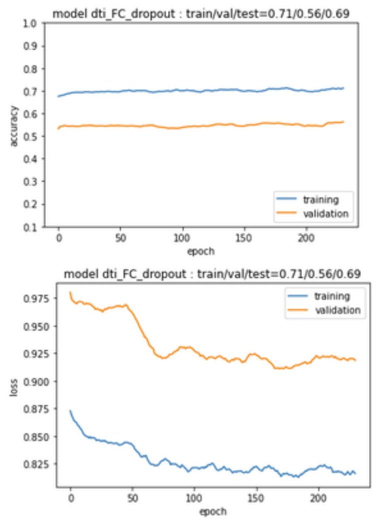
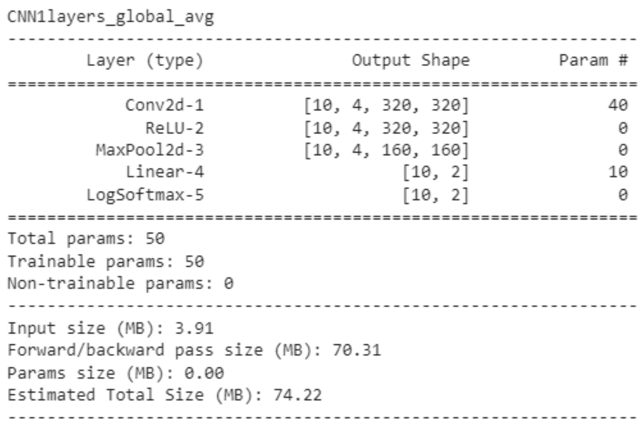


Figure 2: 3 layer FC dti classifier with 0.25 dropout.

2. **Image Quality Classification:**

   2.1 For the second task we attempt to classify images as being either fully-sampled or accelerated and reconstructed with compressed sensing with various CNN architectures. First we attempt a simple 1 layer CNN whose structure and training performance are summarized below. From this we see that the model under fits the data



```
CNN1layers_global_avg
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [10, 4, 320, 320]              40
              ReLU-2         [10, 4, 320, 320]               0
         MaxPool2d-3         [10, 4, 160, 160]               0
            Linear-4                   [10, 2]              10
        LogSoftmax-5                   [10, 2]               0
================================================================
Total params: 50
Trainable params: 50
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 3.91
Forward/backward pass size (MB): 70.31
Params size (MB): 0.00
Estimated Total Size (MB): 74.22
----------------------------------------------------------------
```

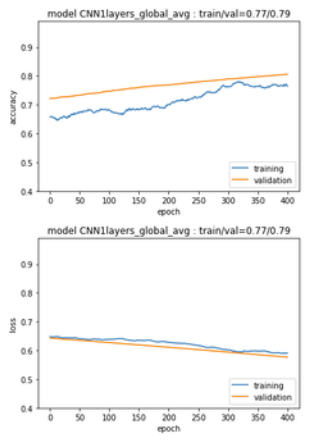
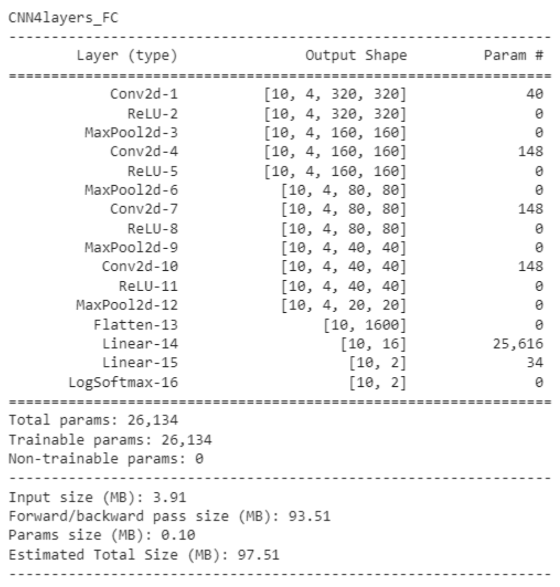


Figure 3: model: 1 layered CNN with global averaging.

   2.2 Since the previous model under fits our data, we now attempt to extend our network to 4 layers with a fully connected layer at the end. From the results below we see that this model performs well while training but fails to generalize the data on the validation set. Hence, is worse off on the validation data. This model overfits our training data.

   2.3 To tune the model a little better we replace the FC layer with a global averaging layer that takes a 4-channel input from the previous convolution layers and gives an output



```
CNN4layers_FC
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [10, 4, 320, 320]              40
              ReLU-2          [10, 4, 320, 320]               0
         MaxPool2d-3          [10, 4, 160, 160]               0
            Conv2d-4          [10, 4, 160, 160]             148
              ReLU-5          [10, 4, 160, 160]               0
         MaxPool2d-6            [10, 4, 80, 80]               0
            Conv2d-7            [10, 4, 80, 80]             148
              ReLU-8            [10, 4, 80, 80]               0
         MaxPool2d-9            [10, 4, 40, 40]               0
           Conv2d-10            [10, 4, 40, 40]             148
             ReLU-11            [10, 4, 40, 40]               0
        MaxPool2d-12            [10, 4, 20, 20]               0
          Flatten-13                 [10, 1600]               0
           Linear-14                   [10, 16]          25,616
           Linear-15                    [10, 2]              34
       LogSoftmax-16                    [10, 2]               0
================================================================
Total params: 26,134
Trainable params: 26,134
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 3.91
Forward/backward pass size (MB): 93.51
Params size (MB): 0.10
Estimated Total Size (MB): 97.51
----------------------------------------------------------------
```

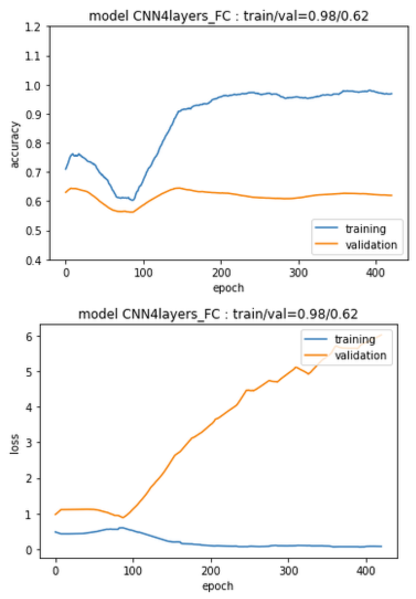
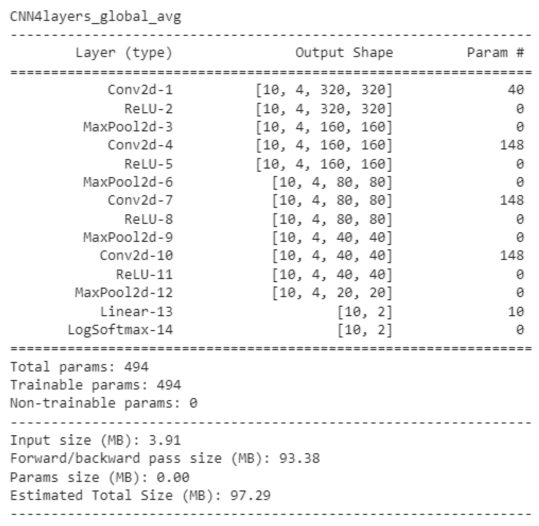


Figure 4: model: 4 layered CNN with a FC layer.

vector which is the classification result. We see that this model has the best performance compared to the previous models.



```
CNN4layers_global_avg
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [10, 4, 320, 320]              40
              ReLU-2          [10, 4, 320, 320]               0
         MaxPool2d-3          [10, 4, 160, 160]               0
            Conv2d-4          [10, 4, 160, 160]             148
              ReLU-5          [10, 4, 160, 160]               0
         MaxPool2d-6            [10, 4, 80, 80]               0
            Conv2d-7            [10, 4, 80, 80]             148
              ReLU-8            [10, 4, 80, 80]               0
         MaxPool2d-9            [10, 4, 40, 40]               0
           Conv2d-10            [10, 4, 40, 40]             148
             ReLU-11            [10, 4, 40, 40]               0
        MaxPool2d-12            [10, 4, 20, 20]               0
           Linear-13                    [10, 2]              10
       LogSoftmax-14                    [10, 2]               0
================================================================
Total params: 494
Trainable params: 494
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 3.91
Forward/backward pass size (MB): 93.38
Params size (MB): 0.00
Estimated Total Size (MB): 97.29
----------------------------------------------------------------
```

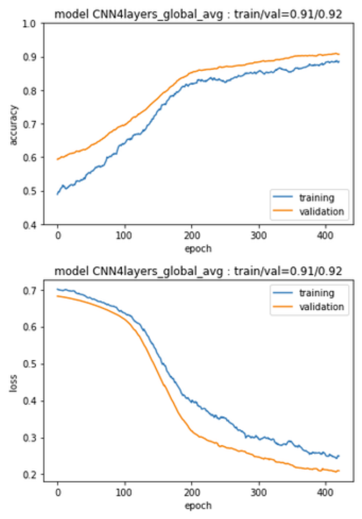
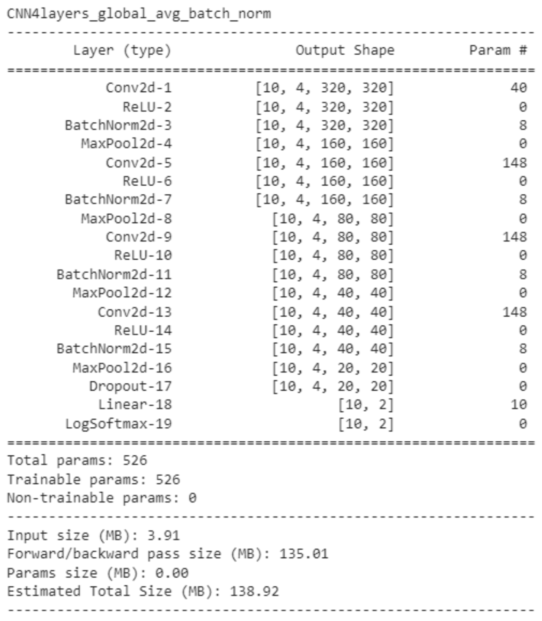


Figure 5: model: 4 layered CNN with global average.

3. An improvement is made on the layer with the following included:
   1. Increase data set size by augmenting training data.
   2. Introduce batch normalization
   3. Insert a dropout layer

4. We see that both the training and validation data accuracy and loss metrics out perform

the previous model.



```
CNN4layers_global_avg_batch_norm
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [10, 4, 320, 320]              40
              ReLU-2         [10, 4, 320, 320]               0
       BatchNorm2d-3         [10, 4, 320, 320]               8
         MaxPool2d-4         [10, 4, 160, 160]               0
            Conv2d-5         [10, 4, 160, 160]             148
              ReLU-6         [10, 4, 160, 160]               0
       BatchNorm2d-7         [10, 4, 160, 160]               8
         MaxPool2d-8           [10, 4, 80, 80]               0
            Conv2d-9           [10, 4, 80, 80]             148
             ReLU-10           [10, 4, 80, 80]               0
      BatchNorm2d-11           [10, 4, 80, 80]               8
        MaxPool2d-12           [10, 4, 40, 40]               0
           Conv2d-13           [10, 4, 40, 40]             148
             ReLU-14           [10, 4, 40, 40]               0
      BatchNorm2d-15           [10, 4, 40, 40]               8
        MaxPool2d-16           [10, 4, 20, 20]               0
          Dropout-17           [10, 4, 20, 20]               0
           Linear-18                   [10, 2]              10
       LogSoftmax-19                   [10, 2]               0
================================================================
Total params: 526
Trainable params: 526
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 3.91
Forward/backward pass size (MB): 135.01
Params size (MB): 0.00
Estimated Total Size (MB): 138.92
----------------------------------------------------------------
```

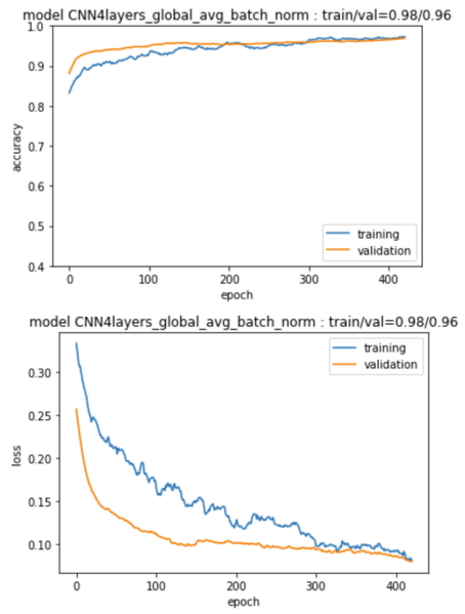


Figure 6: model: suggested model.