MILS – ASSIGNMENT I

RE6131058 蔡秉言

I. Task A: Designing a Convolution Module for Variable Input Channels

1. Base model

   I selected ResNet18 as the base model to compare with in this experiment. The first layer is a convolutional layer where kernel_size=7, stride=2, padding=3, followed by four blocks of which each includes four convolutional layers where kernel_size=3 and padding=1.

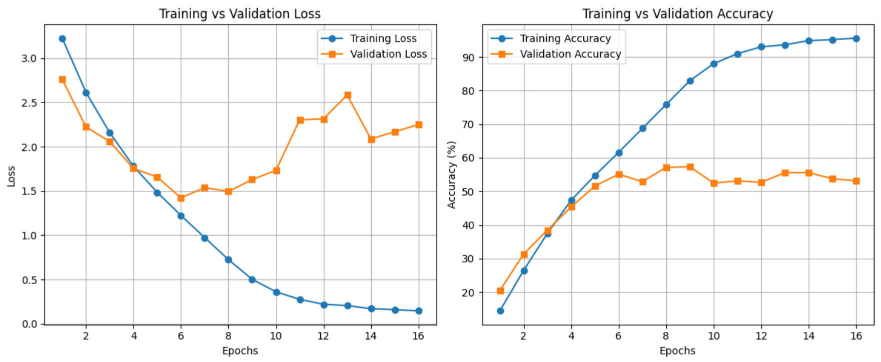
2. Dynamic convolutional module model

   The four basic blocks in the middle are remained the same as ResNet18. A dynamic convolutional layer is designed in the beginning to take an arbitrary number of channel of image by setting the max channel = 3. Then an MLP with input size 128 is connected to the dynamic convolutional layer to generate kernel weight.

3. Experiment results

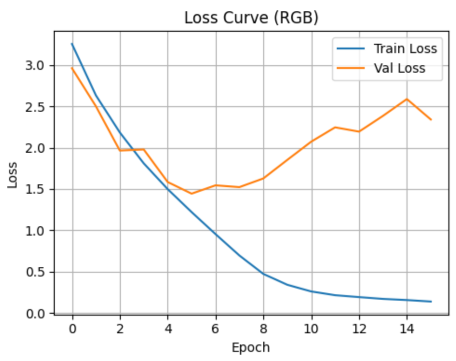
   A. Results of base model

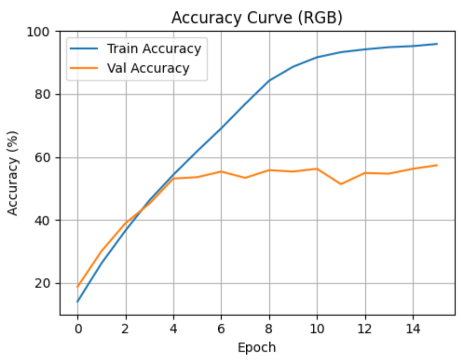
   The validation accuracy have reached 55.1% after 6 epochs, and the best accuracy 57.33% within 16 epochs in the base model.

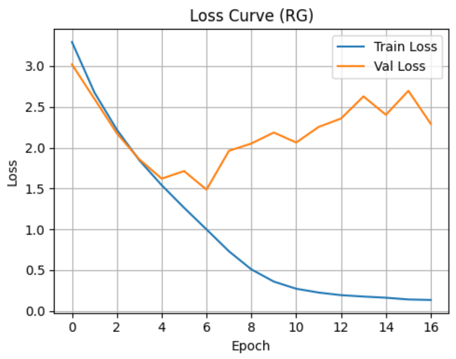


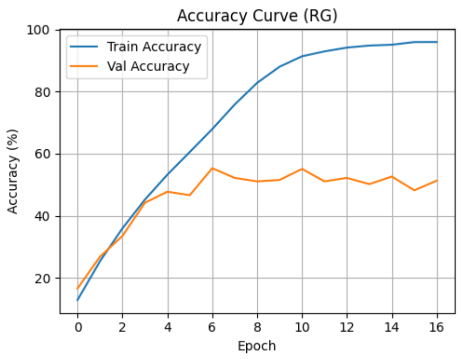
   B. Results of dynamic convolutional module model

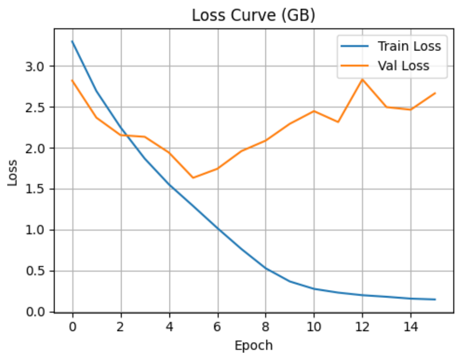
   The figures below are the loss curve and accuracy of RGB, RG, GB, RB, R, G, and B channel as well as the summary table of final results. In this model, all 7 conditions worked similarly but a bit off compared to the base model, with all of which reached over 45% accuracy after 6 epochs, and in most condition can reach around 50% within 20 epochs.

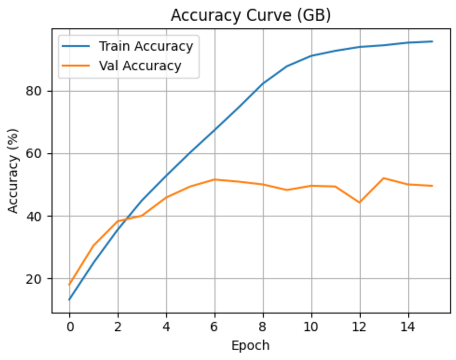

Loss Curve (RGB)
Train Loss
Val Loss
Loss
Epoch

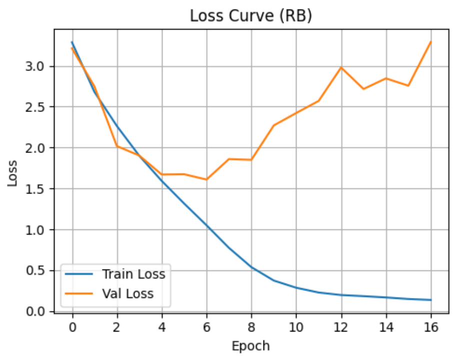

Accuracy Curve (RGB)
Train Accuracy
Val Accuracy
Accuracy (%)
Epoch

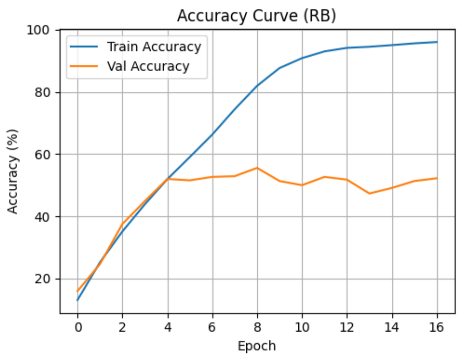

Loss Curve (RG)
Train Loss
Val Loss
Loss
Epoch

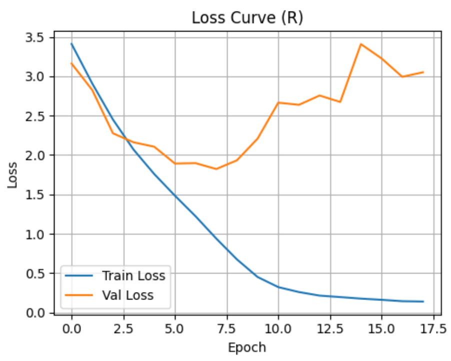

Accuracy Curve (RG)
Train Accuracy
Val Accuracy
Accuracy (%)
Epoch

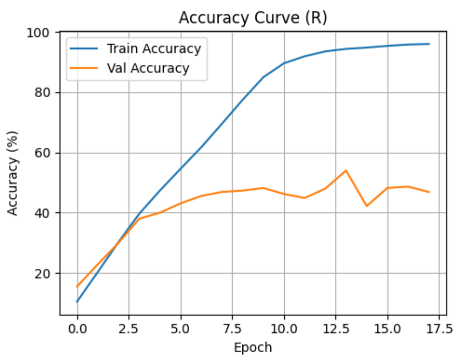

Loss Curve (GB)
Train Loss
Val Loss
Loss
Epoch

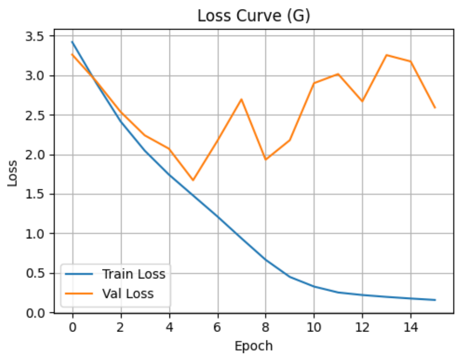

Accuracy Curve (GB)
Train Accuracy
Val Accuracy
Accuracy (%)
Epoch

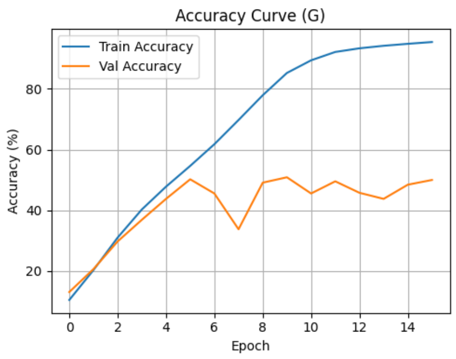

Loss Curve (RB)
Train Loss
Val Loss
Loss
Epoch

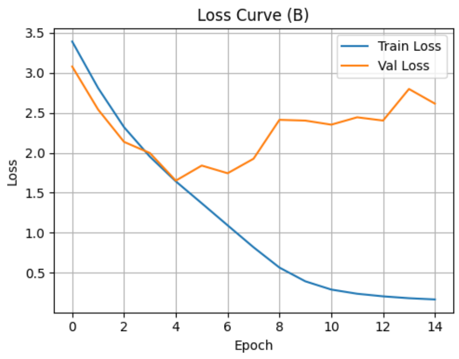

Accuracy Curve (RB)
Train Accuracy
Val Accuracy
Accuracy (%)
Epoch

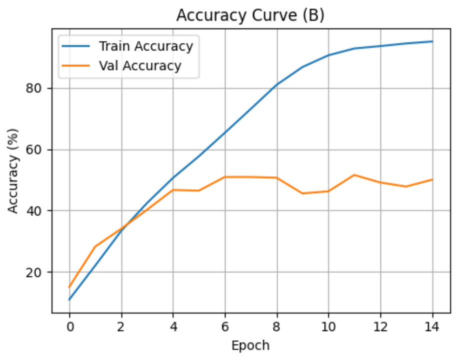


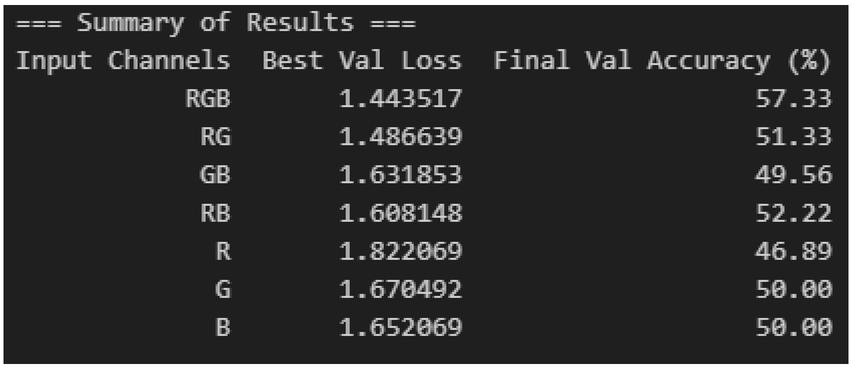






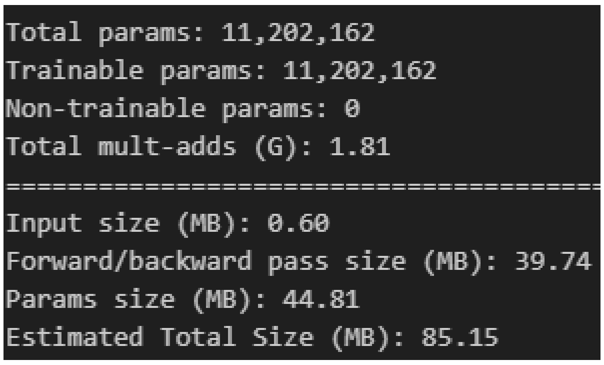





```
=== Summary of Results ===
Input Channels  Best Val Loss  Final Val Accuracy (%)
           RGB       1.443517                   57.33
            RG       1.486639                   51.33
            GB       1.631853                   49.56
            RB       1.608148                   52.22
             R       1.822069                   46.89
             G       1.670492                   50.00
             B       1.652069                   50.00
```

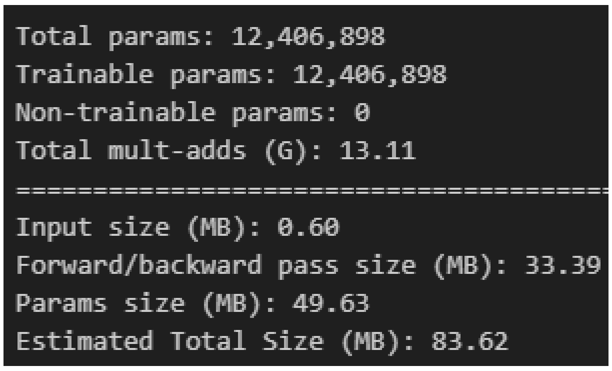
4. Discussion

It seems like the dynamic convolutional layer combining the kernel weight generator was working well from all the input channel adjustment experiment. No matter the input channel was 3, 2, or 1, they all have reached over 45% validation accuracy in 20 epochs.



```
Total params: 11,202,162
Trainable params: 11,202,162
Non-trainable params: 0
Total mult-adds (G): 1.81
========================================
Input size (MB): 0.60
Forward/backward pass size (MB): 39.74
Params size (MB): 44.81
Estimated Total Size (MB): 85.15
```

Fig. Model summary of base model



```
Total params: 12,406,898
Trainable params: 12,406,898
Non-trainable params: 0
Total mult-adds (G): 13.11
========================================
Input size (MB): 0.60
Forward/backward pass size (MB): 33.39
Params size (MB): 49.63
Estimated Total Size (MB): 83.62
```

Fig. Model summary of dynamic convolutional module model

In terms of parameter size, the base model ResNet18 has 11.2 million parameters and my proposed model has 12.4 million parameters, which was a 10.8% increase. For a more complex function, from the smoothness of loss curve and the outcome of validation accuracy, I reckon the increasement in parameter size is quite reasonable.

II. Task B: Designing a Two-Layer Network for Image Classification

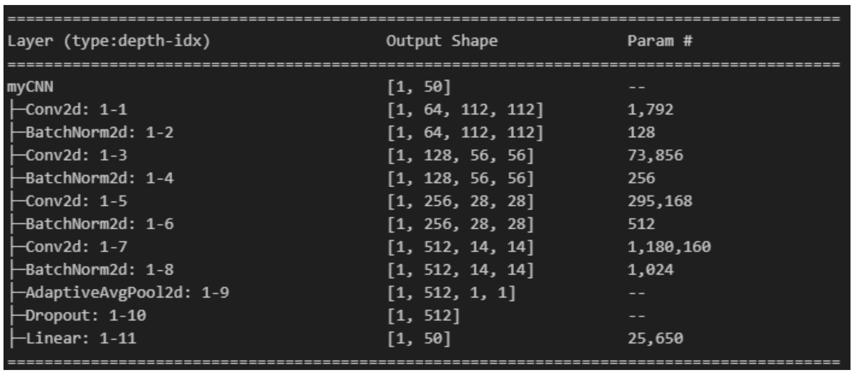
1. Base model

   ResNet34 is the base model to compare with in this experiment.

   The first layer is a convolutional layer where kernel_size=7, stride=2, padding=3, followed by four blocks of convolutional layers which each includes 6, 8, 12, 6 layers where kernel_size=3 and padding=1.

2. My 4-layer CNN model

   There are four convolutional layers followed by batch normalization on each block. The numbers of filters of each convolution layers are 64, 128, 256, and 512. The kernel size is 3, stride = 3, and padding = 1. In the end there is an adaptive average pooling layer and drop-out layer at 0.4 rate.

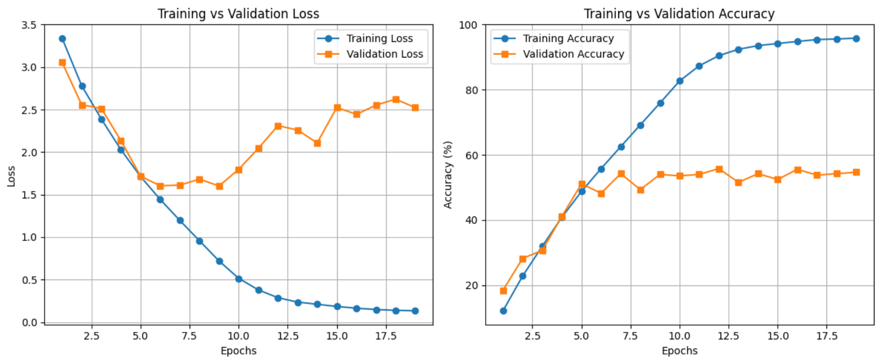


```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
myCNN                                    [1, 50]                   --
├─Conv2d: 1-1                            [1, 64, 112, 112]         1,792
├─BatchNorm2d: 1-2                       [1, 64, 112, 112]         128
├─Conv2d: 1-3                            [1, 128, 56, 56]          73,856
├─BatchNorm2d: 1-4                       [1, 128, 56, 56]          256
├─Conv2d: 1-5                            [1, 256, 28, 28]          295,168
├─BatchNorm2d: 1-6                       [1, 256, 28, 28]          512
├─Conv2d: 1-7                            [1, 512, 14, 14]          1,180,160
├─BatchNorm2d: 1-8                       [1, 512, 14, 14]          1,024
├─AdaptiveAvgPool2d: 1-9                 [1, 512, 1, 1]            --
├─Dropout: 1-10                          [1, 512]                  --
├─Linear: 1-11                           [1, 50]                   25,650
==========================================================================================
```

3. Experiment Results

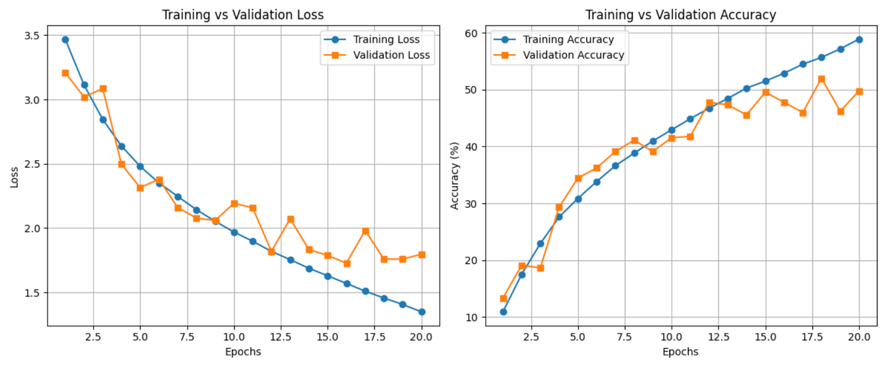
   A. Loss curve and accuracy of Base model (ResNet34)

      The validation accuracy nearly reached its bottleneck at 7 epochs which was 54.22% in 7 epochs and steadily remained its performance in the following 10 epochs.



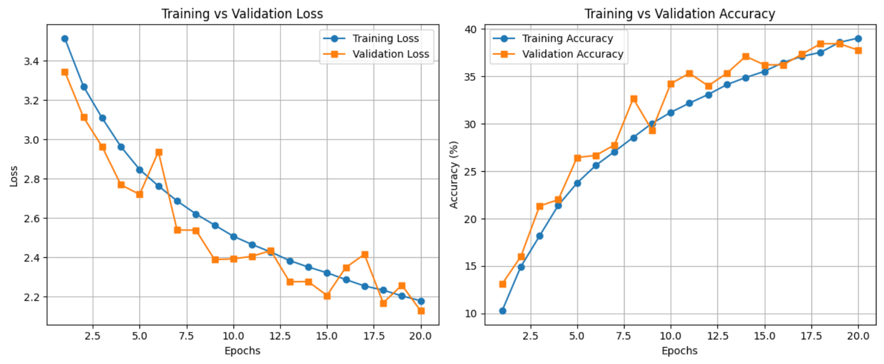
B. Loss curve and accuracy of my CNN model

The validation accuracy was only 39.11% at 7 epochs, but it gradually increased to 52% within 20 epochs.

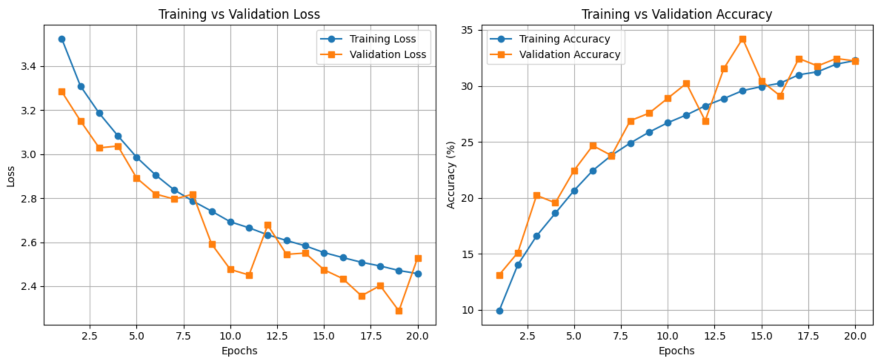


4. Ablation Study

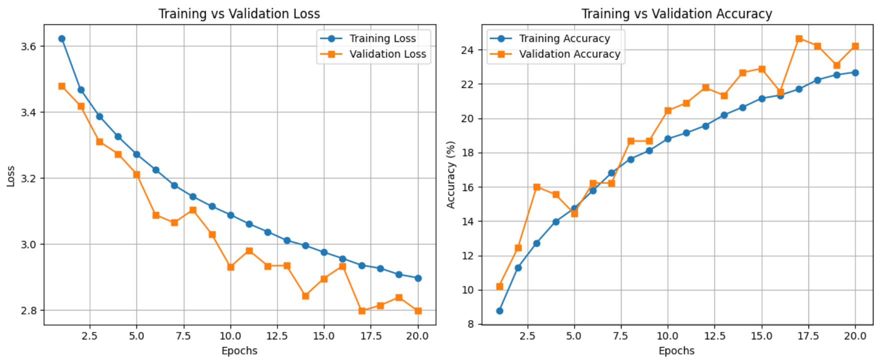
	Validation Accuracy
My 4-layer CNN model	52%
Remove convolution layer 1	38.44%
Remove convolution layer 4	32.44%
Adjust learning rate to 0.0001	24.67%



Remove layer 1



Remove layer 4



Adjust learning rate to 0.0001

5. Discussion

Theoretically, it is difficult to beat the base model ResNet 34 with only 4 effective layers, somehow it did perform well as there is enough training time, about 20 epochs, in our experiment. Though the four layers have the same kernel size, the receptive field is still getting larger and be able to capture global features in the design. From the ablation study, removing one layer off tremendously impacts the result by about 20% accuracy. Especially when the fourth layer was removed, the number of parameters decreased significantly, and the accuracy was only 32.44%. Surprisingly, the learning rate 0.001 may be based on many researchers' experience. Adjusting it to 0.0001 did not find the best convergence. It slowed down the convergence too much and hasn't reached its minimum loss within 20 epochs. Which indicates a constant 0.001 learning rate may be optimal, or a complex learning rate decay schedule need to be conducted in this case.