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.

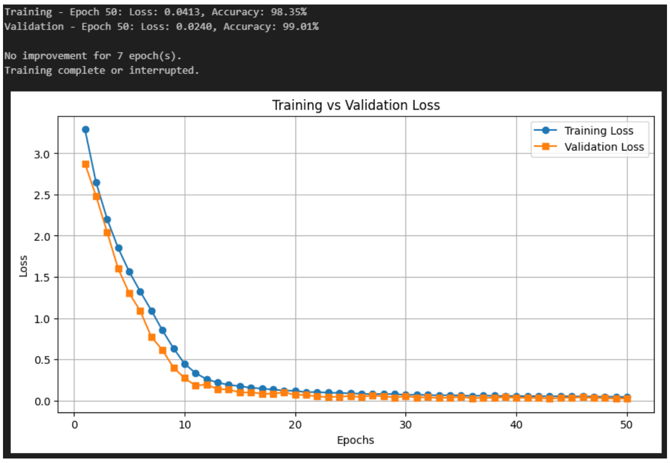
2. Dynamic convolutional module model

   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 91.13% after 10 epochs, and gradually reached 99.01% after 50 epochs in the base model.



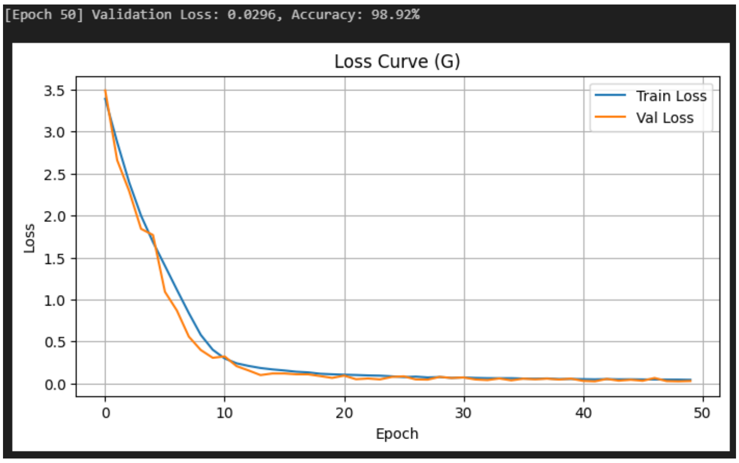
```
Training - Epoch 50: Loss: 0.0413, Accuracy: 98.35%
Validation - Epoch 50: Loss: 0.0240, Accuracy: 99.01%

No improvement for 7 epoch(s).
Training complete or interrupted.
```

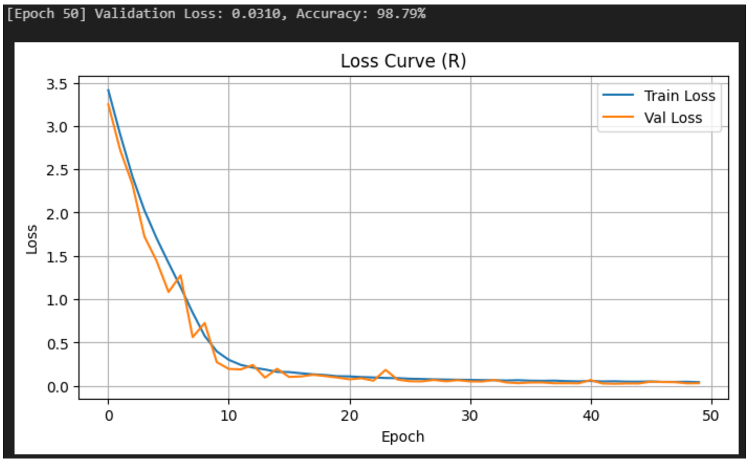
   B. Results of dynamic convolutional module model

   The figures below are the loss curve of RGB, RG, GB, RB, R, G, and B channel as well as the summary table of final results.

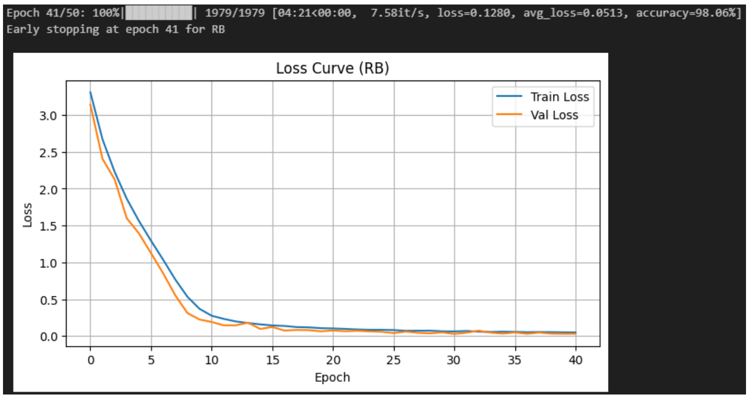
   All 7 conditions worked well in this model, with all of which reached over 90% accuracy after 10 epochs, and reached over 98% accuracy after 50 epochs.



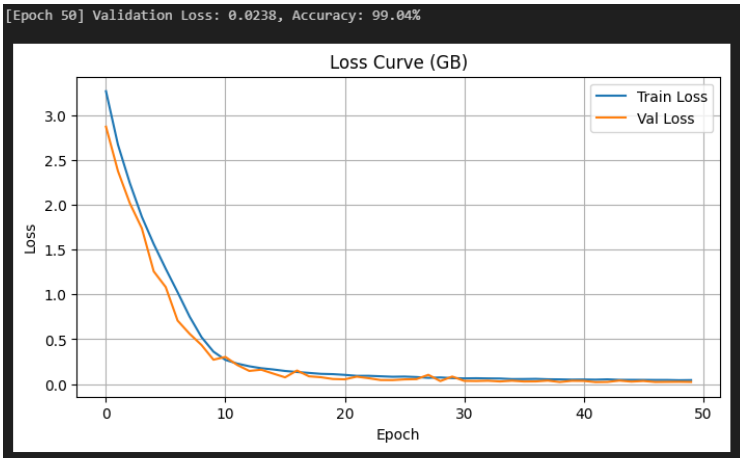
```
[Epoch 50] Validation Loss: 0.0296, Accuracy: 98.92%
```



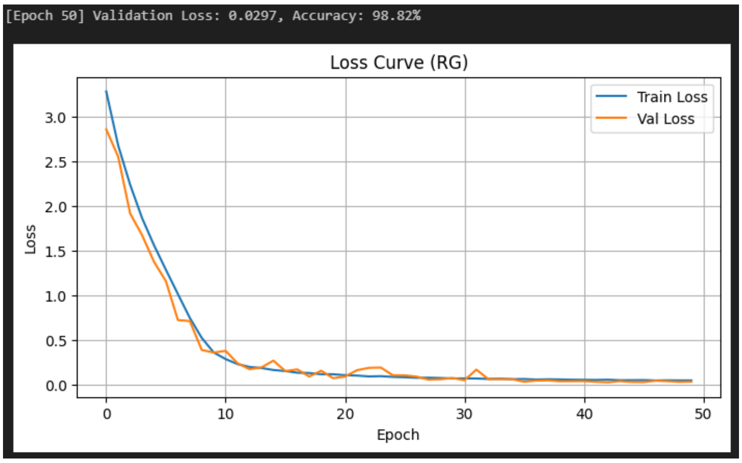
```
[Epoch 50] Validation Loss: 0.0310, Accuracy: 98.79%
```



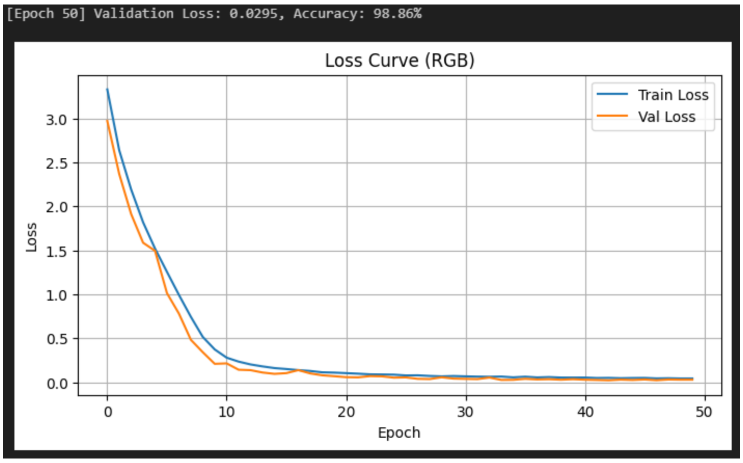
```
Epoch 41/50: 100%|██████████| 1979/1979 [04:21<00:00,  7.58it/s, loss=0.1280, avg_loss=0.0513, accuracy=98.06%]
Early stopping at epoch 41 for RB
```



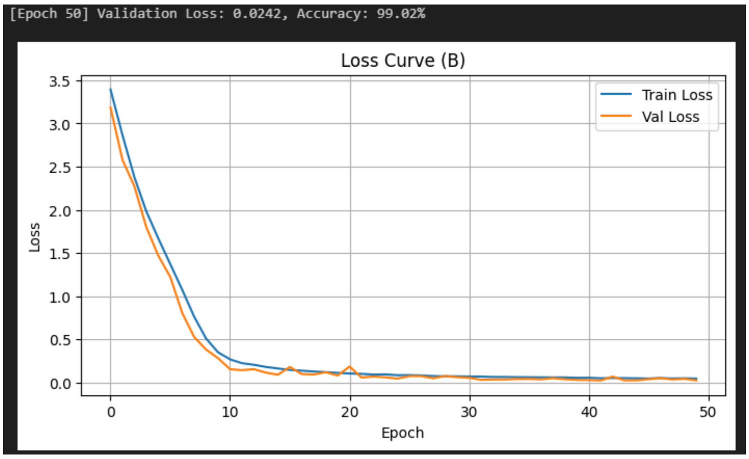
```
[Epoch 50] Validation Loss: 0.0238, Accuracy: 99.04%
```



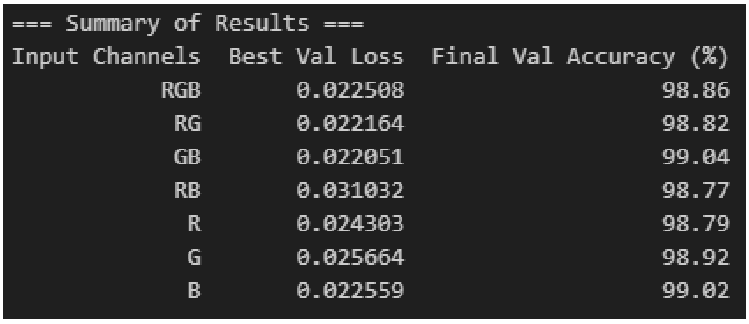
```
[Epoch 50] Validation Loss: 0.0297, Accuracy: 98.82%
```



```
[Epoch 50] Validation Loss: 0.0295, Accuracy: 98.86%
```



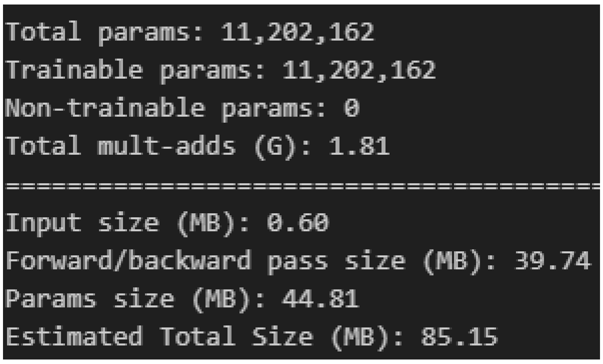
```
[Epoch 50] Validation Loss: 0.0242, Accuracy: 99.02%
```



```
=== Summary of Results ===
Input Channels  Best Val Loss  Final Val Accuracy (%)
           RGB       0.022508                   98.86
            RG       0.022164                   98.82
            GB       0.022051                   99.04
            RB       0.031032                   98.77
             R       0.024303                   98.79
             G       0.025664                   98.92
             B       0.022559                   99.02
```

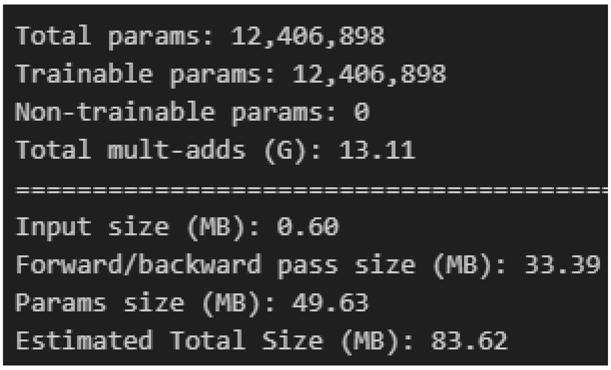
4. Discussion

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 98% validation accuracy in 50 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

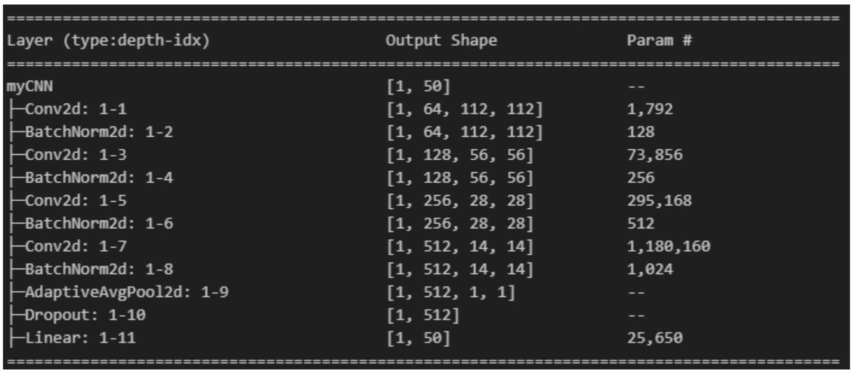
1. Base model

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.

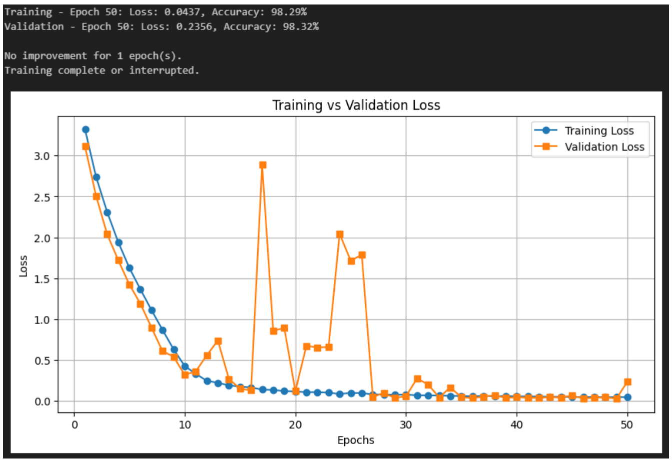


```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
myCNN                                    [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 of Base model (ResNet34)

Although the loss fluctuated in 10 to 30 epochs, the validation accuracy reached 91.85% in 10 epochs and steadily increased to 98.32% in the following 40 epochs.

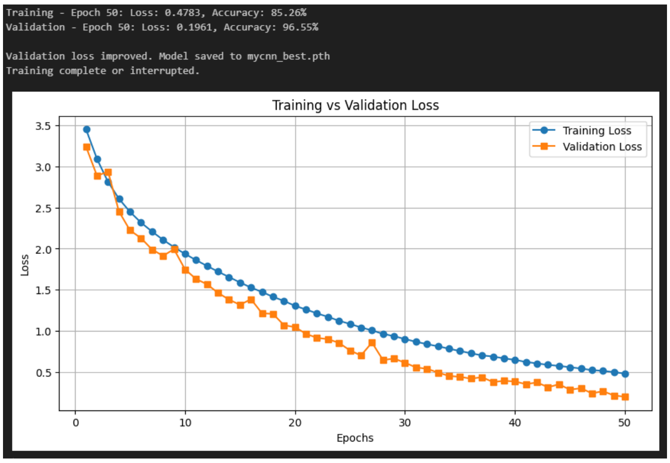


```
Training - Epoch 50: Loss: 0.0437, Accuracy: 98.29%
Validation - Epoch 50: Loss: 0.2356, Accuracy: 98.32%

No improvement for 1 epoch(s).
Training complete or interrupted.
```

B. Loss curve of my CNN model

The validation accuracy was only 50.11% at 10 epochs, 77.9% at 25 epochs, 90.33% at 38 epochs and finally reached 96.55% at 50 epochs.

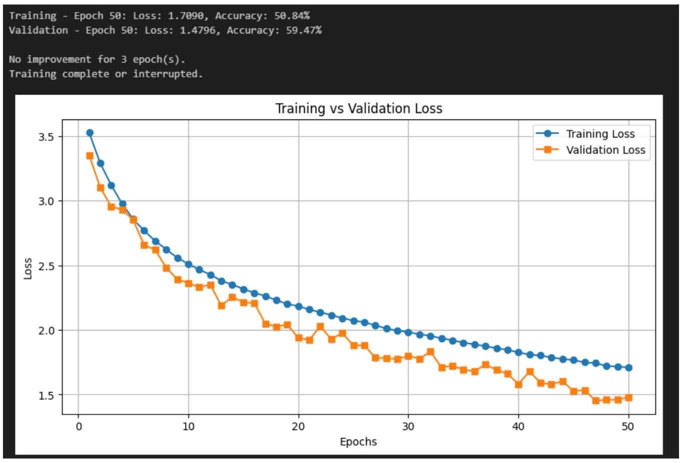


```
Training - Epoch 50: Loss: 0.4783, Accuracy: 85.26%
Validation - Epoch 50: Loss: 0.1961, Accuracy: 96.55%

Validation loss improved. Model saved to mycnn_best.pth
Training complete or interrupted.
```

4. Ablation Study

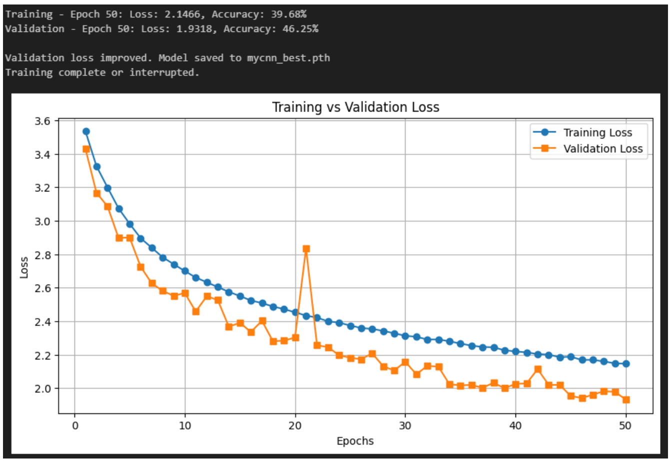
	Validation Accuracy
My 4-layer CNN model	96.55%
Remove convolution layer 1	59.47%
Remove convolution layer 4	46.25%
Adjust learning rate to 0.01	59.91%



```
Training - Epoch 50: Loss: 1.7090, Accuracy: 50.84%
Validation - Epoch 50: Loss: 1.4796, Accuracy: 59.47%

No improvement for 3 epoch(s).
Training complete or interrupted.
```

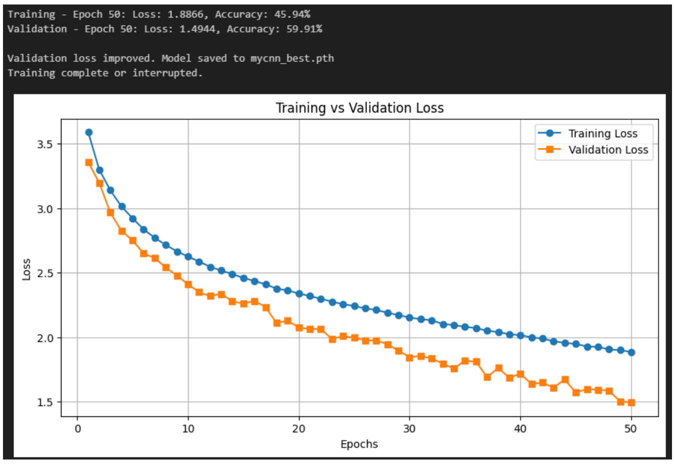
Remove layer 1



```
Training - Epoch 50: Loss: 2.1466, Accuracy: 39.68%
Validation - Epoch 50: Loss: 1.9318, Accuracy: 46.25%

Validation loss improved. Model saved to mycnn_best.pth
Training complete or interrupted.
```

Remove layer 4



```
Training - Epoch 50: Loss: 1.8866, Accuracy: 45.94%
Validation - Epoch 50: Loss: 1.4944, Accuracy: 59.91%

Validation loss improved. Model saved to mycnn_best.pth
Training complete or interrupted.
```

Adjust learning rate to 0.01

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 50 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 40% accuracy. Especially when the fourth layer was removed, the number of parameters decreased significantly, and the accuracy was only 46.25%. Surprisingly, the learning rate 0.001 may be based on many researchers' experience. Adjusting it to 0.01 did not speed up the convergence. On the contrary, it slowed down the convergence and only reached 59.91% accuracy in 50 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.