HW 5

1 Problem Statement

In this programming assignment, we train a convolutional neural network on the CIFAR-10 dataset (link). The dataset has 50000 training samples and 5000 testing samples. Each sample is a 32×32×3 image and the model need to classify it into the correct category. We choose the batch size as 500 and the convolutional model structure can be seen in figure 1:

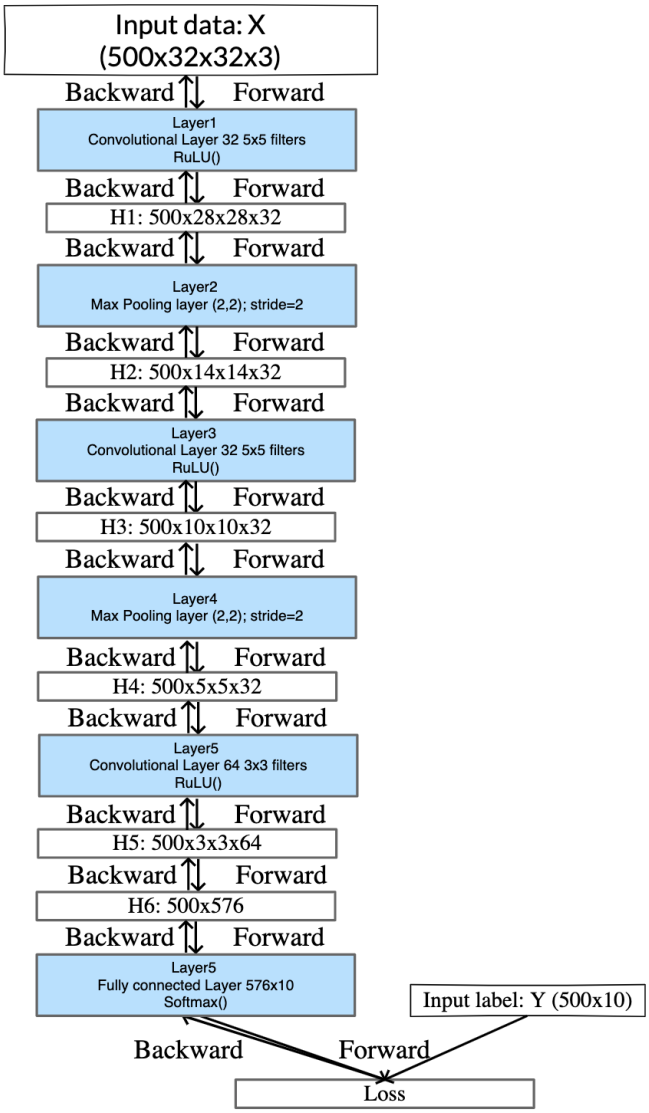

Figure 1: Neural Network Structure

2 Pseudo-code

Algorithm 1 Neural Network Training

Initialize Model parameter $\mathbf{\Theta}^0$, training data and labels $\mathbf{X}, \mathbf{Y}$, testing data and labels $\mathbf{X}_{test}, \mathbf{Y}_{test}$, learning rate α
for $t = 0, 1, 2, \ldots, T$ **do**
  take batch data and labels $\mathbf{X}^t, \mathbf{Y}^t$
  compute loss: $\mathcal{L}(\mathbf{\Theta}^t; \mathbf{X}^t, \mathbf{Y}^t)$
  compute testing accuracy and classification errors
  compute gradient: $\nabla_{\mathbf{\Theta}} \mathcal{L}(\mathbf{\Theta}^t; \mathbf{X}^t, \mathbf{Y}^t)$
  optimize parameters : $\mathbf{\Theta}^{t+1} = \textbf{optimizer}(\mathbf{\Theta}^t, \nabla_{\mathbf{\Theta}} \mathcal{L}(\mathbf{\Theta}^t; \mathbf{X}^t, \mathbf{Y}^t))$
  if An epoch of data have been trained **then**
    compute testing loss $\mathcal{L}(\mathbf{\Theta}^t; \mathbf{X}_{test}, \mathbf{Y}_{test})$
    compute testing accuracy and classification errors
  end if
end for

3 Environment setting

We choose the batch size as 500. The optimizer is Adam and we choose the learning rate (step size) as 0.001. We train the model on the RTX 2080 graphic card for 80 epochs. The training accuracy reaches above 80% and the testing accuracy reaches above 70%. I submit this question as jupyter notebook file because the model is not very large and can be run on jupyter notebook

4 Loss and accuracy curves

The environment we implement our codes is tensorflow 2.0 with RTX 2080 Ti.
Figure 2a and 2b are error curves and accuracy curves respectively.

(a) loss curves (b) Accuracy curves

Figure 2: Loss and accuracy curves

5 Change model

Next we change our model to ResNet-18. I have to submit the codes for this question by python script because the model is large enough and can only run on GPU devices. We change the batch size as 100 and ran 30 epochs. The accuracy is higher than the previous model.

```
Epoch: 1 training loss: 1.4768442 training accuracy 0.45224005 testing loss 1.4681258 testing accuracy 0.45249996
Epoch: 2 training loss: 1.1565504 training accuracy 0.57920015 testing loss 1.1728756 testing accuracy 0.57860017
Epoch: 3 training loss: 0.9906448 training accuracy 0.64251983 testing loss 1.0287963 testing accuracy 0.6319001
Epoch: 4 training loss: 0.8666021 training accuracy 0.68420047 testing loss 0.9324807 testing accuracy 0.6615999
Epoch: 5 training loss: 0.7029386 training accuracy 0.7520199 testing loss 0.79753464 testing accuracy 0.7243999
Epoch: 6 training loss: 0.65965945 training accuracy 0.77137977 testing loss 0.7928581 testing accuracy 0.73440003
Epoch: 7 training loss: 0.60921085 training accuracy 0.7860194 testing loss 0.79009086 testing accuracy 0.7409
Epoch: 8 training loss: 0.5776866 training accuracy 0.7944393 testing loss 0.8330753 testing accuracy 0.7374
Epoch: 9 training loss: 0.44643795 training accuracy 0.8415598 testing loss 0.7463656 testing accuracy 0.7675
Epoch: 10 training loss: 0.541157 training accuracy 0.81223917 testing loss 0.9296771 testing accuracy 0.7264
Epoch: 11 training loss: 0.35830873 training accuracy 0.87440073 testing loss 0.8169637 testing accuracy 0.7657001
Epoch: 12 training loss: 0.29848754 training accuracy 0.8960607 testing loss 0.8251926 testing accuracy 0.7757999
Epoch: 13 training loss: 0.18980473 training accuracy 0.9347408 testing loss 0.851542 testing accuracy 0.79290015
Epoch: 14 training loss: 0.19446318 training accuracy 0.93344027 testing loss 0.92454666 testing accuracy 0.7817001
Epoch: 15 training loss: 0.15705743 training accuracy 0.94534093 testing loss 0.9181235 testing accuracy 0.78820014
Epoch: 16 training loss: 0.106157586 training accuracy 0.9630209 testing loss 0.93104225 testing accuracy 0.8023001
Epoch: 17 training loss: 0.10959086 training accuracy 0.9613007 testing loss 0.9351486 testing accuracy 0.80329984
Epoch: 18 training loss: 0.10493816 training accuracy 0.96424055 testing loss 0.9696451 testing accuracy 0.7918
Epoch: 19 training loss: 0.11381411 training accuracy 0.9597408 testing loss 1.145103 testing accuracy 0.78379995
Epoch: 20 training loss: 0.05837743 training accuracy 0.98002017 testing loss 1.0619949 testing accuracy 0.80049986
Epoch: 21 training loss: 0.06337192 training accuracy 0.9788403 testing loss 1.1006219 testing accuracy 0.79620004
Epoch: 22 training loss: 0.08704442 training accuracy 0.97280055 testing loss 1.1576781 testing accuracy 0.7950001
Epoch: 23 training loss: 0.042850412 training accuracy 0.9854397 testing loss 1.0850525 testing accuracy 0.8045001
Epoch: 24 training loss: 0.022563914 training accuracy 0.99283934 testing loss 1.0471674 testing accuracy 0.8126001
Epoch: 25 training loss: 0.043063305 training accuracy 0.9855203 testing loss 1.1402146 testing accuracy 0.80150014
Epoch: 26 training loss: 0.030424362 training accuracy 0.98988 testing loss 1.018499 testing accuracy 0.8068001
Epoch: 27 training loss: 0.020503132 training accuracy 0.9932192 testing loss 1.0377213 testing accuracy 0.8159998
Epoch: 28 training loss: 0.022374304 training accuracy 0.9921796 testing loss 1.007716 testing accuracy 0.8144
Epoch: 29 training loss: 0.018063089 training accuracy 0.9943196 testing loss 1.1385915 testing accuracy 0.8177998
Epoch: 30 training loss: 0.026114795 training accuracy 0.9917994 testing loss 1.111399 testing accuracy 0.8114002
```

Figure 3: Results of question1.2

6 Data augmentation

We added augmentation (image flipping and contrasting color changing) code on the data processing part. We train the images on the same model as the previous one. The testing accuracy reaches 62%.