Two late day using

Project2 – Report

Hanjie(Huntley) Liu

Team: Fsy

Team member: Hanjie(Huntley), Liu Mike Wang, Ying Zhou

Student number: 301404949

In part one, I used Alexnet as my CNN base structure sample, with many individual changes. This network is training with CIFAR-100,.

At beginning. The normalize parameter are 0 mean and 1 dev and crop to 32 * 32 with 4 padding sizes, horizontal flip with chance of 50% for strengthen training data.

We used 8 convolution kernels and a 2 x 2 Maxpooling kernel with stride 2.
Some kernels are for deepen our learning and repeat convolution.

For each convolution we give a BatchNorm1d and BatchNorm2d to Normalize output and increase the learning speed.

We have 30 layers in the high-dimension region. Our FC layer has 4 Linear operation,

We use ReLU as our activation function.

We use Dropout() and set its probability to 0.5 to decrease overfitting

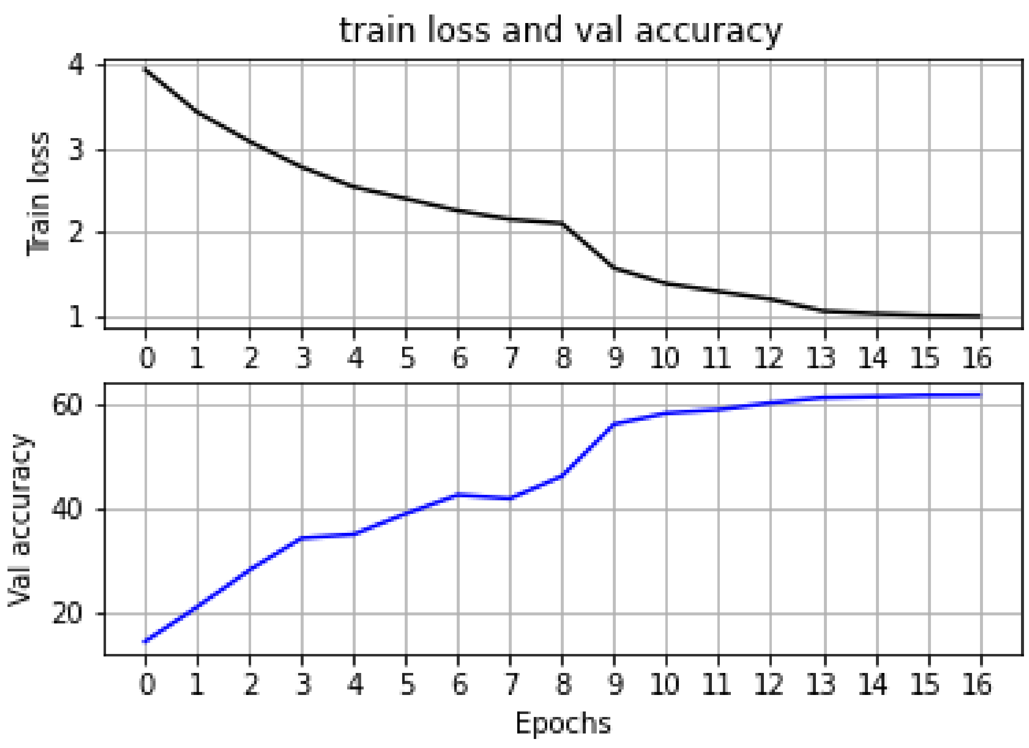
We use 32 batch and run 17 EPOCHS, with learning rate Start by 0.01 and momentum of 0.9, and weight decay is 0.001. We schedule that on epoch 9th, 13th, 15th, learning rate will be divided by 10.

Ablation study: We have above actions to improve our model and increasing our profermance, but the most intuitive action is making a schedule for learning rate. We found once the Accuracy and loss tend to be invarient or influction, the learning rate decreasing will largely improved it.

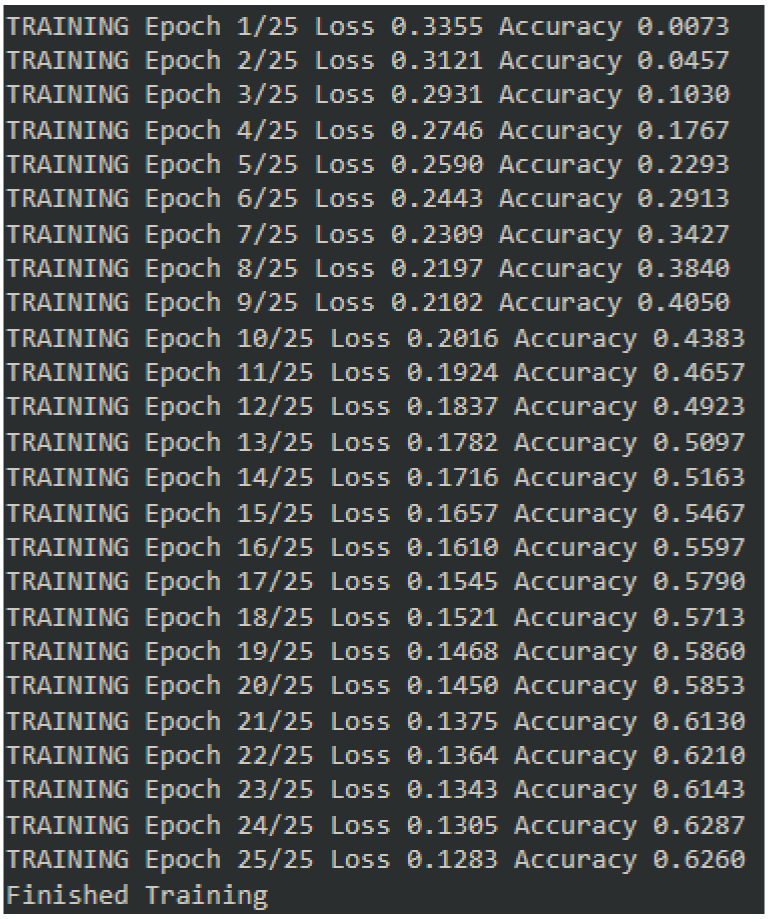
The best traning accuracy reached 65% on Colab and test score is 0.51 on Kaggle

Layer No.	Layer Type	Kernel Size (Conv Layer)	Input|Output Dimension	Input|Output Channels (Conv Layer)
1	conv2d	3	32|32	3|96
2	relu	-	32|32	-
3	bn	-	32|32	-
4	conv2d	3	32|32	96|256
5	relu	-	32|32	-
6	bn	-	32|32	-
7	mpool	2	32|16	-
8	conv2d	3	16|16	256|384
9	relu	-	16|16	-
10	bn	-	16|16	-
11	conv2d	3	16|16	384|256
12	relu	-	16|16	-
13	bn	-	16|16	-
14	conv2d	3	16|16	256|384
15	relu	-	16|16	-
16	bn	-	16|16	-
17	mpool	2	16|8	-
18	conv2d	3	8|8	384|512
19	relu	-	8|8	-
20	bn	-	8|8	-
21	conv2d	3	8|8	512|384
22	relu	-	8|8	-
23	bn	-	8|8	-

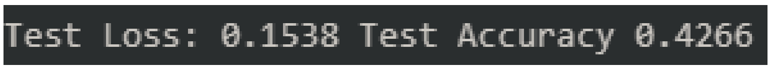
24	conv2d	3	8|8	384|512
25	relu	-	8|8	-
26	bn	-	8|8	-
27	conv2d	3	8|8	512|512
28	relu	-	8|8	-
29	bn	-	8|8	-
30	mpool	2	8|4	-
31	linear	-	8192|1024	-
32	relu	-	1024|1024	-
33	bn	-	1024|1024	-
34	linear	-	1024|1024	-
35	relu	-	1024|1024	-
36	bn	-	1024|1024	-
37	linear	-	1024|100	-
38	relu	-	100|100	-
39	linear	-	100|50	-
40	relu	-	50|50	-
41	linear	-	50|100	-



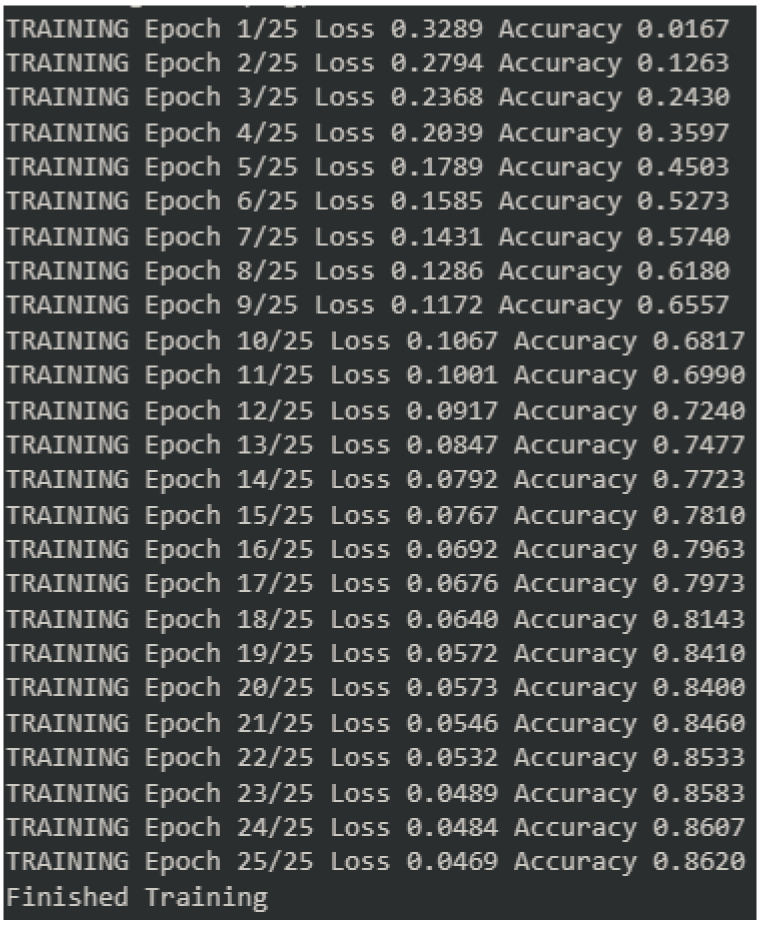
For part 2, I replaced the fully connected layer with nn.Sequential(nn.Linear(num_feats, num_classes)), and added few transformations. Here is the results from only having the last layer trainable for the training phase:


```
TRAINING Epoch 1/25 Loss 0.3355 Accuracy 0.0073
TRAINING Epoch 2/25 Loss 0.3121 Accuracy 0.0457
TRAINING Epoch 3/25 Loss 0.2931 Accuracy 0.1030
TRAINING Epoch 4/25 Loss 0.2746 Accuracy 0.1767
TRAINING Epoch 5/25 Loss 0.2590 Accuracy 0.2293
TRAINING Epoch 6/25 Loss 0.2443 Accuracy 0.2913
TRAINING Epoch 7/25 Loss 0.2309 Accuracy 0.3427
TRAINING Epoch 8/25 Loss 0.2197 Accuracy 0.3840
TRAINING Epoch 9/25 Loss 0.2102 Accuracy 0.4050
TRAINING Epoch 10/25 Loss 0.2016 Accuracy 0.4383
TRAINING Epoch 11/25 Loss 0.1924 Accuracy 0.4657
TRAINING Epoch 12/25 Loss 0.1837 Accuracy 0.4923
TRAINING Epoch 13/25 Loss 0.1782 Accuracy 0.5097
TRAINING Epoch 14/25 Loss 0.1716 Accuracy 0.5163
TRAINING Epoch 15/25 Loss 0.1657 Accuracy 0.5467
TRAINING Epoch 16/25 Loss 0.1610 Accuracy 0.5597
TRAINING Epoch 17/25 Loss 0.1545 Accuracy 0.5790
TRAINING Epoch 18/25 Loss 0.1521 Accuracy 0.5713
TRAINING Epoch 19/25 Loss 0.1468 Accuracy 0.5860
TRAINING Epoch 20/25 Loss 0.1450 Accuracy 0.5853
TRAINING Epoch 21/25 Loss 0.1375 Accuracy 0.6130
TRAINING Epoch 22/25 Loss 0.1364 Accuracy 0.6210
TRAINING Epoch 23/25 Loss 0.1343 Accuracy 0.6143
TRAINING Epoch 24/25 Loss 0.1305 Accuracy 0.6287
TRAINING Epoch 25/25 Loss 0.1283 Accuracy 0.6260
Finished Training
```

Here is for the testing phase:

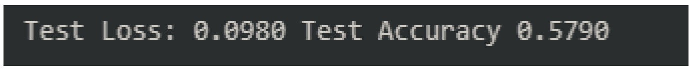

```
Test Loss: 0.1538 Test Accuracy 0.4266
```

In contrast, here is the results from having the whole model trainable:



```
TRAINING Epoch 1/25 Loss 0.3289 Accuracy 0.0167
TRAINING Epoch 2/25 Loss 0.2794 Accuracy 0.1263
TRAINING Epoch 3/25 Loss 0.2368 Accuracy 0.2430
TRAINING Epoch 4/25 Loss 0.2039 Accuracy 0.3597
TRAINING Epoch 5/25 Loss 0.1789 Accuracy 0.4503
TRAINING Epoch 6/25 Loss 0.1585 Accuracy 0.5273
TRAINING Epoch 7/25 Loss 0.1431 Accuracy 0.5740
TRAINING Epoch 8/25 Loss 0.1286 Accuracy 0.6180
TRAINING Epoch 9/25 Loss 0.1172 Accuracy 0.6557
TRAINING Epoch 10/25 Loss 0.1067 Accuracy 0.6817
TRAINING Epoch 11/25 Loss 0.1001 Accuracy 0.6990
TRAINING Epoch 12/25 Loss 0.0917 Accuracy 0.7240
TRAINING Epoch 13/25 Loss 0.0847 Accuracy 0.7477
TRAINING Epoch 14/25 Loss 0.0792 Accuracy 0.7723
TRAINING Epoch 15/25 Loss 0.0767 Accuracy 0.7810
TRAINING Epoch 16/25 Loss 0.0692 Accuracy 0.7963
TRAINING Epoch 17/25 Loss 0.0676 Accuracy 0.7973
TRAINING Epoch 18/25 Loss 0.0640 Accuracy 0.8143
TRAINING Epoch 19/25 Loss 0.0572 Accuracy 0.8410
TRAINING Epoch 20/25 Loss 0.0573 Accuracy 0.8400
TRAINING Epoch 21/25 Loss 0.0546 Accuracy 0.8460
TRAINING Epoch 22/25 Loss 0.0532 Accuracy 0.8533
TRAINING Epoch 23/25 Loss 0.0489 Accuracy 0.8583
TRAINING Epoch 24/25 Loss 0.0484 Accuracy 0.8607
TRAINING Epoch 25/25 Loss 0.0469 Accuracy 0.8620
Finished Training
```

Here is for the testing phase:



```
Test Loss: 0.0980 Test Accuracy 0.5790
```

We set following variable:

Traning_Rate: 0.001,
NUM_EPOCHS: 25,
BATCH_SIZE: 16

Training and Test data transformation:

```
transforms.Resize(256),
    transforms.RandomResizedCrop(224),
transforms.Normalize((0,0,0), (1,1,1)),
```