Project2 Report
Student name: Hankun Wang
Student id: 301416863
Kaggle group name: Fsy
Late day: 1 day

Part 1

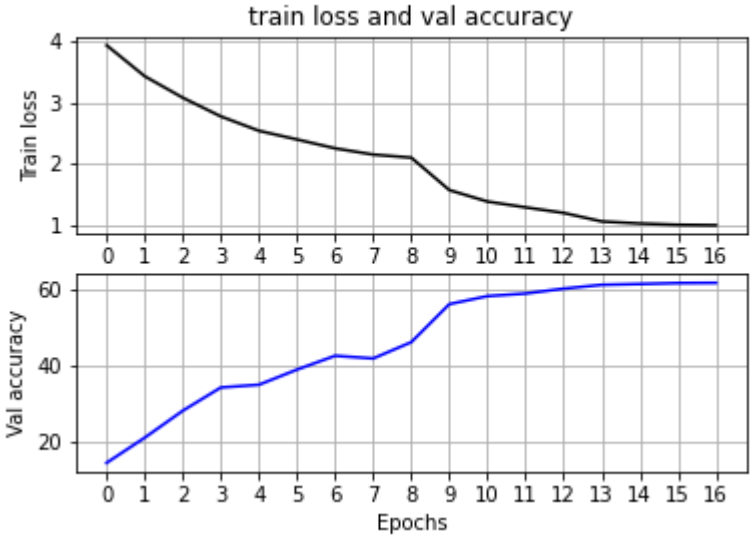
Our network structure is shown below. We have totally 41 layers, among which there are 30 layers in the high-dimension region (including conv2d, relu, bn and mpool). Our FC part contains 11 layers.

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	-

As shown above. We set each conv unit as a combination of three layers: conv2d, relu and BatchNorm(bn). We set 9 conv units totally and put max pooling at layer No.7, No.17 and No.30 to extract features and lower image dimensions. Our final output dimension after the last mpool is 4 * 4 * 512. Our first two linear layers are to strengthen features, thus we put bn after them. The last two linear layers are to describe features from 100 (TOTAL_CLASSES) pics dataset, thus we didn't put bn. This network is basically a simplified version of VGG 16 network.

Our plot of loss & accuracy is shown below:



We choose 17 as our epochs since we found training increasing rate after 17 will become very low, i.e. the accuracy and loss changes slowly. Therefore, we consider if that's the case of overfitting.
The best accuracy is about 65%, but when uploading on Kaggle, the highest score is only 0.517. This is the best accuracy we have.

Ablation study: After finishing our network construction, we find that our accuracy increases very slowly and the highest accuracy is about 52%. We did a lot of study and found out it may be due to too high learning rate or too low learning rate. We finally changed the lr to 0.01 initially and set a timer to change its lr at the 9th, the 13th and the 15th train to make it fit our model. After this change, our accuracy increases greatly.

Part 2

I Use epochs as 50, and get output as below:

TRAINING Epoch 1/50 Loss 0.6799 Accuracy 0.0033
TRAINING Epoch 2/50 Loss 0.6664 Accuracy 0.0073
TRAINING Epoch 3/50 Loss 0.6560 Accuracy 0.0187
TRAINING Epoch 4/50 Loss 0.6478 Accuracy 0.0217
TRAINING Epoch 5/50 Loss 0.6436 Accuracy 0.0217
TRAINING Epoch 6/50 Loss 0.6342 Accuracy 0.0363
TRAINING Epoch 7/50 Loss 0.6277 Accuracy 0.0400
TRAINING Epoch 8/50 Loss 0.6208 Accuracy 0.0573
TRAINING Epoch 9/50 Loss 0.6151 Accuracy 0.0753
TRAINING Epoch 10/50 Loss 0.6071 Accuracy 0.0780
TRAINING Epoch 11/50 Loss 0.6001 Accuracy 0.1037
TRAINING Epoch 12/50 Loss 0.5956 Accuracy 0.1070
TRAINING Epoch 13/50 Loss 0.5884 Accuracy 0.1237
TRAINING Epoch 14/50 Loss 0.5823 Accuracy 0.1373
TRAINING Epoch 15/50 Loss 0.5757 Accuracy 0.1573
TRAINING Epoch 16/50 Loss 0.5707 Accuracy 0.1547
TRAINING Epoch 17/50 Loss 0.5651 Accuracy 0.1813

TRAINING Epoch 18/50 Loss 0.5587 Accuracy 0.1833
TRAINING Epoch 19/50 Loss 0.5528 Accuracy 0.2000
TRAINING Epoch 20/50 Loss 0.5480 Accuracy 0.2213
TRAINING Epoch 21/50 Loss 0.5437 Accuracy 0.2277
TRAINING Epoch 22/50 Loss 0.5388 Accuracy 0.2400
TRAINING Epoch 23/50 Loss 0.5333 Accuracy 0.2467
TRAINING Epoch 24/50 Loss 0.5295 Accuracy 0.2683
TRAINING Epoch 25/50 Loss 0.5228 Accuracy 0.2837
TRAINING Epoch 26/50 Loss 0.5174 Accuracy 0.2843
TRAINING Epoch 27/50 Loss 0.5120 Accuracy 0.2990
TRAINING Epoch 28/50 Loss 0.5076 Accuracy 0.3103
TRAINING Epoch 29/50 Loss 0.5063 Accuracy 0.2997
TRAINING Epoch 30/50 Loss 0.5010 Accuracy 0.3163
TRAINING Epoch 31/50 Loss 0.4957 Accuracy 0.3320
TRAINING Epoch 32/50 Loss 0.4907 Accuracy 0.3367
TRAINING Epoch 33/50 Loss 0.4867 Accuracy 0.3580
TRAINING Epoch 34/50 Loss 0.4806 Accuracy 0.3630
TRAINING Epoch 35/50 Loss 0.4780 Accuracy 0.3657
TRAINING Epoch 36/50 Loss 0.4742 Accuracy 0.3640
TRAINING Epoch 37/50 Loss 0.4701 Accuracy 0.3837
TRAINING Epoch 38/50 Loss 0.4676 Accuracy 0.3723
TRAINING Epoch 39/50 Loss 0.4637 Accuracy 0.3920
TRAINING Epoch 40/50 Loss 0.4596 Accuracy 0.3913
TRAINING Epoch 41/50 Loss 0.4548 Accuracy 0.4017
TRAINING Epoch 42/50 Loss 0.4529 Accuracy 0.3937
TRAINING Epoch 43/50 Loss 0.4476 Accuracy 0.4117
TRAINING Epoch 44/50 Loss 0.4445 Accuracy 0.4267
TRAINING Epoch 45/50 Loss 0.4385 Accuracy 0.4387
TRAINING Epoch 46/50 Loss 0.4358 Accuracy 0.4477
TRAINING Epoch 47/50 Loss 0.4359 Accuracy 0.4277
TRAINING Epoch 48/50 Loss 0.4309 Accuracy 0.4513
TRAINING Epoch 49/50 Loss 0.4285 Accuracy 0.4470
TRAINING Epoch 50/50 Loss 0.4258 Accuracy 0.4487
Finished Training

```
test(model,  criterion)
```

```
Test Loss: 0.4148 Test Accuracy 0.3406
```

This is the test output shown above.

Then I modified my hyperdata similar as part 1, and get output as below:

```
print(    .10)
```

```
TRAINING Epoch 1/20 Loss 0.1591 Accuracy 0.0417
TRAINING Epoch 2/20 Loss 0.1199 Accuracy 0.2080
TRAINING Epoch 3/20 Loss 0.0947 Accuracy 0.3543
TRAINING Epoch 4/20 Loss 0.0785 Accuracy 0.4657
TRAINING Epoch 5/20 Loss 0.0694 Accuracy 0.5200
TRAINING Epoch 6/20 Loss 0.0617 Accuracy 0.5700
TRAINING Epoch 7/20 Loss 0.0560 Accuracy 0.6090
TRAINING Epoch 8/20 Loss 0.0520 Accuracy 0.6393
TRAINING Epoch 9/20 Loss 0.0504 Accuracy 0.6413
TRAINING Epoch 10/20 Loss 0.0467 Accuracy 0.6687
TRAINING Epoch 11/20 Loss 0.0440 Accuracy 0.6957
TRAINING Epoch 12/20 Loss 0.0422 Accuracy 0.7053
TRAINING Epoch 13/20 Loss 0.0403 Accuracy 0.7103
TRAINING Epoch 14/20 Loss 0.0386 Accuracy 0.7217
TRAINING Epoch 15/20 Loss 0.0369 Accuracy 0.7330
TRAINING Epoch 16/20 Loss 0.0363 Accuracy 0.7507
TRAINING Epoch 17/20 Loss 0.0354 Accuracy 0.7490
TRAINING Epoch 18/20 Loss 0.0339 Accuracy 0.7627
TRAINING Epoch 19/20 Loss 0.0333 Accuracy 0.7680
TRAINING Epoch 20/20 Loss 0.0329 Accuracy 0.7667
Finished Training
----------
```

```
test(model,  criterion)
```

```
Test Loss: 0.0680 Test Accuracy 0.4517
```

This is output with RESNET_LAST_ONLY = True. I also get output with this var = False, shown below:

```
/usr/local/lib/python3.8/dist-packages/torch/utils/data/
  warnings.warn(_create_warning_msg(
/usr/local/lib/python3.8/dist-packages/torchvision/model
  warnings.warn(
/usr/local/lib/python3.8/dist-packages/torchvision/model
  warnings.warn(msg)
TRAINING Epoch 1/20 Loss 0.1509 Accuracy 0.0553
TRAINING Epoch 2/20 Loss 0.0976 Accuracy 0.2890
TRAINING Epoch 3/20 Loss 0.0719 Accuracy 0.4407
TRAINING Epoch 4/20 Loss 0.0624 Accuracy 0.5133
TRAINING Epoch 5/20 Loss 0.0513 Accuracy 0.6043
TRAINING Epoch 6/20 Loss 0.0433 Accuracy 0.6620
TRAINING Epoch 7/20 Loss 0.0389 Accuracy 0.6993
TRAINING Epoch 8/20 Loss 0.0348 Accuracy 0.7390
TRAINING Epoch 9/20 Loss 0.0320 Accuracy 0.7590
TRAINING Epoch 10/20 Loss 0.0300 Accuracy 0.7700
TRAINING Epoch 11/20 Loss 0.0270 Accuracy 0.8000
TRAINING Epoch 12/20 Loss 0.0270 Accuracy 0.7963
TRAINING Epoch 13/20 Loss 0.0245 Accuracy 0.8203
TRAINING Epoch 14/20 Loss 0.0230 Accuracy 0.8257
TRAINING Epoch 15/20 Loss 0.0222 Accuracy 0.8280
TRAINING Epoch 16/20 Loss 0.0213 Accuracy 0.8370
TRAINING Epoch 17/20 Loss 0.0216 Accuracy 0.8437
TRAINING Epoch 18/20 Loss 0.0201 Accuracy 0.8490
TRAINING Epoch 19/20 Loss 0.0190 Accuracy 0.8540
TRAINING Epoch 20/20 Loss 0.0190 Accuracy 0.8570
Finished Training
----------

Test  Loss:  0.0436  Test  Accuracy  0.5680
```

Hyper params I used at the best condition are (EPOCHS = 20, LEARNING_RATE = 0.01, BATCH_SIZE = 32, RESNET_LAST_ONLY = False).