Vedant Jain 301438620

CMPT 412 Project 2(Using 1 late day for this submission)

PART 1:

I have submitted the .CSV file on Kaggle under the name: Vedant Ashish Jain with the best accuracy of 62.20%

Illustration of my final network:

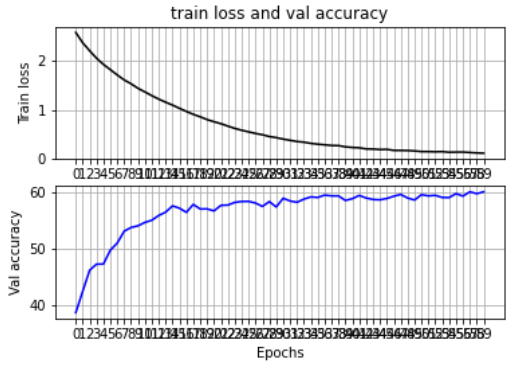
Layer no.	Layer type	Kernel size (for conv layers)	Input | Output dimension	Input | Output Channels (conv)	Stride
1	Conv2d	3	32 | 32	3 | 64	1
2	Batchnorm2d	-	32 | 32	-	-
3	Relu	-	32 | 32	-	-
4	Conv2d	3	32 | 32	64 | 64	1
5	Batchnorm2d	-	32 | 32	-	-
6	Relu	-	32 | 32	-	-
7	Maxpool2d	2	32 | 16	-	2
8	Conv2d	3	16 | 16	64 | 128	1
9	Batchnorm2d	-	16 | 16	-	-
10	Relu	-	16 | 16	-	-
11	Conv2d	3	16 | 16	128 | 128	1
12	Batchnorm2d	-	16 | 16	-	-
13	Relu	-	16 | 16	-	-
14	Maxpool2d	2	16 | 8	-	2
15	Conv2d	3	8 | 8	128 | 256	1
16	Batchnorm2d	-	8 | 8	-	-
17	Relu	-	8 | 8	-	-
18	Conv2d	3	8 | 8	256 | 256	1
19	Batchnorm2d	-	8 | 8	-	-
20	Relu	-	8 | 8	-	-
21	Maxpool2d	2	8 | 4	-	2
22	Conv2d	3	4 | 4	256 | 512	1
23	Batchnorm2d	-	4 | 4	-	-
24	Relu	-	4 | 4	-	-
25	Conv2d	3	4 | 4	512 | 512	1
36	Linear	-	2048 | 4096	-	-
37	Batchnorm1d	-	4096 | 4096	-	-
38	Relu	-	4096 | 4096	-	-
39	Linear	-	4096 | 2048	-	-
40	Batchnorm1d	-	2048 | 2048	-	-
41	Relu	-	2048 | 2048	-	-
42	Linear	-	2048 | 1024	-	-
43	Batchnorm1d	-	1024 | 1024	-	-
44	Relu	-	1024 | 1024	-	-
45	Linear	-	1024 | 100	-	-

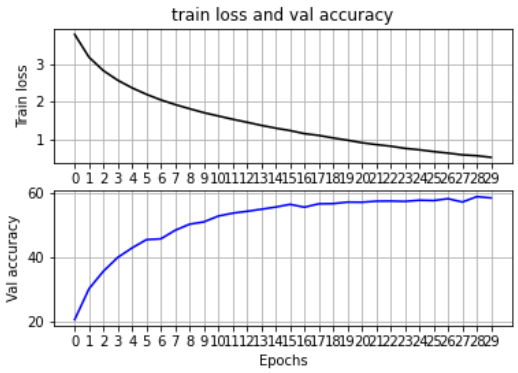
Final network illustration from the code and loss and accuracy from the training:

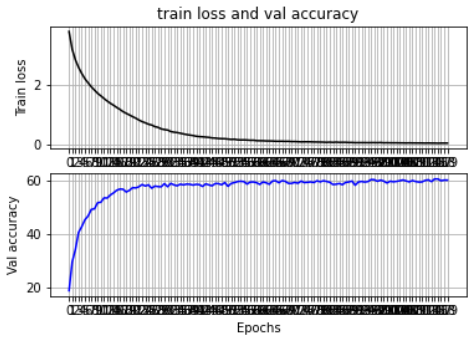
```
BaseNet(
  (conv1): Conv2d(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (batchn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (rl1): ReLU()
  (conv2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (batchn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (rl2): ReLU()
  (pool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv3): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (batchn3): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (rl3): ReLU()
  (conv4): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (batchn4): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (rl4): ReLU()
  (pool2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv5): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (batchn5): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (rl5): ReLU()
  (conv6): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (batchn6): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (rl6): ReLU()
  (pool3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv7): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (batchn7): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (rl7): ReLU()
  (conv8): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (batchn8): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (rl8): ReLU()
  (pool4): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (fc_net): Sequential(
    (0): Linear(in_features=2048, out_features=4096, bias=True)
    (1): BatchNorm1d(4096, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU()
    (3): Linear(in_features=4096, out_features=2048, bias=True)
    (4): BatchNorm1d(2048, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (5): ReLU()
    (6): Linear(in_features=2048, out_features=1024, bias=True)
    (7): BatchNorm1d(1024, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (8): ReLU()
    (9): Linear(in_features=1024, out_features=100, bias=True)
  )
)
```

```
[48] loss: 0.172
Accuracy of the network on the val images: 59 %
[49] loss: 0.170
Accuracy of the network on the val images: 59 %
[50] loss: 0.161
Accuracy of the network on the val images: 58 %
[51] loss: 0.148
Accuracy of the network on the val images: 59 %
[52] loss: 0.148
Accuracy of the network on the val images: 59 %
[53] loss: 0.144
Accuracy of the network on the val images: 59 %
[54] loss: 0.148
Accuracy of the network on the val images: 59 %
[55] loss: 0.135
Accuracy of the network on the val images: 59 %
[56] loss: 0.138
Accuracy of the network on the val images: 59 %
[57] loss: 0.139
Accuracy of the network on the val images: 59 %
[58] loss: 0.130
Accuracy of the network on the val images: 60 %
[59] loss: 0.121
Accuracy of the network on the val images: 59 %
[60] loss: 0.115
Accuracy of the network on the val images: 60 %
Finished Training
```

My networks performance and loss(60, 20 and 120 epochs in order):







Ablation study:

The base network that was provided was not very deep and would only provide an accuracy of about 20%, after adding more layers to the network and making the network deeper I was able to get the network to give results that were about 62% accurate. I added more layers to the network to make the network better. I changed the standard deviation of the dataset to 1. The extra layers added to

the network made it better for the network to learn. I tested the network with several different epochs, I tested the network with epochs set to 10, 30, 40, 60, 80 and 120 I realized that for my network the performance would start to plateau or even drop a little hence I chose to stick with 60 epochs. I also added some normalization layers and linear layers to increase the potential of the network.

PART 2:

Report the train and test accuracy achieved by using the ResNet as a fixed feature extractor vs. fine-tuning the whole network.

For the accuracy of ResNet on a fixed fixture is 68% on training set and 40% on testing dataset.

After I was done fine tuning the network with hyper parameters the accuracy returned by the network was 88.67% on training set and 59.02% on the testing set.

```
TRAINING Epoch 46/50 Loss 0.0120 Accuracy 0.8693
TRAINING Epoch 47/50 Loss 0.0114 Accuracy 0.8737
TRAINING Epoch 48/50 Loss 0.0112 Accuracy 0.8803
TRAINING Epoch 49/50 Loss 0.0110 Accuracy 0.8880
TRAINING Epoch 50/50 Loss 0.0107 Accuracy 0.8867
Finished Training
----------

Test Loss: 0.0259 Test Accuracy 0.5902
```

Report any hyperparameter settings you used (batch_size, learning_rate, resnet_last_only, num_epochs).

The number of epochs was increased to 50, the learning rate was brought up to 0.001 and the batch_size was increased to 64.

Visualizing