CMPT 412 -COMPUTER VISION

PROJECT-2 REPORT (NAVJOT KAUR: 301404765)

Kaggle Submission under Navjott77

I am claiming my 1 late day for this assignment.

PART 1: Improved BaseNet on CIFAR100

1. **Include a table illustrating your final network and describe what it is.**

   Table is as follows:

Layer No.	Layer Type	Kernel size (for conv layers)	Input | Output dimension	Input | Output Channels (for conv layers)
1	Conv2d	3	32|32	3|9
2	BatchNorm2d	-	32|32	-
3	ReLu	-	32|32	-
4	Conv2d	3	32|32	9|9
5	BatchNorm2d	-	32|32	-
6	ReLu	-	32|32	-
7	Conv2d	3	32|32	9|27
8	BatchNorm2d	-	32|32	-
9	ReLu	-	32|32	-
10	MaxPool2d	2	32|16	-
11	Conv2d	3	16|16	27|243
12	BatchNorm2d	-	16|16	-
13	ReLu	-	16|16	-
14	Conv2d	3	16|16	243|729
15	BatchNorm2d	-	16|16	-
16	ReLu	-	16|16	-
17	Conv2d	3	16|16	729|729
18	BatchNorm2d	-	16|16	-
19	ReLu	-	16|16	-
20	MaxPool2d	2	16|8	-
21	linear	-	46656|5184	-
22	ReLu	-	5184|5184	-
23	linear	-	5184|100	-

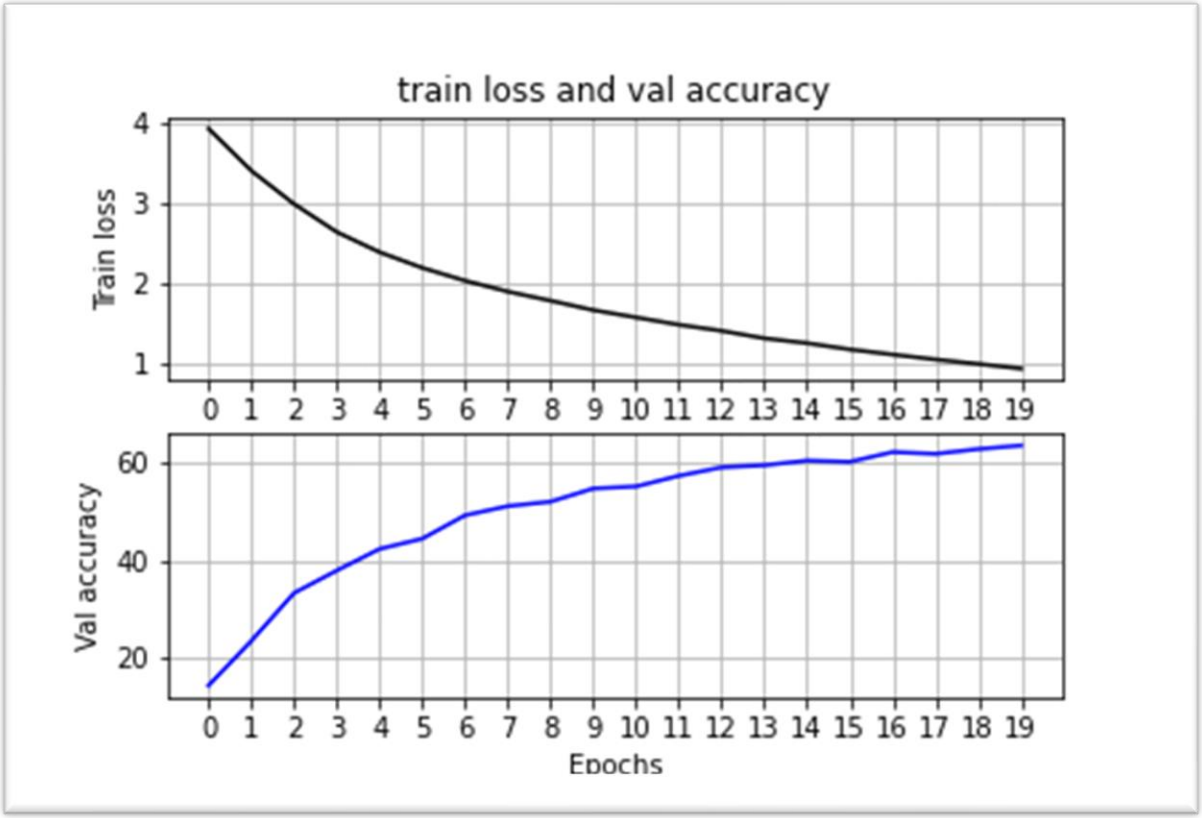
As asked in the question I used the below given functions to make the data well-conditioned for improved training.

```
transforms.RandomCrop(32, padding=4, padding_mode = 'reflect'),
transforms.RandomHorizontalFlip(),
transforms.ToTensor(),
transforms.Normalize((0),(1))
```

Next the epochs have been updated to 20 along with adding 6 convolutional layers to attain better accuracy of the model.

2. Include a plot.png from Collab notebook, illustrating the training loss and the validation accuracy.

The plot.png file is as follows:



3. **Include at least one ablation study, reporting the performance improvement**.

I believe the performance improvement is because of adding more convolutional layers to the network and adding normalization layer (nn.BatchNorm1d) after ReLu. The first accuracy after applying all the changes was 56% and after trial and error i.e, using different values for layers and changing epochs increased the accuracy percentage to 63% (which is greater than 50%) with a loss of 0.944.

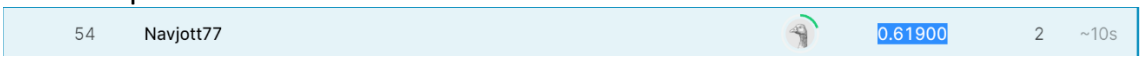
4. **Base performance**

Below is a screenshot of my google collab including the loss and accuracy percentage.

```
[141] [1] loss: 3.924
      Accuracy of the network on the val images: 14 %
      [2] loss: 3.401
      Accuracy of the network on the val images: 23 %
      [3] loss: 2.988
      Accuracy of the network on the val images: 33 %
      [4] loss: 2.640
      Accuracy of the network on the val images: 37 %
      [5] loss: 2.390
      Accuracy of the network on the val images: 42 %
      [6] loss: 2.195
      Accuracy of the network on the val images: 44 %
      [7] loss: 2.036
      Accuracy of the network on the val images: 49 %
      [8] loss: 1.901
      Accuracy of the network on the val images: 51 %
      [9] loss: 1.788
      Accuracy of the network on the val images: 52 %
      [10] loss: 1.670
      Accuracy of the network on the val images: 54 %
      [11] loss: 1.580
      Accuracy of the network on the val images: 55 %
      [12] loss: 1.489
      Accuracy of the network on the val images: 57 %
      [13] loss: 1.413
      Accuracy of the network on the val images: 59 %
      [14] loss: 1.320
      Accuracy of the network on the val images: 59 %
      [15] loss: 1.260
      Accuracy of the network on the val images: 60 %
      [16] loss: 1.182
      Accuracy of the network on the val images: 60 %
      [17] loss: 1.119
      Accuracy of the network on the val images: 62 %
      [18] loss: 1.058
      Accuracy of the network on the val images: 62 %
      [19] loss: 1.002
      Accuracy of the network on the val images: 63 %
      [20] loss: 0.944
      Accuracy of the network on the val images: 63 %
      Finished Training
```

5. **Relative performance**

I uploaded a csv file to the Kaggle which reports the accuracy of 0.61900. Below is a screenshot provided.



Part 2: Transfer Learning

The vary Hyperparams used for this part are:

```
NUM_EPOCHS = 35
LEARNING_RATE = 0.001
BATCH_SIZE = 18
RESNET_LAST_ONLY = False
```

And the data is transformed via below functions:

```
data_transforms = {
    'train': transforms.Compose([
        transforms.Resize(256),
        # transforms.CenterCrop(224),
        #TODO: Transforms.RandomResizedCrop() instead of CenterCrop(), Ran
domRoate() and Horizontal Flip()
        transforms.RandomResizedCrop(224),
        transforms.RandomRotation(40),
        transforms.RandomHorizontalFlip(),
        transforms.ToTensor(),
        transforms.Normalize((0),(1)),
        #TODO: Transforms.Normalize()
    ]),
    'test': transforms.Compose([
        transforms.Resize(256),
        transforms.CenterCrop(224),
        transforms.ToTensor(),
        transforms.Normalize((0),(1)),
        #TODO: Transforms.Normalize()
    ]),
}
```

After applying all these changes, I got an accuracy of 81.83%. which is more than 80% as required. (See figure)

```
  cpuset_checked))
TRAINING Epoch 1/35 Loss 0.2963 Accuracy 0.0117
TRAINING Epoch 2/35 Loss 0.2675 Accuracy 0.0687
TRAINING Epoch 3/35 Loss 0.2331 Accuracy 0.1520
TRAINING Epoch 4/35 Loss 0.2073 Accuracy 0.2333
TRAINING Epoch 5/35 Loss 0.1877 Accuracy 0.3043
TRAINING Epoch 6/35 Loss 0.1690 Accuracy 0.3793
TRAINING Epoch 7/35 Loss 0.1559 Accuracy 0.4243
TRAINING Epoch 8/35 Loss 0.1448 Accuracy 0.4677
TRAINING Epoch 9/35 Loss 0.1347 Accuracy 0.4987
TRAINING Epoch 10/35 Loss 0.1264 Accuracy 0.5370
TRAINING Epoch 11/35 Loss 0.1182 Accuracy 0.5607
TRAINING Epoch 12/35 Loss 0.1112 Accuracy 0.5930
TRAINING Epoch 13/35 Loss 0.1060 Accuracy 0.6010
TRAINING Epoch 14/35 Loss 0.1019 Accuracy 0.6110
TRAINING Epoch 15/35 Loss 0.0967 Accuracy 0.6470
TRAINING Epoch 16/35 Loss 0.0916 Accuracy 0.6570
TRAINING Epoch 17/35 Loss 0.0886 Accuracy 0.6760
TRAINING Epoch 18/35 Loss 0.0849 Accuracy 0.6847
TRAINING Epoch 19/35 Loss 0.0797 Accuracy 0.7027
TRAINING Epoch 20/35 Loss 0.0762 Accuracy 0.7243
TRAINING Epoch 21/35 Loss 0.0747 Accuracy 0.7320
TRAINING Epoch 22/35 Loss 0.0720 Accuracy 0.7267
TRAINING Epoch 23/35 Loss 0.0686 Accuracy 0.7483
TRAINING Epoch 24/35 Loss 0.0670 Accuracy 0.7547
TRAINING Epoch 25/35 Loss 0.0634 Accuracy 0.7673
TRAINING Epoch 26/35 Loss 0.0617 Accuracy 0.7763
TRAINING Epoch 27/35 Loss 0.0610 Accuracy 0.7767
TRAINING Epoch 28/35 Loss 0.0578 Accuracy 0.7897
TRAINING Epoch 29/35 Loss 0.0578 Accuracy 0.7853
TRAINING Epoch 30/35 Loss 0.0570 Accuracy 0.7880
TRAINING Epoch 31/35 Loss 0.0536 Accuracy 0.8053
TRAINING Epoch 32/35 Loss 0.0558 Accuracy 0.7953
TRAINING Epoch 33/35 Loss 0.0519 Accuracy 0.8107
TRAINING Epoch 34/35 Loss 0.0522 Accuracy 0.8100
TRAINING Epoch 35/35 Loss 0.0489 Accuracy 0.8183
Finished Training
----------
```

Below us a figure of test accuracy which is 55.69%