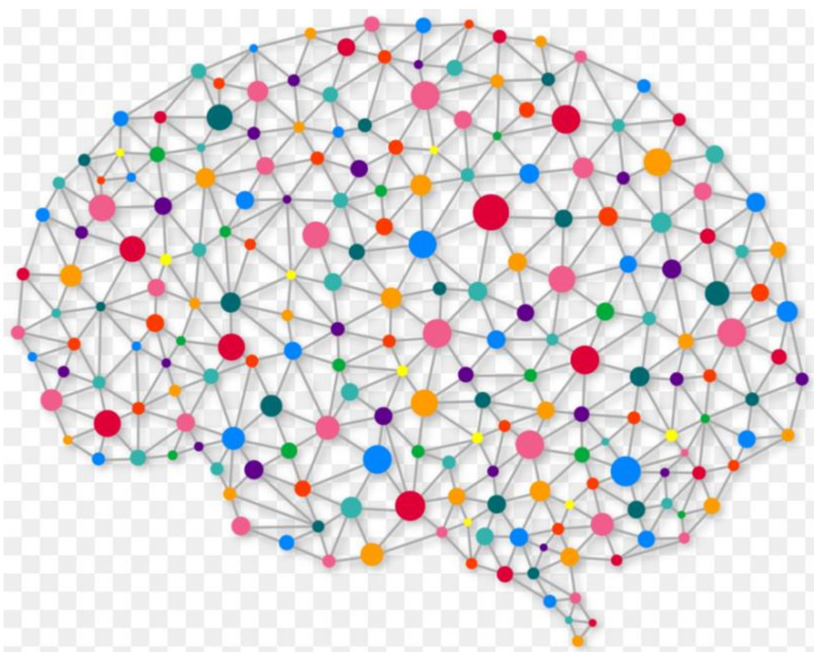
Project 6

Deep learning by PyTorch

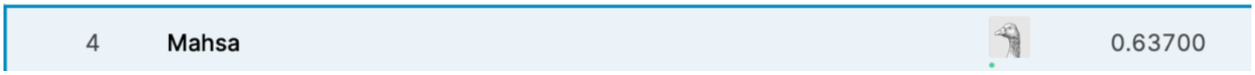
Computer Vision - CMPT 762



Part 1: Improving BaseNet on CIFAR100

Name in Kaggle: Mahsa (Mahsa Maleki)

Best Accuracy: 0.63700



Factors which helped improve your model performance:

- Data normalization

  I've normalized train and test set by computing mean and standard deviation. This can help the training to be more robust.

- Data augmentation

  I've augmented data by Random Crop and Random Horizontal Flip which improve the training.

- Deeper network

  I've added more layers here. But we should also take care of overfitting. We should not put a Max Pool layer after every conv layer in our deeper network as it leads to too much loss of information.

- Normalization layers

  I've added Batch Norm layer here after each convolution and linear layer. It normalizes the activations of the previous layer and help reduce overfitting.

- Number of epochs

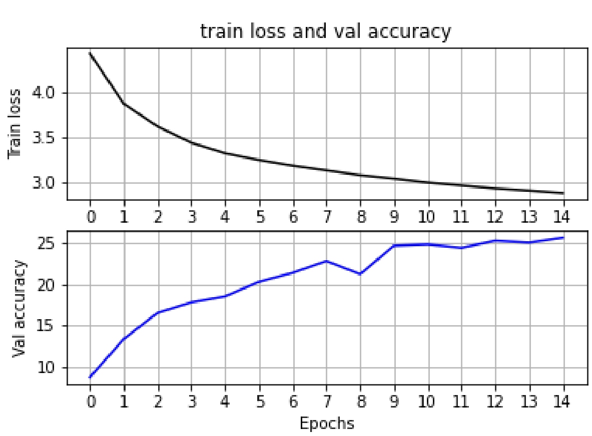
  This number should be chose based on early stopping technique. We should see after how many epochs the network doesn't learn anymore.

Here you can see the network architecture:

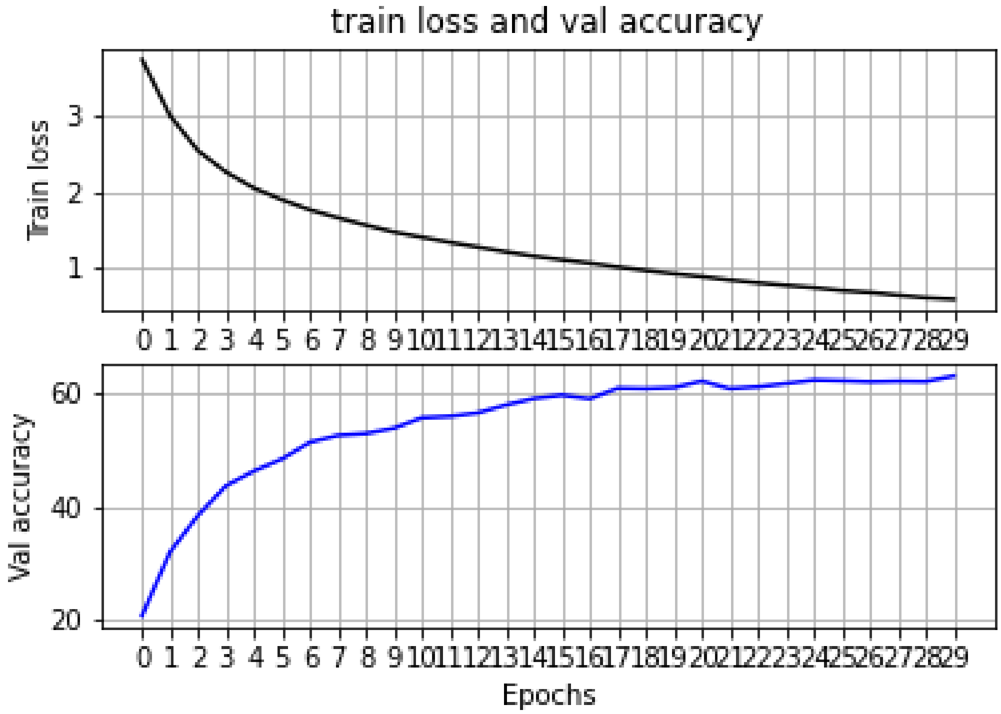
Layer No.	Layer Type	Kernel size (for conv layers)	Input | Output dimension	Input | Output Channels (for conv layers)
1	conv2d	3	32 | 32	3 | 64
2	BatchNorm2d	-	32 | 32	
3	ReLU	-	32 | 32	
4	conv2d	3	32 | 32	64 | 64
5	BatchNorm2d	-	32 | 32	
6	ReLU	-	32 | 32	
7	MaxPool2d	2	32 | 16	

8	conv2d	3	16 | 16	64 | 128
9	BatchNorm2d	-	16 | 16	
10	ReLU	-	16 | 16	
11	conv2d	3	16 | 16	128 | 128
12	BatchNorm2d	-	16 | 16	
13	ReLU	-	16 | 16	
14	MaxPool2d	2	16 | 8	
15	conv2d	3	8 | 8	128 | 256
16	BatchNorm2d	-	8 | 8	
17	ReLU	-	8 | 8	
18	conv2d	3	8 | 8	256 | 256
19	BatchNorm2d	-	8 | 8	
20	ReLU	-	8 | 8	
21	MaxPool2d	2	8 | 8	
22	conv2d	3	8 | 4	256 | 256
23	BatchNorm2d	-	4 | 4	
24	ReLU	-	4 | 4	
25	conv2d	3	4 | 4	256 | 512
26	BatchNorm2d	-	4 | 4	
27	ReLU	-	4 | 4	
28	MaxPool2d	2	4 | 4	
29	conv2d	3	4 | 2	512 | 512
30	BatchNorm2d	-	2 | 2	
31	ReLU	-	2 | 2	
32	conv2d	3	2 | 2	512 | 512
33	BatchNorm2d	-	2 | 2	
34	ReLU	-	2 | 2	
35	MaxPool2d	2	2 | 2	
36	Linear	-	512 | 256	
37	BatchNorm1d	-	256 | 256	
38	ReLU	-	256 | 256	
39	Linear	-	256 | 100	

Here is the plot.png for BaseNet:



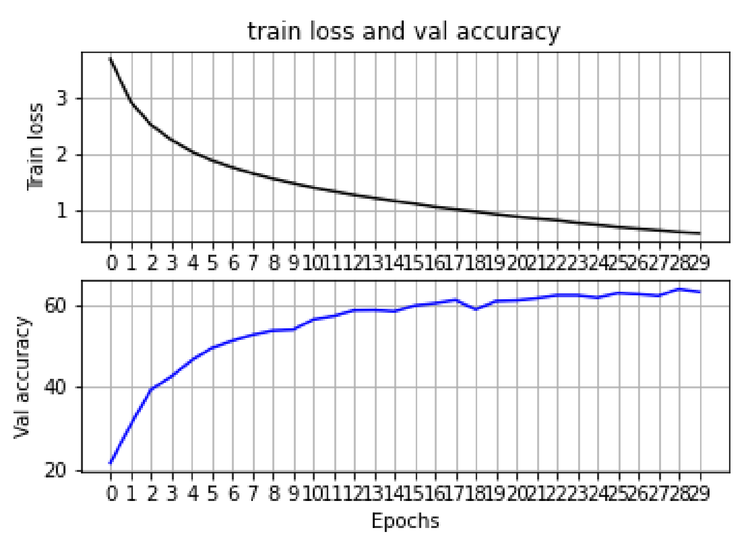
Final architecture's plot for training loss and validation accuracy:



At least one ablation study to validate the above choices, i.e., a comparison of performance for two variants of a model, one with and one without a certain feature or implementation choice:

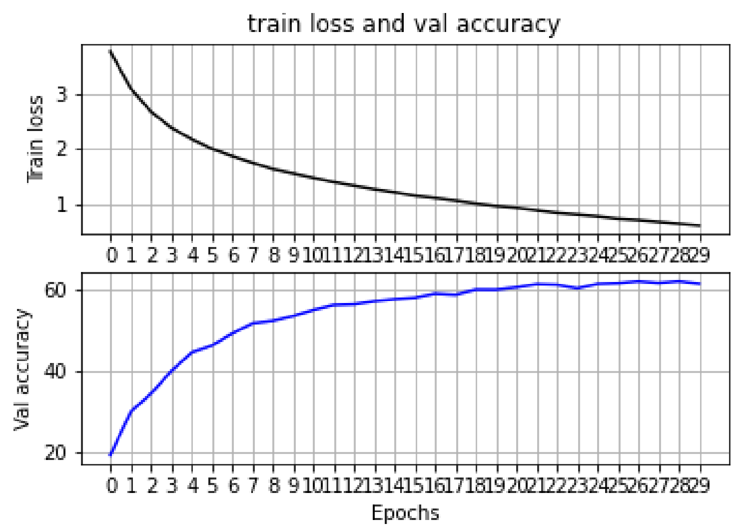
First) If we remove data normalization, we will get this result:

Score on Kaggle: 0.61400



Second) If we don't have batch norm after every conv layer: (remove layers 5, 12, 19, 26, 33)

Score on Kaggle: 0.60800



Part 2: Transfer Learning

I've tried to change hyperparameters and adding augmentation and regularization (weight_dacay) for better generalization.

Here you can see hyperparameters and differences that we have in comparison to the original code:

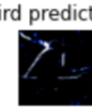
```
NUM_EPOCHS = 25
LEARNING_RATE = 0.001
BATCH_SIZE = 16

optimizer = optim.SGD(model.parameters(), lr=LEARNING_RATE, momentum=0.9,
weight_decay=5e-4)


data_transforms = {
    'train': transforms.Compose([
        transforms.Resize(256),
        transforms.RandomResizedCrop(224),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
    ]),
```

```
    'test': transforms.Compose([
        transforms.Resize(256),
        transforms.CenterCrop(224),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
    ]),
}
```

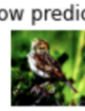
Here you can see train and test accuracy for ResNet as a fixed feature extractor and fine-tuning the ResNet.

RESNET_LAST_ONLY	Train Accuracy	Test Accuracy
True	0.6437	0.4375
False	0.8653	0.5760

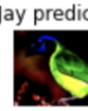
class: 078.Gray_Kingbird predicted: 007.Parakeet_Auklet



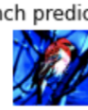
class: 113.Baird_Sparrow predicted: 113.Baird_Sparrow



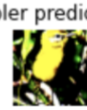
class: 075.Green_Jay predicted: 075.Green_Jay



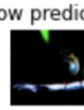
class: 035.Purple_Finch predicted: 035.Purple_Finch



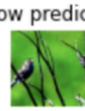
class: 176.Prairie_Warbler predicted: 182.Yellow_Warbler



class: 125.Lincoln_Sparrow predicted: 125.Lincoln_Sparrow



class: 127.Savannah_Sparrow predicted: 127.Savannah_Sparrow



class: 084.Red_legged_Kittiwake predicted: 062.Herring_Gull