Deep Learning by PyTorch

CMPT 412

Aditya Rajvanshi – 301349754

Part 1: Improving BaseNet on CIFAR100

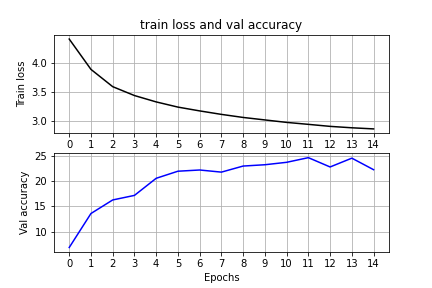
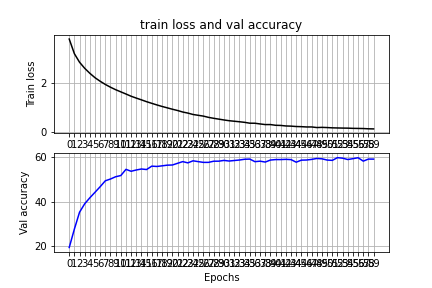
The name used for Kaggle is Aditya Rajvanshi, with an accuracy of 58.60%

Network Architecture

I used the VGG-19 architecture identified in the following article:  
<https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035>

The table illustration is:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer no. | Layer type | Kernel size  (for conv layers) | Input | Output dimension | Input | Output Channels (conv) |
| 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 | 4 | - |
| 22 | Conv2d | 3 | 4 | 4 | 256 | 512 |
| 23 | Batchnorm2d | - | 4 | 4 | - |
| 24 | Relu | - | 4 | 4 | - |
| 25 | Conv2d | 3 | 4 | 4 | 512 | 512 |
| 26 | Batchnorm2d | - | 4 | 4 | - |
| 27 | Relu | - | 4 | 4 | - |
| 28 | Maxpool2d | 2 | 4 | 2 | - |
| 29 | Conv2d | 3 | 2 | 2 | 512 | 512 |
| 20 | 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 | 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 | - |

Comparison of base performance vs. Deeper net improved performance

On the left, we see the base performance, while the right side shows the deeper net. A list of improvements is provided below:

1. The data was normalized with a zero mean and a standard deviation of one. This made the training easier and more robust.
2. The data was augmented with transformations provided in the following article:

* <https://pytorch.org/docs/stable/torchvision/transforms.html>

Where colour jitter was applied along with random rotations, random horizontal flips, and random crop in entirely random order. This helps in generating more and varied training data which can increase accuracy.

1. The network was made deeper by having many more convolutional layers and fully conncected layers. The VGG-19 architecture was followed roughly
2. Normalization layers were added to reduce overfitting and improve the training of the model. BatchNorm2d were added after conv layers and BatchNorm1d after ReLU layers.
3. Training was stopped after 60 epochs since there were no improvements beyond this point.

Ablation study

Initially, the network was achieving accuracies of roughly 20%-22%. However, after making the changes listed above, the accuracy increased to roughly 58%-59%.

Part 2: Transfer Learning

The accuracy of the fixed feature extractor was approximately 63% on the training data and 45% on the test data.

The accuracy with fine tuning the network was approximately 89% on the training data and approximately 57% on the test data.

Screenshots below:





Hyperparameters

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. A dropout was added. The resnet\_last\_only was set to false so that the entire network could be tweaked.