Supervised ResNet

Choosing Model

I tried both ResNet as well as a vanilla model and expectedly ResNet gave a better result. The ResNet was framed and trained from scratch without any pretrained weights. Since it used residual layers, it helped reducing overfitting. It pushed val_accuracy from 65% to over 70% and val_loss to a minimum of 0.87 compared to a loss of above 1 for base.

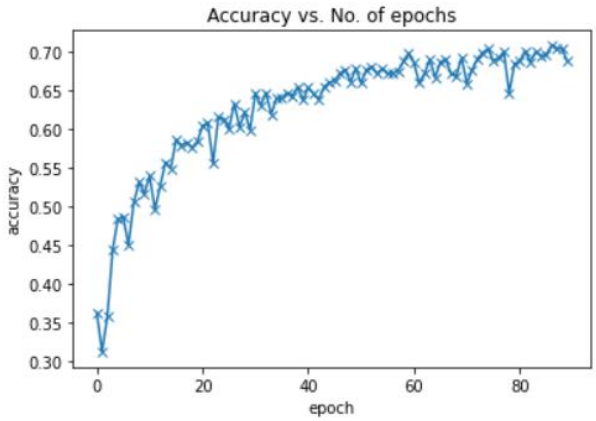
```
ResNet(
  (conv_1): Sequential(
    (0): Conv2d(3, 50, kernel_size=(3, 3), stride=(1, 1), bias=False)
    (1): BatchNorm2d(50, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU()
  )
  (pool_1): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (res_1): Sequential(
    (0): Sequential(
      (0): Conv2d(50, 50, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (1): BatchNorm2d(50, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (2): ReLU()
    )
    (1): Sequential(
      (0): Conv2d(50, 50, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (1): BatchNorm2d(50, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (2): ReLU()
    )
  )
  (conv_2): Sequential(
    (0): Conv2d(50, 100, kernel_size=(3, 3), stride=(1, 1), bias=False)
    (1): BatchNorm2d(100, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU()
  )
  (pool_2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (flatten): Flatten(start_dim=1, end_dim=-1)
  (dropout_1): Dropout(p=0.3, inplace=False)
  (dense_1): Linear(in_features=48400, out_features=512, bias=True)
  (dropout_2): Dropout(p=0.3, inplace=False)
  (dense_2): Linear(in_features=512, out_features=32, bias=True)
  (dropout_3): Dropout(p=0.3, inplace=False)
  (out): Linear(in_features=32, out_features=10, bias=True)
)
```

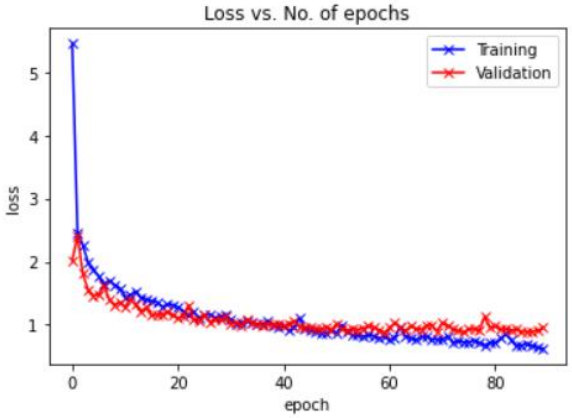
Data Augmentation

Augmenting the data by adding transformations like rotation, flip and colour jitter helped reduce overfitting. Dropout layers also helped boost val_accuracy

Choosing Epochs

I trained the model over a range of epochs (from 60 to 150). And for the chosen learning rate the model gave the best performance at around 90 epochs i.e. high accuracy without overfitting


Accuracy vs. No. of epochs
accuracy
epoch
0.70
0.65
0.60
0.55
0.50
0.45
0.40
0.35
0.30
0
20
40
60
80


Loss vs. No. of epochs
Training
Validation
loss
epoch
5
4
3
2
1
0
20
40
60
80

Important metrics for evaluation

Focussing only on test accuracy can lead to bad evaluation of biased datasets. For example in cancer detection, if all cases are declared as negative then the algorithm will have a high accuracy but it will be a very bad algorithm for the required purpose.

Thus, recall is a very important metric. It is defined as the fraction of samples from a class which are correctly predicted by the model.

F1 score combines these two metrics by finding the harmonic mean of the two and is a very widely used deciding factor.

Other than metrics which evaluate directly how well the model performs its objective. Given two models, which achieve similar F1 scores, the model which is less complex i.e., takes up less time to fit and uses up less memory space will be preferred to a heavier model