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.

```
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)
)
```

	Validation Loss	Validation Accuracy
Vanilla Model	0.9479	66.49%
ResNet Model	0.8840	70.57%

Data Augmentation

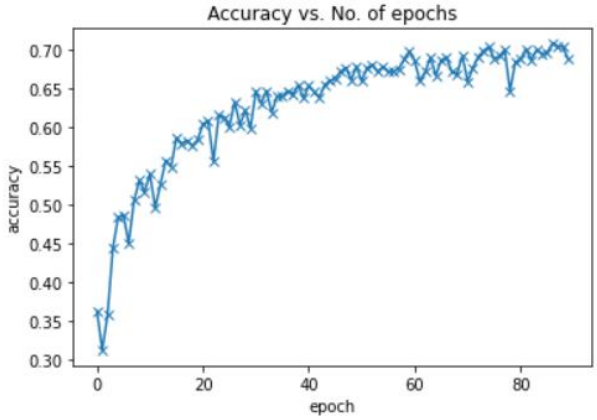
Augmenting the data by adding transformations helped reduce overfitting. The transformations were: -

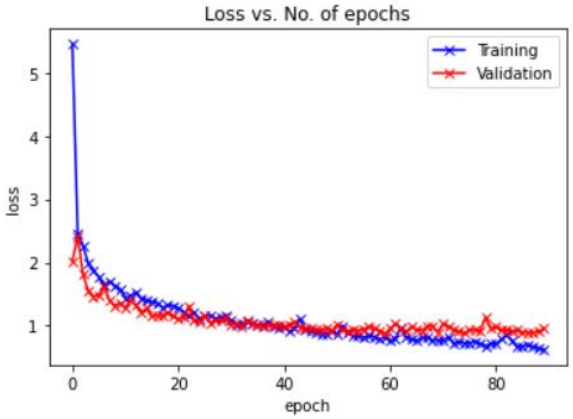
- Random Horizontal Flip
- Random Rotation with a range of 10°
- Colour Jitter – Brightness and Contrast with a range of 0.2, Hue and Saturation with a range of 0.1

Dropout layers also helped boost val_accuracy.

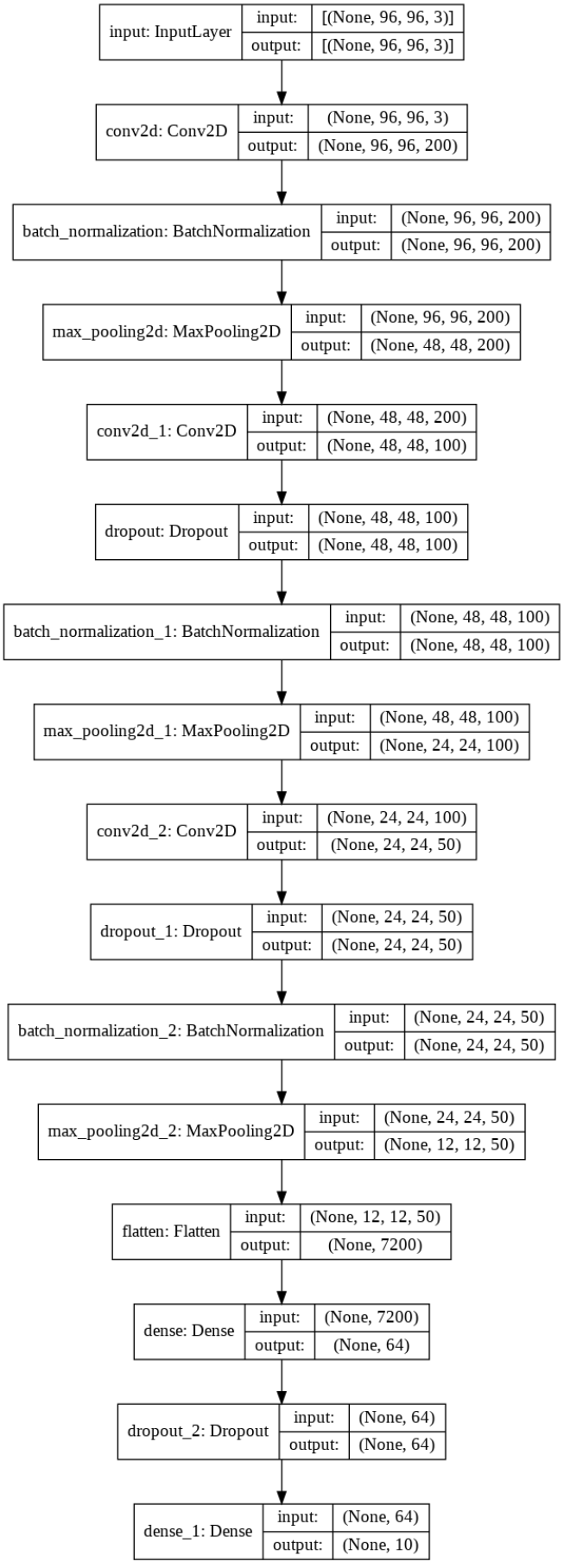
Choosing Epochs

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





Semi Supervised Learning

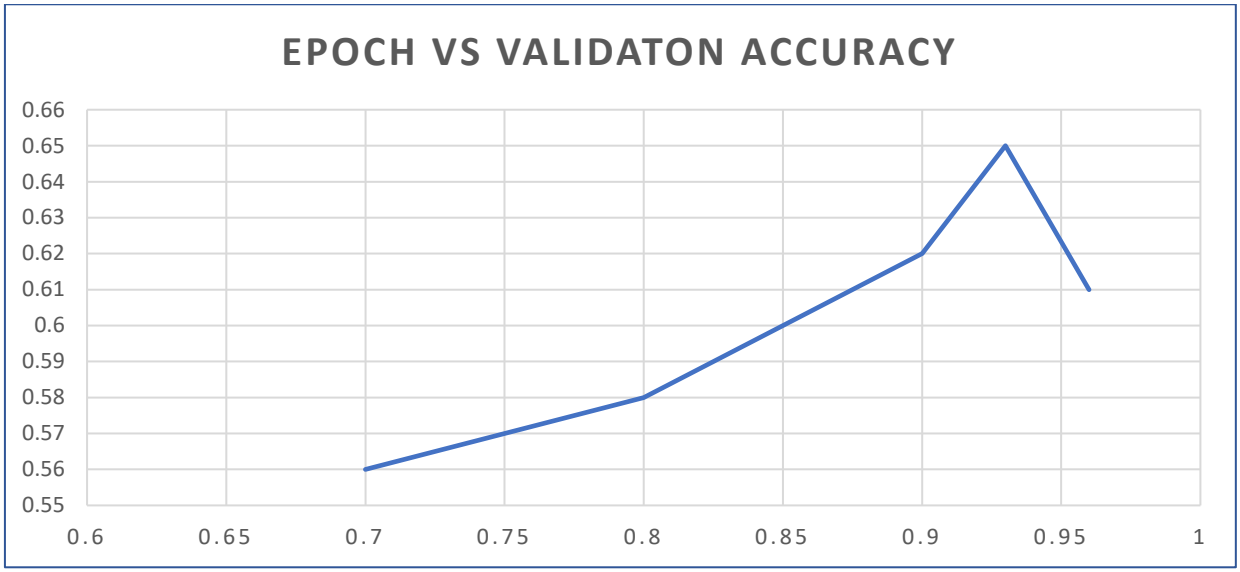


Choosing Model

Though ResNet gave a higher test accuracy on while running supervised learning, CNN gave a better final test accuracy when we carried out semi-supervised using pseudo labelling. The ResNet model overfit on the dataset extremely fast.

Choosing Threshold

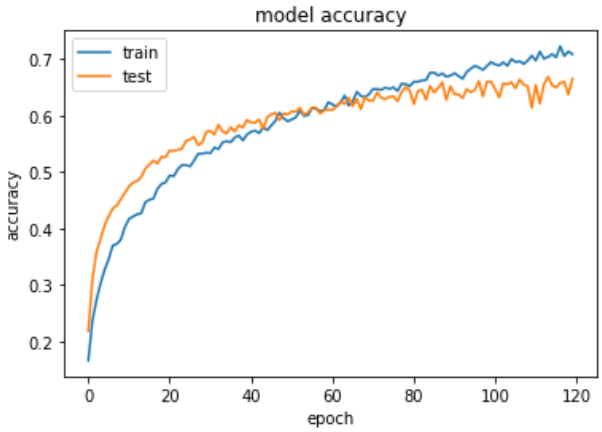
I tried multiple threshold's ranging from 0.7 all the way to 0.96, the best result on the test accuracy was given by 0.93. This can be justified by the fact that a threshold of 0.93 would produce pseudo labels on a fairly large number of images from the unlabelled data set while maintaining a high level of accuracy. (13,925 to be exact)

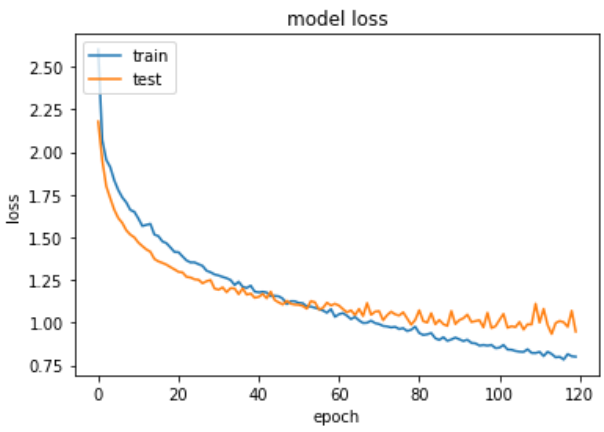


Choosing Epochs

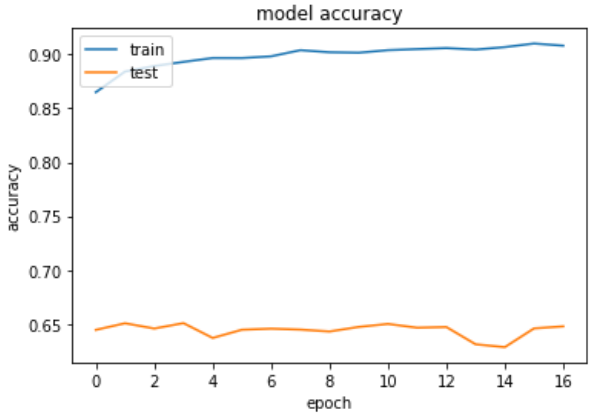
Rather than choosing the number of epochs manually I decided to utilise the earlyCallback method in .fit() function of Keras. I set the monitor to be val_loss and ended the training if the val_loss stopped decreasing in value.

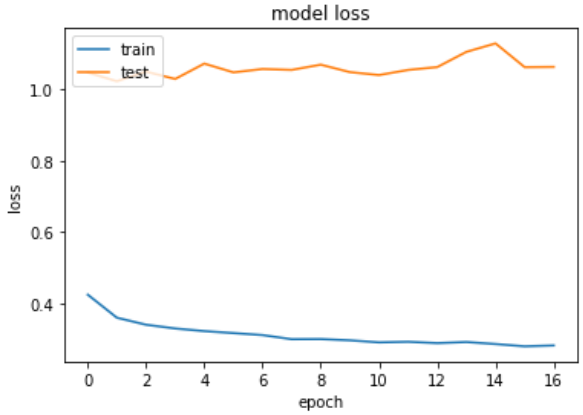
- Training the initial supervised classifier: -





- Training the model further on the new dataset: -





Overfitting

One drawback of the semi supervised model is that it's overfitting. If we put the test data as the validation set for the fitting process it is even more clear. When we train the model on the train dataset, we get a val_accuracy of approximately 0.66 and a val_loss of about 0.95. However, when we add pseudo labels to the untrained dataset and train the model on that, accuracy falls to 0.65 and val_loss increases to 1.02. Though this shift is minor, train_loss continues to decrease along with an increase in train_accuracy thus showcasing overfitting

Conclusion

After running a semi-supervised model on both CNN and ResNet it is evident that the model is clearly overfitting and thus a supervised learning based on ResNet is the optimum solution for the given dataset getting us an val_accuracy of over 0.7 and a val_loss of 0.87

Self-Supervised SimCLR

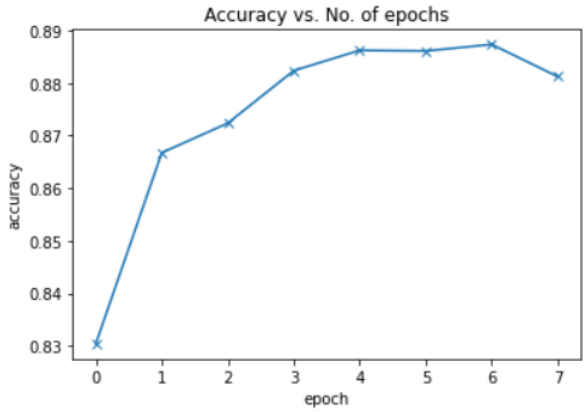
Selecting Model

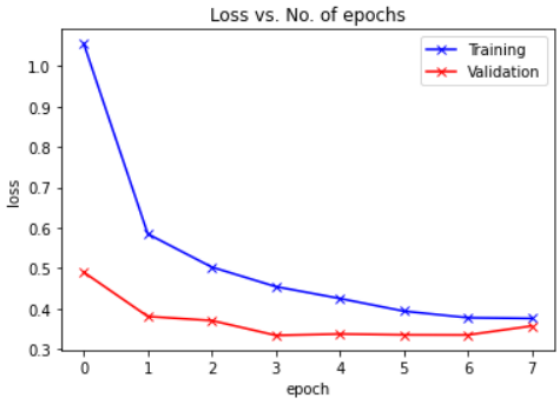
Since training contrastive loss requires heavy computing, (two image augmentations have to be processed simultaneously and two backpropagations must be carried out after that) and cannot be done on a single GPU stream, I imported SimCLR embeddings. As classifier I framed a 3 layered fully connected neural network and trained it by inputting the embeddings obtained after passing the images through the ResNet50 model.

```
Classifier(
  (dropout_1): Dropout(p=0.12, inplace=False)
  (dense_1): Linear(in_features=2048, out_features=512, bias=True)
  (dropout_2): Dropout(p=0.27, inplace=False)
  (dense_2): Linear(in_features=512, out_features=64, bias=True)
  (dropout_3): Dropout(p=0.2, inplace=False)
  (out): Linear(in_features=64, out_features=10, bias=True)
)
```

Training the model

Since we only need to train a simple classifier, overfitting isn't much of an issue and can be easily avoided using image augmentations and dropout layers. Putting a dropout layer before the first Dense layer helps prevent overdependence on any of the 2048 features in the output of the ResNet50 model. Furthermore only 8 epochs are enough to obtain an optimum accuracy and minimum loss without overfitting on the train set.





Questions to be answered

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

Auto Augmentation

Previous models on all datasets have almost exclusively used manual image augmentation which takes experience on data manipulation.

In SimCLR we have to apply two data augmentations on each image and calculate the loss based on the embeddings. Thus, choosing the right augmentation is going to be very important, choosing two augmentations that are very similar will lead to no learning as the embeddings generated will be inherently similar. At the same time, we do not want augmentations which will lead to starkly different embeddings as it will lead to a very vague and generalised final result. Therefore, in SimCLR if implemented properly auto augmentation can help increase the accuracy by a significant margin.

For datasets like MNIST or OCR detection purposes, image augmentations like image ratio manipulation, stretching and compressing will be very useful while colour shifts wont affect the efficiency much

On the other hand, image datasets which are used to classify objects and animals like ImageNet, stretching and compression should be restricted to a very small extent while augmentation should focus more on colour shifts and random cropping. Colour shift will prevent the algorithm from focussing on a given colour while cropping will prevent the algorithm from focussing on a specific feature of a given object/animal.

By training an automated image augmentation, we can make the algorithm itself determine which augmentation will help create the greatest increasing in the accuracy.

Augmentation will depend not only on the image subject but also image composition. Darkening an already dark image will lead to featureless images which will lead to no learning. This is an avenue where automated generator will give an advantage over manual augmentation.