1 Foundations of Convolutional Neural Networks

Note: All y-axis are accuracy/loss and x-axis are epochs in graphs.

Part 1 -

On MNIST Dataset -

Learning Curves -

Training Accuracies vs Iterations

Losses vs Iterations

Testing Accuracy - 0.988

F-Scores -

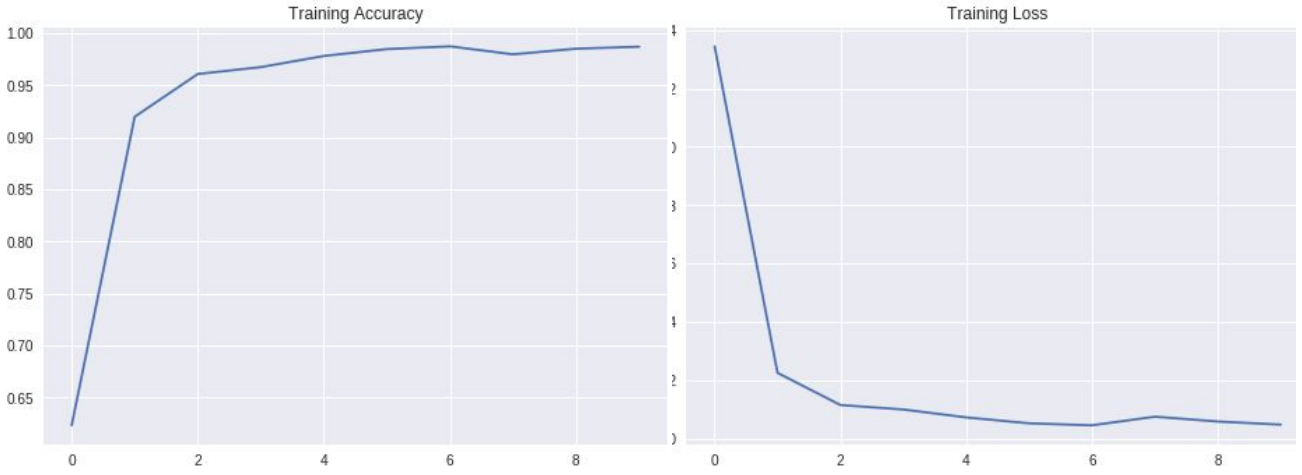
```
[0.98534613 0.9915892  0.98368522 0.98201264 0.98565574 0.97997775
 0.98527865 0.98778701 0.97451898 0.97348674]
```

Confusion Matrices -

```
[[ 978    0    0    0    0    0    0    1    1    0]
 [   0 1133    0    1    0    0    0    0    0    1]
 [   2    0 1007    4    2    0    0   14    2    1]
 [   0    0    0 1001    0    2    0    2    4    1]
 [   1    1    1    0  967    0    4    0    0    8]
 [   2    0    0   10    0  876    1    0    2    1]
 [   4    2    1    0    2    3  945    0    1    0]
 [   0    2    3    1    0    0    0 1021    0    1]
 [   4    1    2    1    1    0    0    2  961    2]
 [   1    2    0    3    5    1    0    3    3  991]]
```

Line Dataset -

Learning Curves -



Training Accuracy vs Iterations

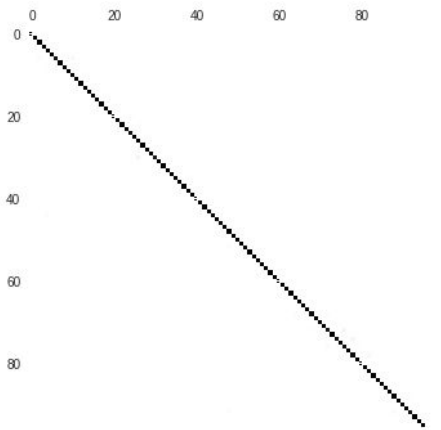
Loss vs Iteration

Testing Accuracy - `0.990625`

F-Scores -

```
[0.99003322 0.99328859 0.98039216 1.         0.97402597 1.
 0.95172414 1.         0.99328859 1.         0.97402597 1.
 0.99667774 1.         1.         1.         0.99003322 1.
 0.99665552 1.         0.97278912 0.99337748 0.98648649 1.
 0.96598639 0.9771987  0.94936709 1.         0.99665552 1.
 0.95833333 1.         1.         1.         1.         1.
 0.99665552 1.         1.         1.         0.99665552 0.99667774
 0.98996656 0.99667774 0.97972973 0.98684211 0.97278912 1.
 0.97068404 0.99667774 1.         1.         0.98305085 1.
 0.99667774 1.         1.         1.         0.99665552 1.
 1.         1.         1.         1.         0.98349835 1.
 0.99333333 1.         0.96621622 1.         0.99003322 1.
 0.92255892 1.         0.96193772 1.         0.93729373 1.
 0.94389439 1.         0.98684211 0.99665552 0.9771987  1.
 1.         1.         1.         1.         0.97689769 1.
 1.         1.         0.96245734 1.         1.         1.        ]
```

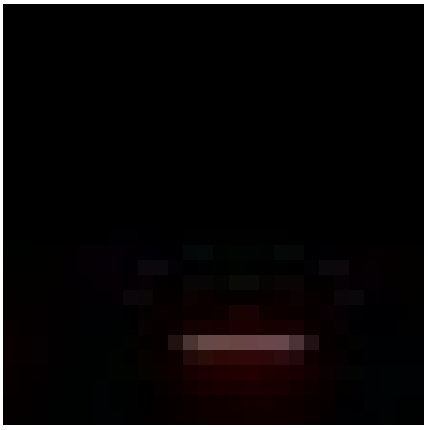
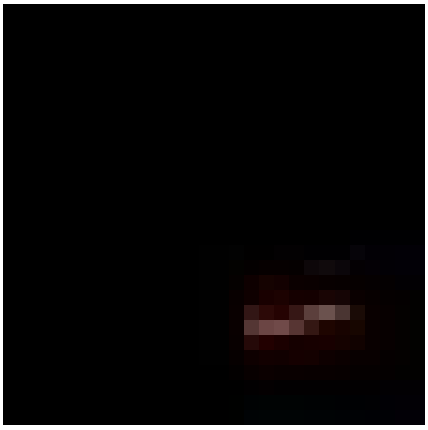
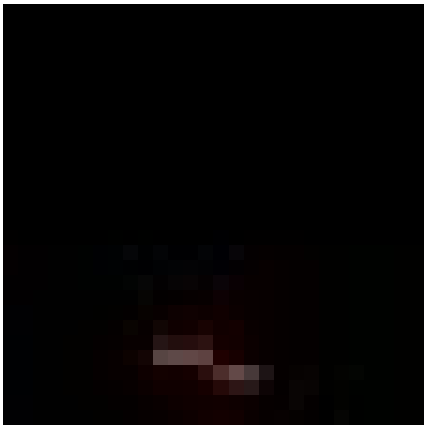
Confusion Matrix -



Inferences -

The model trains better over the line data as compared to the mnist data. The mnist dataset is known to have few errors in ground truth because of which some accuracy is dropped. In the mnist data, the model makes mistakes in classification of 3 and 5 and of 2 and 7. This is expected as many humans write these two digits almost similarly.

The network finds it difficult to distinguish between similar classes like short length with same width but 15 degrees apart as in JPG images the pixels are not much separated and are actually like two shorter lines. The network however is easily able to distinguish between two different colours and doesn't make an error in it. There is also has no confusion in cases of same length and different width which it was unable to distinguish between in the previous assignment. Network gets confused in such distinct classes -

Part 2 -

MNIST Dataset -

Variation 1 -

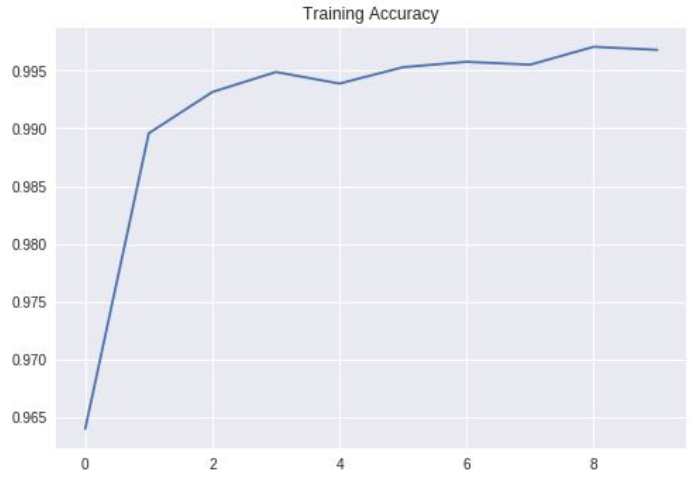
Change in filter size to 3x3 and increase number of trainable conv layers making a deeper network.

Testing Accuracy - `0.9878`

Learning Curves -

Variation 2 -
Add dropout layers with 0.25 to drop.

Testing accuracy - `0.9896`



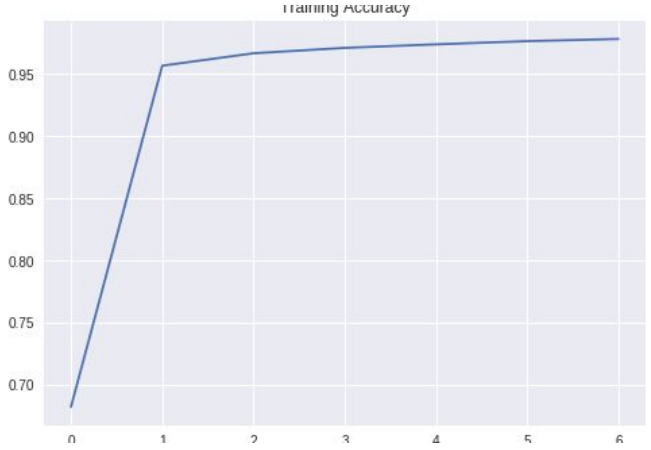
Variation 3 -
Experiment with the final FC Layer. Change number of neurons in it from 1024 to 128. This increased training speed by a lot with not much difference in accuracy.
Testing accuracy - `0.9891`

Variation 4 -
Experimentation with the optimizer parameters. Changed learning rate to 0.0005 and added beta_1 to 0.95 (default is 0.9). This improved performance and crossed the 99% in testing accuracy. However for convergences needed a couple of more epochs.
Testing accuracy - `0.9901`

Variation 5 -

Change max pooling layers to strided convolutions with strides of 2. No change to the network dimensions but the accuracy is a bit down.

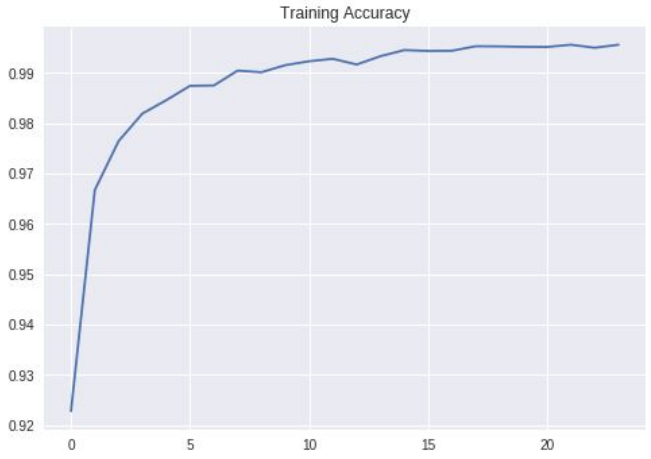
Testing accuracy - `0.986`



Variation 6 -

In this variations we are using residual nets. Two residual nets are connected as in the model of ResNet and then connected finally with 2 FC layers. The dimension of the conv layers used in the residual nets are also reduced to 6. The training is super fast as number of parameters are very less.

Testing accuracy - `0.988`



Inference -

The models exceed 99% on testing mnist data with the best performance from variance 4 of experimenting the hyperparameter of the loss function. Here, we are using more Conv layers, dropouts, and hyperparameterization to refine our model and increase accuracy. We can see that dropping some data randomly with Dropout layer helps in improving the accuracy of predictions. In the model with residual networks the training takes very fast due to less parameters and sees no drop in accuracy due to that. We can infer that to further increase the accuray of the models we have to introduce data augmentation and noise the training set to further generalize the model and achieve testing accuracy close to a 100%.

Line Dataset -

Variation 1 -

Change network structure with two Conv layers and two FC at the end with dimensions 512 and 128. Also both the FC layers are followed by Dropout layers with 0.3 dropout rates.

Testing accuracy - `0.9799`

Variation 2 -

In this variations we are using residual nets. Two residual nets are connected as in the model of ResNet and then connected finally with 2 FC layers. The dimension of the conv layers used in the residual nets are also reduced to 6. The training is super fast as number of parameters are very less.

Testing accuracy - `0.99763888888`

Variation 3 -

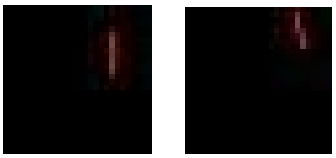
Experimentation with the optimizer parameters like epsilon, clipnorm and decay. The learning rate with high start and decay after each epochs help converge and brings down loss early on also fast.

Testing accuracy - `0.999027777`

Inference -

The model is able to achieve 99.9% accuracy in testing data of the line dataset. This shows that the dataset is a fairly simple and can be easily classified by using CNNs. The original network on part1 performs better than a deeper network in this dataset probably because of overfitting due in the complex polynomial of the deeper network. Being a simple dataset, better performance on a simpler and shallow network is expected. Introducing residual nets for this dataset really surges the performance very less parameters. This is in coherence with the inference from the first shallow network as the data generalizes very well with models with less parameters. In variation 3 starting with a high learning and decay after each epochs help converge better at the end and brings down loss early on also fast. The few

wrong classification that occur are among two consecutive similar angles for the case of 0 and 15 degree and for 105 and 90 degrees for both colors.



Mis-classification is observed between two classes of such type due to splitting of only 7 pixel in two parts for short lines.