CSA 250 : Deep Learning Project 2
Report

Abin Bassam A
21 February 2020

Python Version :3.6.9
Tensorflow Version :2.1.0

Dataset: Fashion-MNIST consists of consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes. Class labels are as follows.

Label	0	1	2	3	4	5	6	7	8	9
Class	T-shirt	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Boot

Here our aim is to implement 2 classifiers based on Deep Neural Network (DNN) and Convolutional Neural Networks (CNN) and compare/evaluate their performance . So for DNN we will be flattening out the image before passing it to the neural networks whereas for CNN we will be passing it as it is for the convolutional layer.

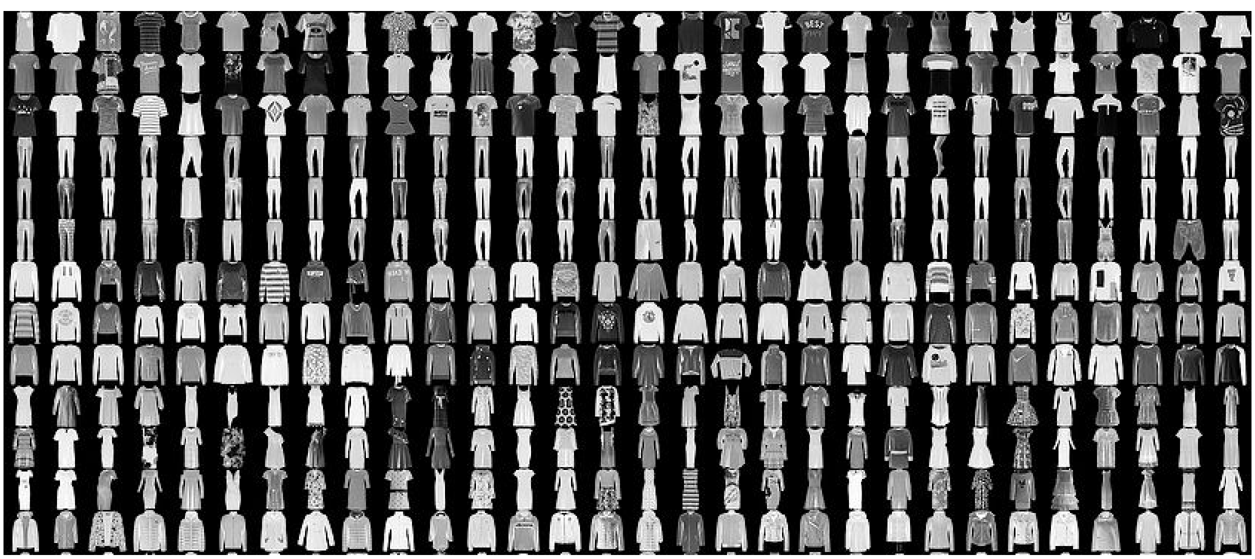


Fig 1. Sample Images in Fashion Mnist Data Set

Deep Neural Network Classifier

Architecture Design: Here we started with a single hidden layer and softmax output layer for the classifier. We used '**relu**' activation function**. Adam'** was our choice of optimiser and '**Cross entropy'** for loss function.
The results are as follows:

No. of Units (in single hidden layer)	**Training Accuracy**	**Testing Accuracy**
64	0.9325	0.8828
128	0.9522	0.8858
256	0.9723	0.8933

The results above are after 100 epochs. We can see clearly that increasing units increases testing accuracy but it also overfits the training data.

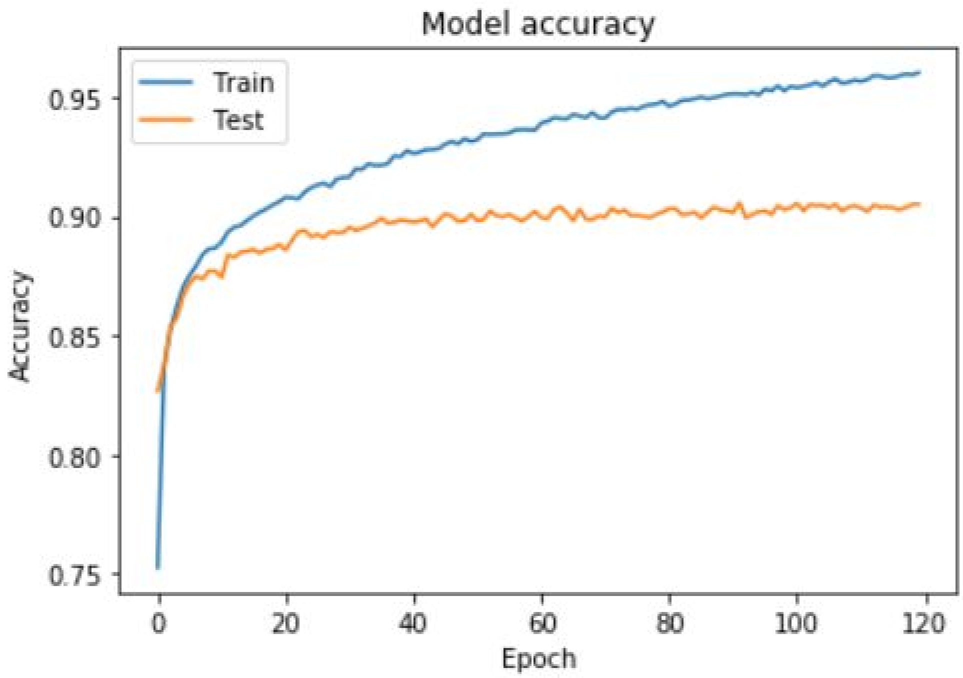
As a natural progression we move onto networks with 2 hidden layers with varying units.

No. of Units (first layer,second layer)	**Training Accuracy**	**Testing Accuracy**
(128,128)	0.9810	0.8878
(128,64)	0.9713	0.8850
(64,32)	0.9446	0.8790

Here we can see a clear case of overfitting. As we have training accuracy close 97% but our testing accuracy is less than that of our single layer network.
To counter the above problem we use Dropout Layer. Since increasing hidden layers didn't increase testing accuracy we will stick to single layer network.

No. of Units (in single hidden layer)	**Dropout (%)**	**Training Accuracy**	**Testing Accuracy**
64	20	0.9212	0.8869
256	50	0.9400	0.8997
1024	**50**	**0.9581**	**0.9080**

Final Model (in ***red****)*
Previously we observed that increasing units increase training accuracy with overfitting but adding a dropout layer increases the testing error by limiting overfitting to training data.
So we finalised on a model with **1024 units** in the hidden layer and a Dropout of **(0.5).** Training vs test accuracy plotted with No.of Epochs.



	precision	recall	f1-score	support
top	0.83	0.87	0.85	1000
trouser	0.98	0.98	0.98	1000
pullover	0.83	0.80	0.82	1000
dress	0.91	0.90	0.91	1000
coat	0.81	0.85	0.83	1000
sandal	0.98	0.97	0.97	1000
shirt	0.75	0.71	0.73	1000
sneaker	0.95	0.96	0.96	1000
bag	0.97	0.98	0.98	1000
ankle boot	0.96	0.97	0.96	1000
accuracy			0.90	10000
macro avg	0.90	0.90	0.90	10000
weighted avg	0.90	0.90	0.90	10000

Class Wise Accuracy

Convolutional Neural Network Classifier

Architecture Design: Similar to DNN classifier we started from Single layer Network (Convolutional layer + Fully Connected Layer) and work up from there.

Network Properties:
Filter Size: 3*3
Max pooling : 2*2
Activation Function : **Relu**
Optimiser: **Adam**
Loss Function: **Cross Entropy Loss**
No. of Epochs: 30

No. of Filter (in Convolutional Layer)	Fully Connected Layer(Units)	Training Accuracy	Testing Accuracy
32	128	0.9589	0.9171
64	128	0.9678	0.9173
128	128	0.9779	0.9209

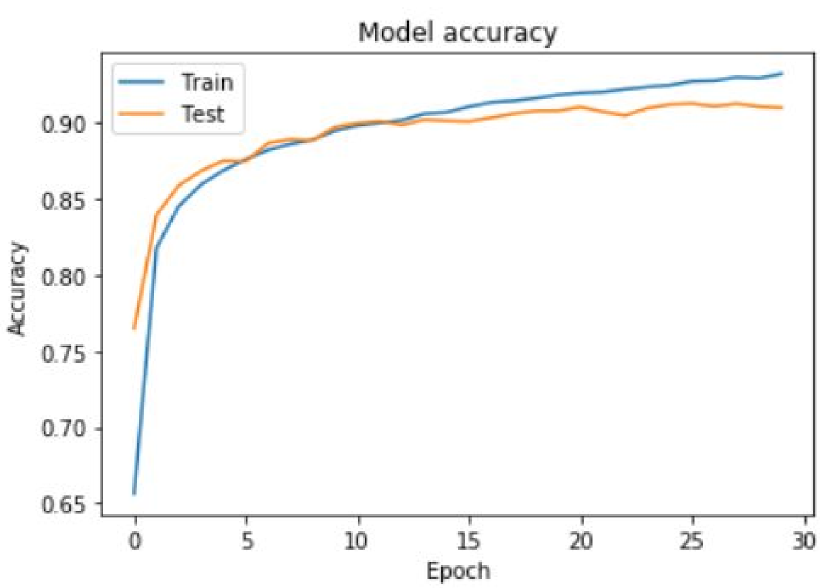
Even with the single layer CNN we outperformed our DNN in testing accuracy. But as we can see increase No.of filters results in overfitting.
So we limit No.of filters in the first layers and add second layer presuming to learn generalized features of the set. (rather than overfitting). The results are as follows:

No. of Filter (in Convolutional Layer)	**Fully Connected Layer(Units)**	**Training Accuracy**	**Testing Accuracy**
(32,32)	128	0.9961	0.9183

Here our Network overfits the data. So similar to DNN we add a dropout layer of (0.5)

No. of Filter (in Convolutional Layer)	**Fully Connected Layer(Units)**	**Training Accuracy**	**Testing Accuracy**
(32,32)	**128**	**0.9726**	**0.9226**
(64,32)	128	0.9202	0.9141
(64,64)	128	0.9399	0.9156

**Final model*




```
             precision    recall  f1-score   support

top               0.85      0.88      0.86      1000
```