Submitted by, Amal Roy - 14989

All the python code is in src directory. Python 3.6 was used. To run the code, execute:

```
python main.py --activation 'relu' --filter-config '[8 32 32]' --dataset Fashion-MNIST
--train-data ../data/Fashion-MNIST/ --test-data ../data/Fashion-MNIST
```

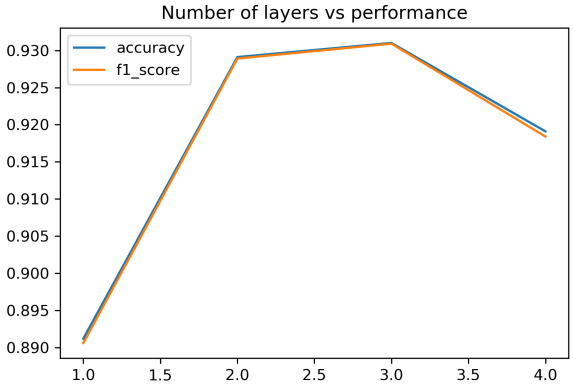
I implemented a CNN using keras. A summary architecture using filter config [8,32,32] is shown below:

layer	output shape	#param
conv2d	(None, 28, 28, 8)	136
batchnorm	(None, 28, 28, 8)	32
conv2d	(None, 28, 28, 32)	2336
batchnorm	(None, 28, 28, 32)	128
maxpool	(None, 14, 14, 32)	0
conv2d	(None, 14, 14, 32)	9248
batchnorm	(None, 14, 14, 32)	128
maxpool	(None, 7, 7, 32)	0
dropout	(None, 7, 7, 32)	0
flatten	(None, 1568)	0
dense	(None,512)	803328
dropout	(None,512)	0
dense	(None,256)	131328
dropout	(None,256)	0
dense	(None,10)	2570

Task 1

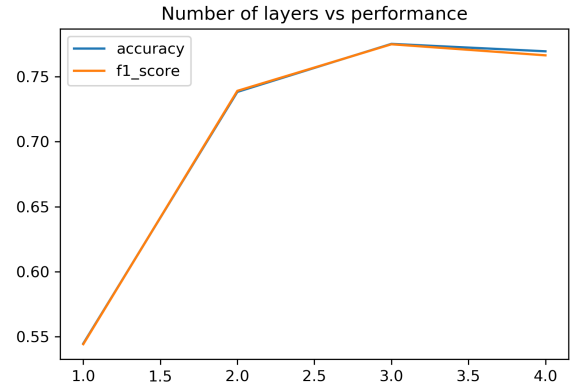
Fashion-MNIST

For different number of convolutional layers, the accuracy and f1-macro scores are plotted. When only one convolutional layer is used, the accuracy and f1 score are compariitively low. When two or more convolutional layers are added, the accuracy increases. No significant increase is observed if the number of hidden layers is increased above 3.



CIFAR-10

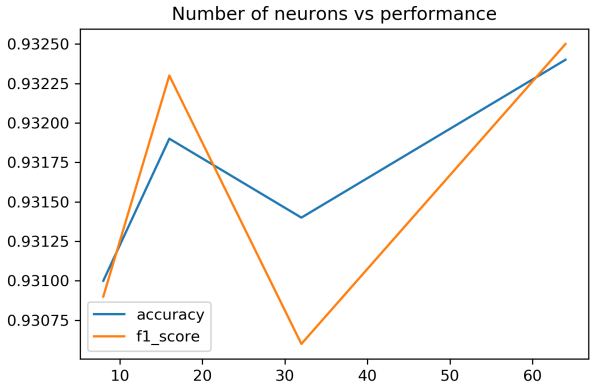
For CIFAR-10, if only one layer is used, the accuracy is significantly lower. Adding hidden layers above 3 is not showing performance gains similar to the fashion-MNIST dataset.



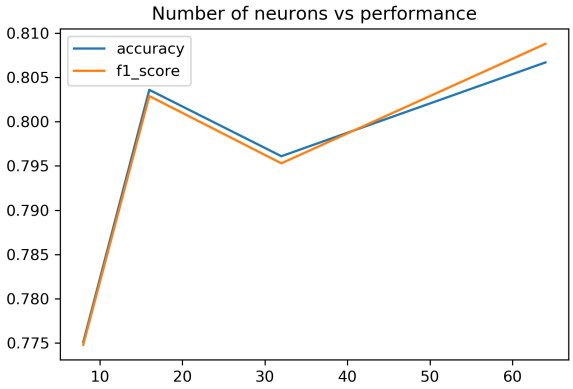
Task 2

Fixing the number of convolutional layers as 3, the number of filters in each layer was tried as: [8,32,32], [16,64,64], [32,64,64], [64,128,128]. The plots of accuracy and f1 score are given below.

Fashion-MNIST



CIFAR-10

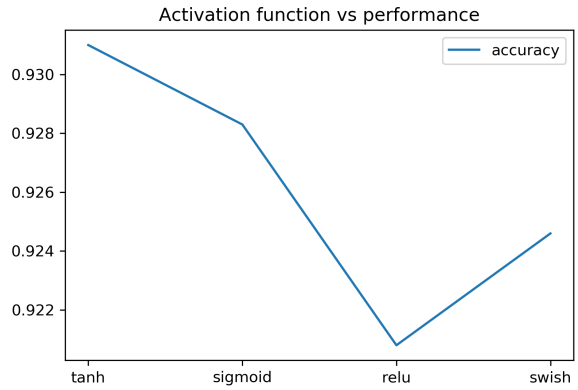


In both cases, there is an increase in accuracy if we increase the number of neurons. This will also increase the number of parameters to be learned significantly.

Task 3

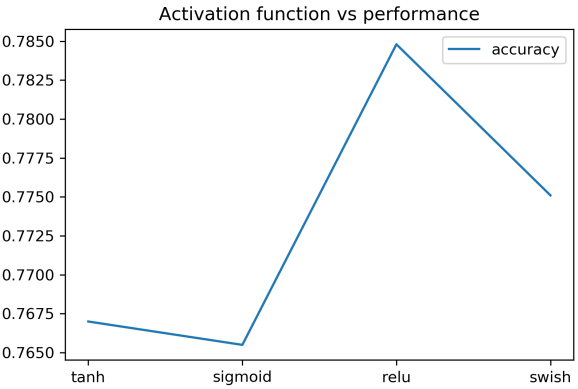
Fashion-MNIST

tanh activation function performed the best in this dataset giving an accuracy of 93.09% on the test set.



CIFAR-10

relu activation performed best in this dataset giving 78.48% accuracy.



Task 4

Keras by default uses Xavier's initialization which works well in practice. If we initialize the weights with zeros, the network doesn't converge. Other initializations available in keras also gives similar accuracy scores.

Task 5

The embedding for each data point was taken as the output of the penultimate dense layer. The implementation can be found in CNN jupyter notebook in src folder. The embedding dimension was fixed at 32 for both the datasets. The clustering was done on this 32 dimensional space using K-means clustering algorithm. Training was done on 50% of the train data for generating the embedding.

Fashion-MNIST

The clustering accuracy was found to be 91.86% and f1 macro score 91.83%. This is very close to the CNN accuracy of 91.74% and f1 macro score 91.71%. So, the embedded vector represents the original data well and a clustering on the embedded space gives a marginally better result.

CIFAR-10

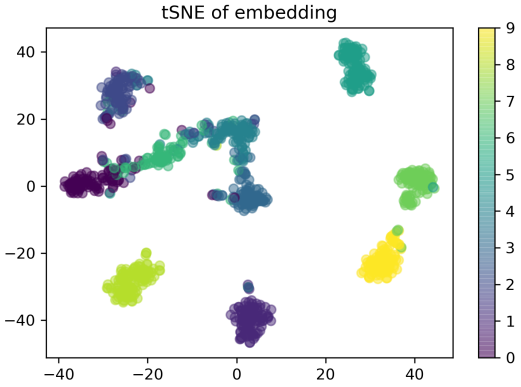
The clustering accuracy was found to be 72.21% and f1 macro score 72.12%. The CNN accuracy was 72.26% and f1 macro score 72.23%. Similar to Fashion MNIST, the semi supervised clustering approach gives a marginally better accuracy.

Task 6

The embedded data is plotted in 2D using tSNE method.

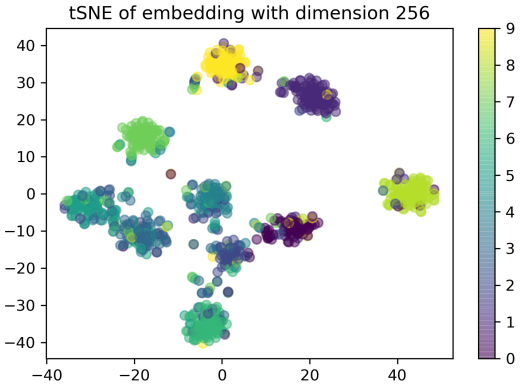
Fashion-MNIST

We can see that many of the clusters are well seperated with a small overlap. But, we can see that there are 4 clusters which lie close together.



CIFAR-10

We can identify the seperation from this dataset also. There is significant overlap between some classes because the classification accuracy of CNN is also lower.



Task 7

Fashion-MNIST

Using a MLP implemented in Keras with two hidden layers of 512 neurons each, an accuracy of 89.57% and f1 macro score of 0.8952 was obtained.

CIFAR-10

Using a MLP implemented in Keras with two hidden layers of 512 neurons each, an accuracy of 51.04% and f1 macro score of 0.503 was obtained.