Report for Assignment-2

Introduction:

Convolutional Neural Networks(CNN) have been incredibly successful in practical applications for image related tasks. For this assignment we had to implement a CNN model for the data set Fashion MNIST. Training dataset consisted of 60000 images and the testing dataset consisted of 10000 images. Each image was of the size 28 * 28 and thus consisted of 784 pixels. The given problem is a multi classification problem in which each of the input images can be a T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag or Ankle boot. We used frameworks like Keras and Tensorflow to aid in the implementation of our models. Following is the summary about the architecture and the performance of our CNN model.

Architecture and Performance of the model:

Architecture:

The most common form of a CNN architecture stacks a few Convolution layers, which is then followed by Maxpool layers, and this pattern is repeated until the given input image has been merged spatially to a small size. After this we transition to fully-connected layers and then use this for classifying the input as one of the output labels. In our model also we have made a similar architecture.

Initially there is a Conv2D layer having 32 filters and the kernel size is 3 * 3. Before we put the next Conv2D layer we perform batch normalization. The next Conv2D layer has again 32 filters and kernel size of 3 * 3. As mentioned before we again perform batch normalization. Next we put our first MaxPool2D layer having pool size of 2 * 2 and follow it with a dropout layer having a dropout rate of 0.2. This is used so as to counter overfitting. Going ahead again there is a Conv2D layer having kernel size 3 * 3 but this time it has 64 filters which is followed by performing of batch normalization and then through a MaxPool2D layer having pool size of 2 * 2 and at last followed by a dropout layer having a

dropout rate of 0.2. Now we stop using the Convolution and Maxpool layers and first flatten the output we get from the previous dropout layer. It is followed by a hidden layer having 32 nodes and activation function as ReLU. We also add a L2 regularizer term in this hidden layer. At last we have the output layer consisting of 10 nodes and having Softmax as activation function as this is a multi classification problem. We also add a L2 regularizer term in this output layer. The L2 regularizer term in the last two layers helps us to counter overfitting. Each of the above Conv2D layers have ReLU as their activation function.

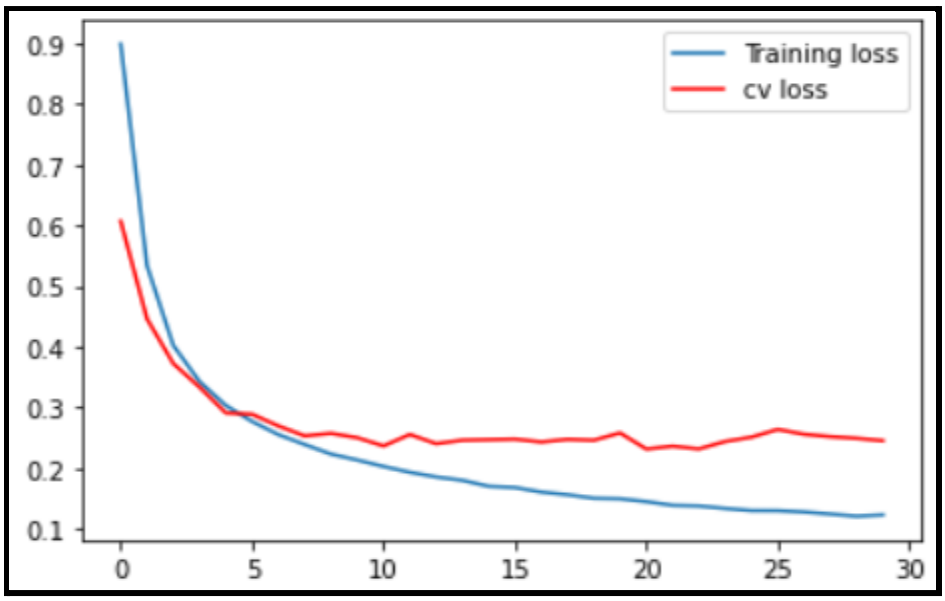
The following is the summary of our model which was described above:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 26, 26, 32)        320
_________________________________________________________________
batch_normalization (BatchNo (None, 26, 26, 32)        128
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 24, 24, 32)        9248
_________________________________________________________________
batch_normalization_1 (Batch (None, 24, 24, 32)        128
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 12, 12, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 12, 12, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 12, 12, 64)        18496
_________________________________________________________________
batch_normalization_2 (Batch (None, 12, 12, 64)        256
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 6, 6, 64)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 6, 6, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 2304)              0
_________________________________________________________________
dense (Dense)                (None, 32)                73760
_________________________________________________________________
dense_1 (Dense)              (None, 10)                330
=================================================================
Total params: 102,666
Trainable params: 102,410
Non-trainable params: 256
```

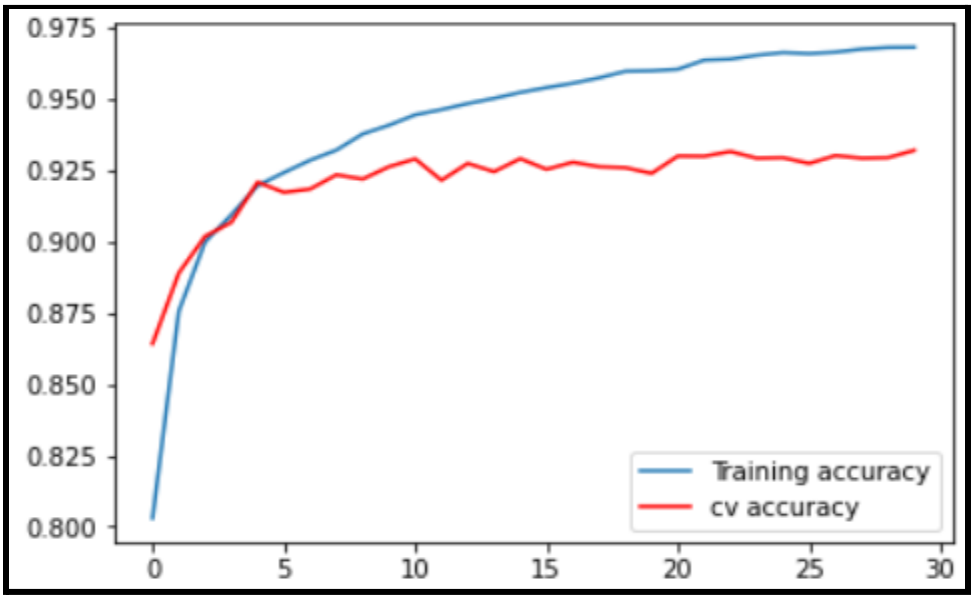
Summary

Performance:

We trained our CNN model for a total of 30 epochs. **The training accuracy was 96.82%**. Upon testing the model, we got **92.91% as the testing accuracy**.



Training and cv loss



Training and cv accuracy

Conclusion:

By having a look at the training and testing errors which we get from the CNN model and those which we got from the Assignment-1 while implementing Deep Learning models we have strong evidence that CNN's work really well in image recognition tasks. This is because of the fact that the pixels in the given input image which are located really close to each other do share a common property and this is what is extracted by the Convolution and Maxpool layers when they work in tandem with each other. In our model to counter overfitting by the CNN network we have added dropout layers in between and also added L2 regularizer terms in the last 2 layers. Also as this was a multi classification problem we used Softmax as the activation function in the output layer.

DONE BY:

Navam Shrivastav 2019A7PS0092H
Aryan Saraf 2019A7PS0138H
Arjun Prasad 2019A7PS0183H
Rishabh Jain 2018A7PS0275H