# Report

The objective of this lab is to build a Convolutional Neural Network model for the Fashion MNIST data set.

* This dataset consists of 60,000 28x28 grayscale training images of clothing items of one of ten types (shirt, dress, etc.)
* It has a test set of 10,000 images

The images are of size 28x28 and there is only one channel as the images are grayscale images.

Various models were designed to get a good accuracy score for the data set. Described below are the architectures of these models.

## Linear 2-Layer DNN

1. Layer 1:

* Fully connected with 28\*28 input nodes and 200 output nodes
* Outputs pass through ReLu

1. Layer 2:

* Fully connected with 200 input nodes and 10 output nodes
* Softmax at the ouput

## Baseline CNN

1. Layer 1:
   1. 2d convolution of 5x5 kernel size, 6 feature maps
   2. each channel then passes through relu
   3. each channel then passes through max pooling 2x2, stride 2x2
   4. output of layer will be 12x12x6
2. Layer 2:
   1. 2d convolution of 5x5x6, 12 feature maps out
   2. each channel then passes through relu
   3. each channel then passes through max pooling 2x2, stride 2x2
   4. output of layer will be 4x4x12 (4x4 image maps, 12 channels)
3. Layer 3:
   1. fully connected layer, 120 outputs
   2. outputs passed through relu
4. Layer 4:
   1. fully connected layer, 60 outputs
   2. outputs passed through relu
5. Layer 5:
   1. softmax layer with 10 outputs corresponding to classes

I tried many variations of the above CNN architecture and arrived at the following architectures that gives the best results

## Variation 1

***Addition of zero padding to the CNN inputs, addition of drop out layer and increased the number of featuremaps. Increased the size of the Fully connected layers***

1. Layer 1:
   1. 2d convolution of 3x3 kernel size, 32 feature maps, padding size = 1
   2. each channel then passes through relu
   3. each channel then passes through max pooling 2x2, stride 2x2
   4. Dropout of 0.25 added at the end of this layer
2. Layer 2:
   1. 2d convolution of 3x3x32, 64 feature maps out
   2. each channel then passes through relu
   3. each channel then passes through max pooling 2x2, stride 2x2
   4. output of layer will be 6x6x64
3. Layer 3:
   1. fully connected layer, 600 outputs
   2. outputs passed through relu
   3. Dropout of 0.25 added at the end of this layer
4. Layer 4:
   1. fully connected layer, 120 outputs
   2. outputs passed through relu
5. Layer 5:
   1. softmax layer with 10 outputs corresponding to classes

## Variation 2

***I added Batch normalization over variation 1***

1. Layer 1:
   1. 2d convolution of 3x3 kernel size, 32 feature maps, padding size = 1
   2. Batch Normalization of size 32 at the output
   3. each channel then passes through relu
   4. each channel then passes through max pooling 2x2, stride 2x2
2. Layer 2:
   1. 2d convolution of 3x3x32, 64 feature maps out
   2. Batch Normalization of size 64 at the output
   3. each channel then passes through relu
   4. each channel then passes through max pooling 2x2, stride 2x2
   5. output of layer will be 6x6x64
3. Layer 3:
   1. fully connected layer, 600 outputs
   2. outputs passed through relu
   3. Dropout of 0.25 added at the end of this layer
4. Layer 4:
   1. fully connected layer, 120 outputs
   2. outputs passed through relu
5. Layer 5:
   1. softmax layer with 10 outputs corresponding to classes

Below were the other hyperparameters that remained the same throughout the experiment:

1. Learning rate = 0.001
2. Batch size = 1000
3. Number of epochs = 10

# Results

|  |  |  |
| --- | --- | --- |
| **Model** | **Training Accuracy after 10 epochs** | **Test Accuracy** |
| Linear 2-Layer DNN | 0.87 | 0.86 |
| Baseline CNN | 0.84 | 0.83 |
| Variation 1 | 0.91 | 0.90 |
| Variation 2 | 0.95 | 0.92 |

**Note**:

* Variation1 model was tried with increasing the number of epochs to 20 and It gave higher test accuracy score of 0.91
* In Variation2 model, I tried using the batch normalization just before the inputs of Layer 2 and 3. This model gave me a training accuracy score of 0.98 and test score of 0.91. This suggests that there was overfit problem in the model.
* The simpler 2-Layer Linear model has better score than the baseline CNN.

The improvement in the performance can be attributed to the following factors:

1. Adding padding to an image gives more space for the kernel to cover the image
2. Increasing the number of feature maps gives the network more confidence in detecting all the features needed to properly classify the image
3. Usage of dropouts helps eliminate overfitting
4. Increasing the size of the linear layers again helps in better classification as the model gets more complex and it can learn the finer details of the dataset.
5. Batch Normalization helps us eliminate the problem of expanding/diminishing gradients which might happen as we have 5 deep layers.

The good thing about the Variation1 and Variation2 models is that there is no overfitting due to increase in the complexity of the network as can be seen from the test accuracy (which is same as that of train accuracy).