Gulfam Hussain
Department of Computer Science
University at Buffalo
Buffalo, NY 14214
gulfamhu@buffalo.edu

Machine Learning – Project 2
Use of Neural network (NN) and Convolutional Neural network for classification purpose

Abstract

As part of this projects, the classification is to be performed on the Fashion MNSIT dataset using various Neural network options. The following three tasks are being performed and the analysis have been recorded in this project report:

1. Build a Neural Network with one hidden layer to be trained and tested on Fashion-MNIST
   dataset. Code from scratch in Python.
2. Build multi-layer Neural Network with open-source neural-network library, Keras on Fashion-MNIST dataset.
3. Build Convolutional Neural Network (CNN) with open-source neural-network library, Keras on Fashion-MNIST dataset.

Introduction:

A neural network is a series of algorithms that endeavours to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria.

A neural network works similarly to the human brain's neural network. A "neuron" in a neural network is a mathematical function that collects and classifies information according to a specific architecture. The network bears a strong resemblance to statistical methods such as curve fitting and regression analysis.

These artificial networks may be used for predictive modelling, adaptive control and applications where they can be trained via a dataset. Self-learning resulting from experience can occur within networks, which can derive conclusions from a complex and seemingly unrelated set of information.

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to analyzing visual imagery.

CNNs are regularized versions of multilayer perceptrons.

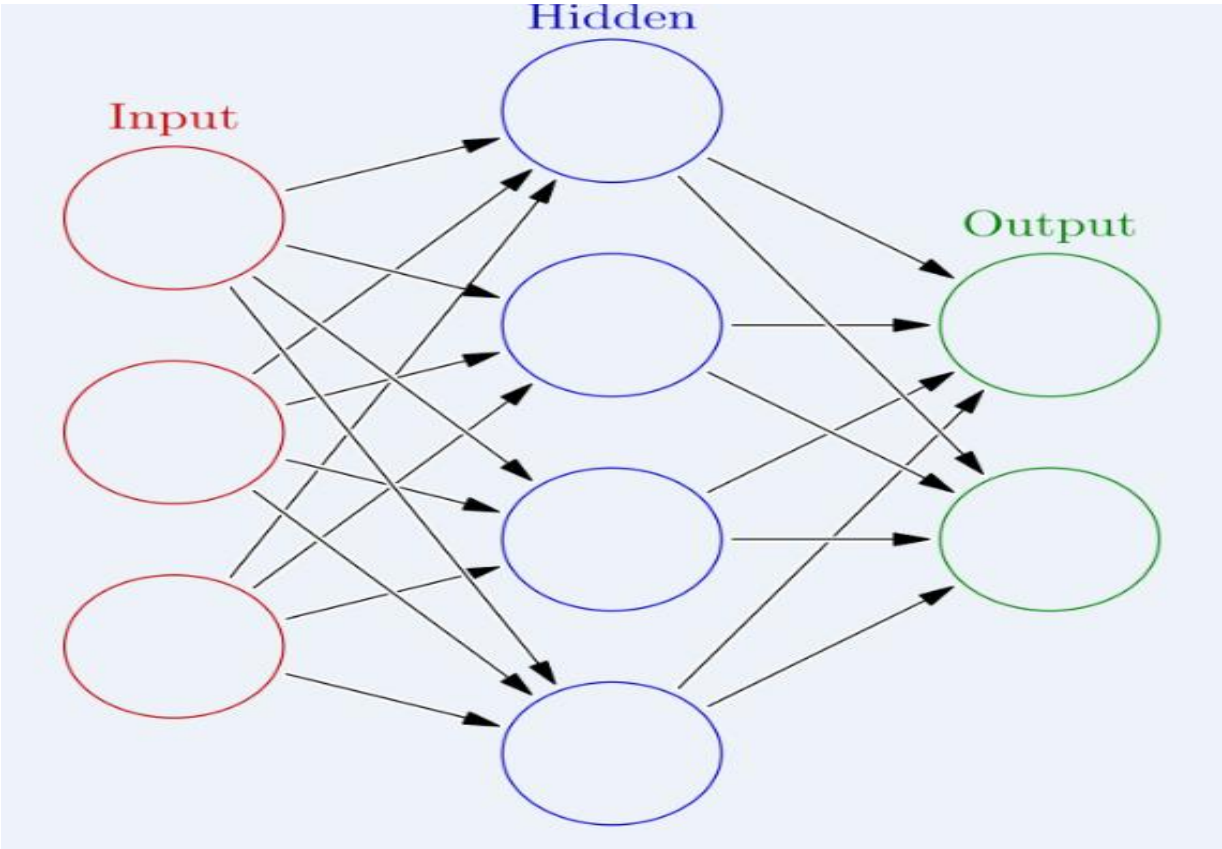
Dataset:

For training and testing of our classifiers, we will use the Fashion-MNIST dataset. The Fashion-MNIST is a dataset of Zalando's article images, 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. Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255. The training and test data sets have 785 columns. The first column consists of the class labels (see above), and represents the article of clothing. The rest of the columns contain the pixel-values of the associated image.
This Fashion MNSIT dataset has been provided as part of this project.

Preprocessing:

The given dataset has been preprocessed before using into the project. Two sets training and test dataset have been divided and the same will be used for the implementation of neural network implementation.

Architecture:


Hidden
Input
Output

Results:

Task 1:

After training the given set of data and tuning the hypermeters, the below accuracy has been achieved on the given set of hypermeters.

Number of hidden layers/nodes = 60

Learning rate = 1
Number of epochs = 1500

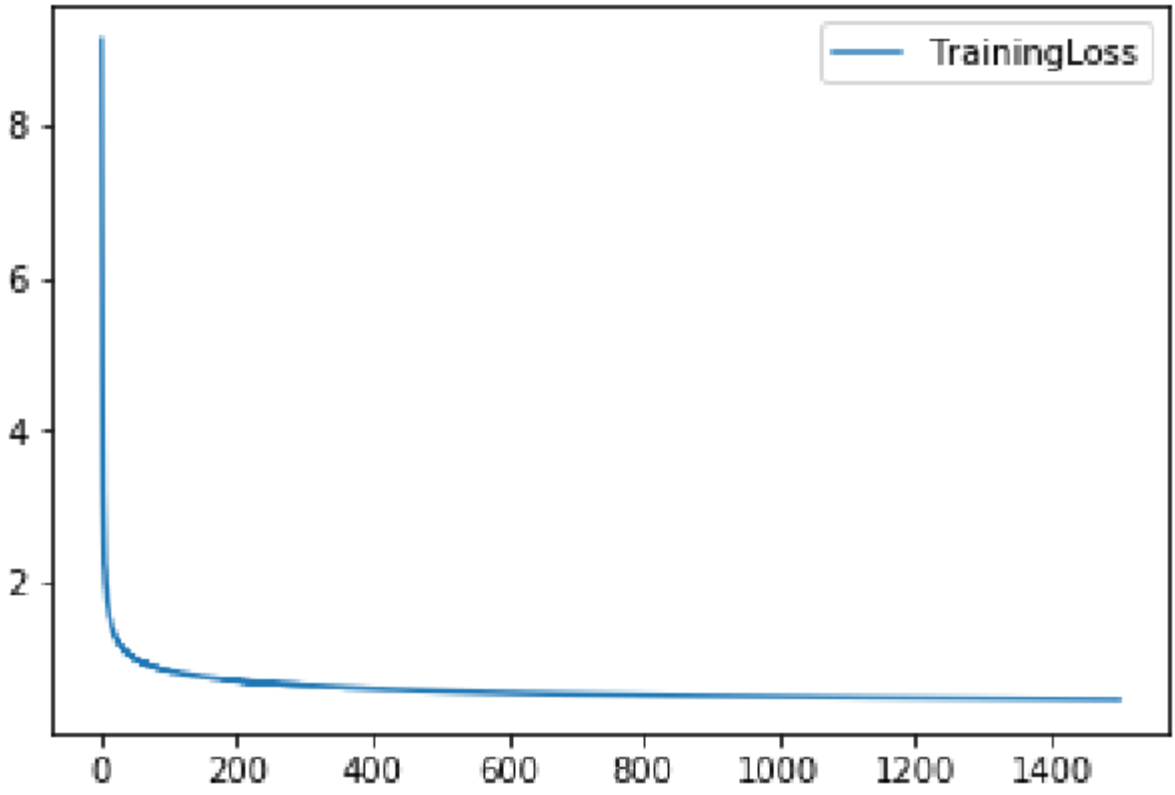
Note: Though the execution time is more but the accuracy rate of 78.94% has been achieved using the above hyperparameters values.

Loss graphs based on epochs:

The number of epochs is a hyperparameter that defines the number times that the learning algorithm will work through the entire training dataset. One epoch means that each sample in the training dataset has had an opportunity to update the internal model parameters.

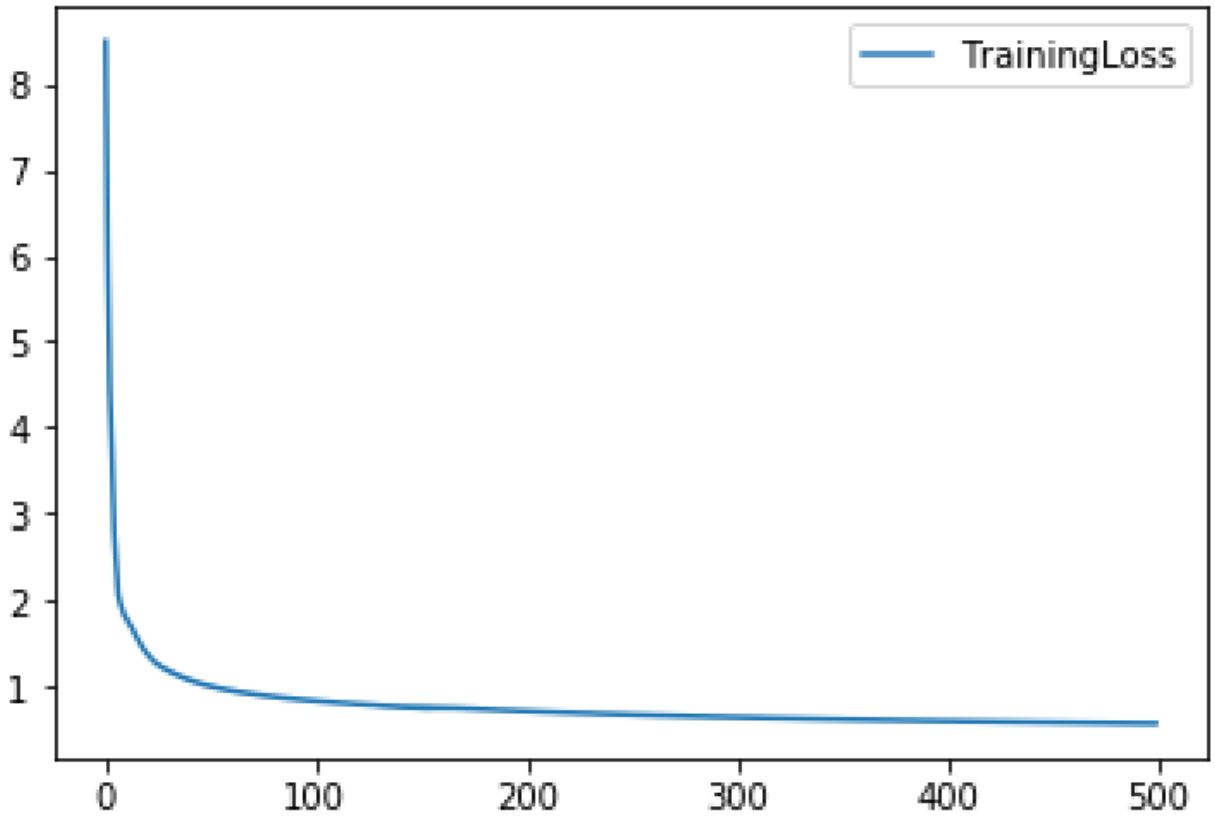
learningrate = 1
epoch = 1500

Number of hidden layers/nodes = 60



learningrate = 1
epoch = 500

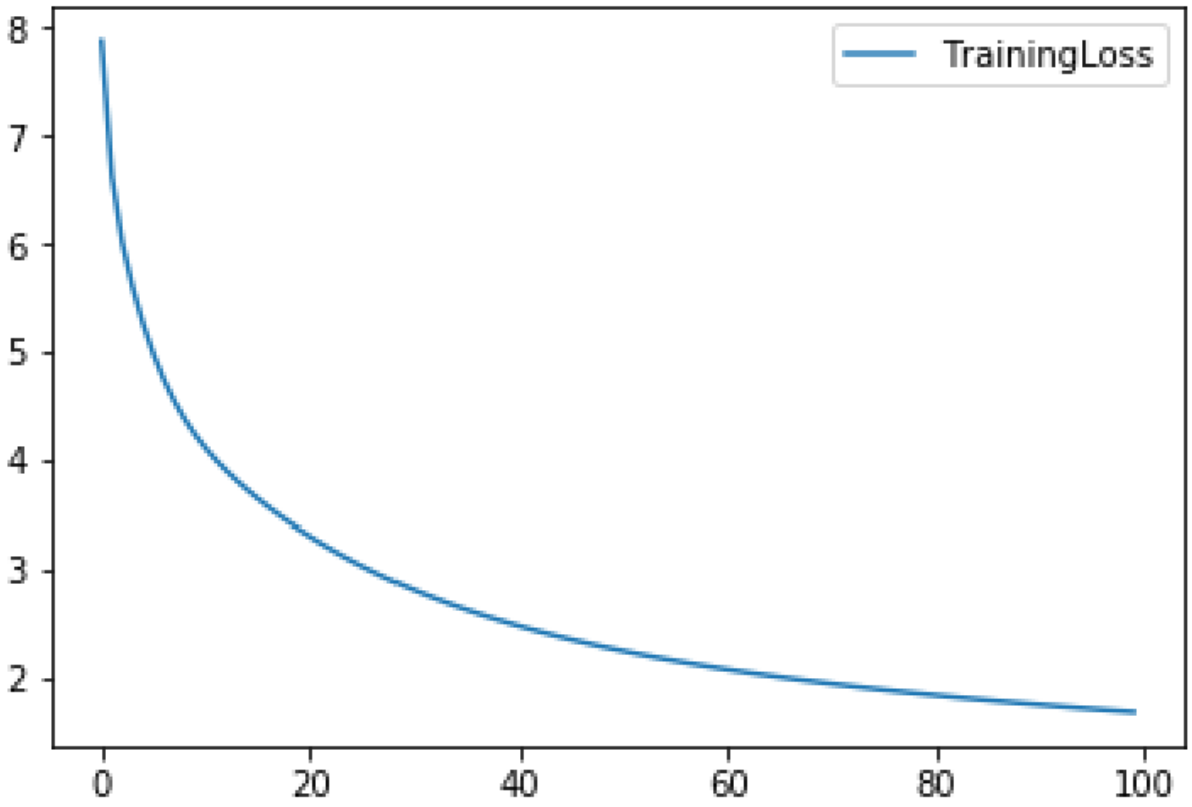
Number of hidden layers/nodes = 60


TrainingLoss
8
7
6
5
4
3
2
1
0
100
200
300
400
500

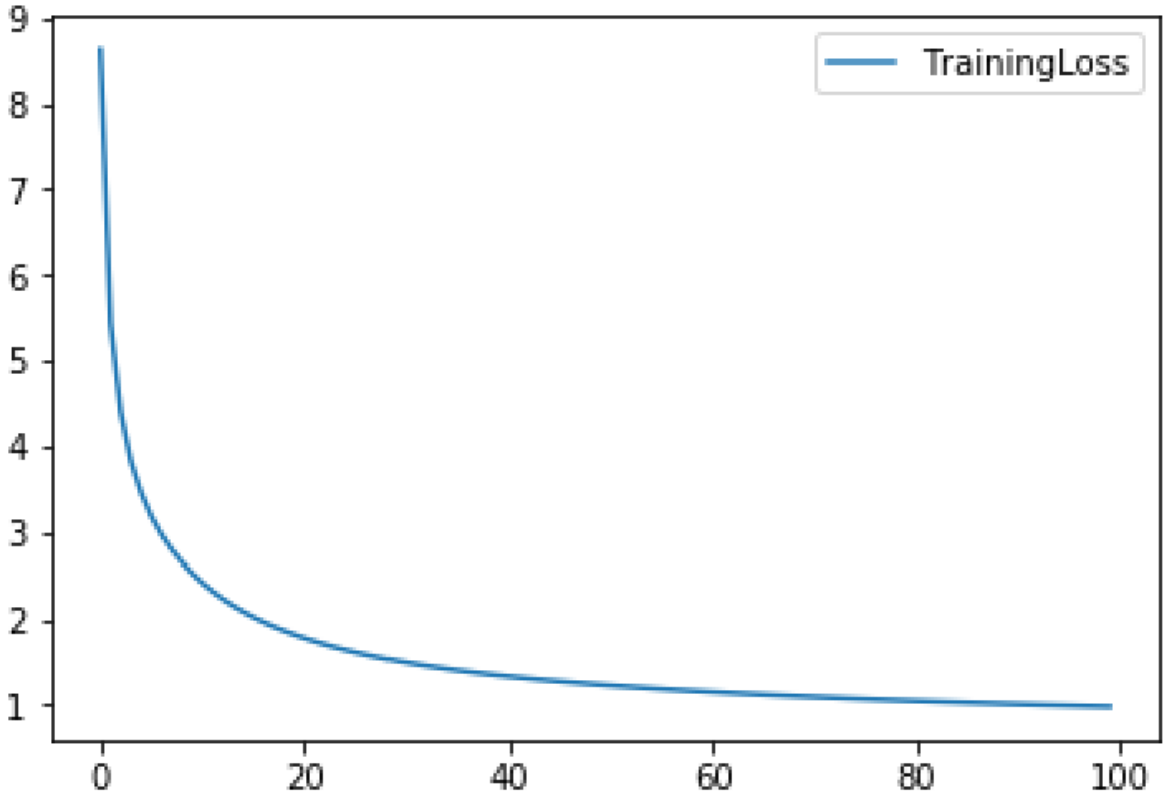
Loss graphs based on learning rate:

The learning rate is a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated. Choosing the learning rate is challenging as a value too small may result in a long training process that could get stuck, whereas a value too large may result in learning a sub-optimal set of weights too fast or an unstable training process.

learningrate = 0.1
epoch= 100



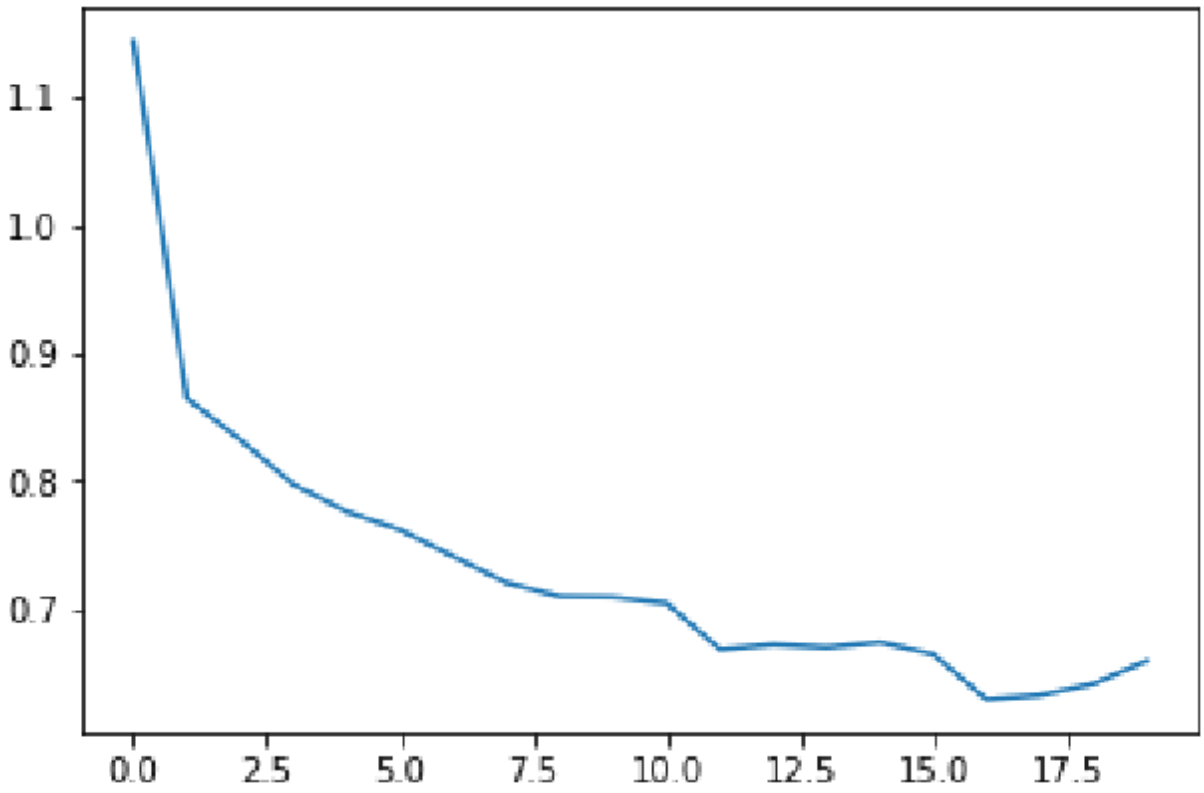
learningrate = 0.5
epoch= 100



Task 2:

Epoch= 50

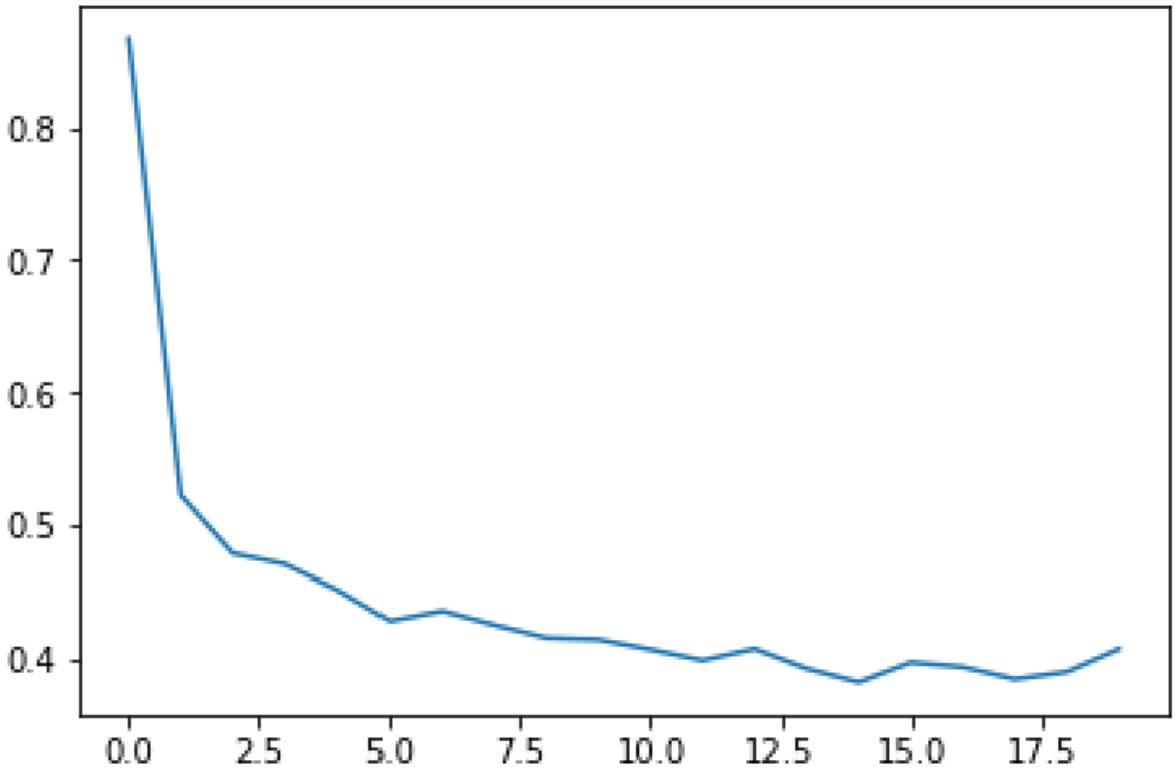
Batch size=20



Task 3:

Epoch= 20

Number of filters =8



Results Performance:

The below images give the idea how the designed neural network modules performed for each of the tasks. The Accuracy (%), loss and the corresponding confusion matrix for each of the tasks have been plotted below.

Task 1:

```
Accuracy:          78.94 %

Confusion matrix:
 [[766  12  26  57  15   2  98   0  24   0]
 [  6 946   7  24  14   1   0   0   2   0]
 [ 14   7 668  11 189   1  92   0  17   1]
 [ 41  36  17 798  60   0  37   2   8   1]
 [  5   3 126  38 736   2  82   0   8   0]
 [  2   1   1   3   0 813   4  99  27  50]
 [168   5 142  46 148   1 457   0  33   0]
 [  0   0   0   0   0  33   0 862   7  98]
 [  5   1  11   5  11  15  25   8 918   1]
 [  0   1   1   0   1  11   3  50   3 930]]
```

Task 2:

```
Accuracy of Task 3:      81.58999999999999 %
Confusion Matrix for Task 3:
 [[793   2  44  62   4   3  72   1  19   0]
 [  3 944   6  32  12   0   1   0   2   0]
 [ 22   0 786  14 145   1  29   0   3   0]
 [ 30  13  31 853  53   2  18   0   0   0]
 [  0   1 153  38 751   0  53   0   4   0]
 [  0   0   0   0   0 935   0  46   2  17]
 [194   2 267  46 151   0 320   0  20   0]
 [  0   0   0   0   0  20   0 909   0  71]
 [ 17   3  17  14  11   6  12   4 913   3]
 [  0   0   1   0   0  12   1  31   0 955]]

Loss of Task 3:  0.49556501092910765
```

Task 3:

```
Accuracy of Task 3:      84.41 %

Loss of Task 3:  0.4366173023223877
Confusion Matrix for Task 3:
 [[802   3  28  52   6   2  98   0   9   0]
 [  2 951   2  27   9   0   9   0   0   0]
 [ 22   1 723  13 140   0  96   0   5   0]
 [ 20  10  13 871  60   0  21   0   5   0]
 [  1   4  90  38 782   0  82   0   3   0]
 [  0   1   0   0   0 943   0  42   3  11]
 [161   2 102  41 127   0 552   0  15   0]
 [  0   0   0   0   0  30   0 926   0  44]
 [  8   3   7   4   9   2   8   3 956   0]
 [  0   0   1   0   0  19   4  39   2 935]]
```

Conclusion:

This particular project based on neural networks design provided an opportunity to learn more on neural network approaches and their implementation in real world problems in the field of AI. The three models were successfully implemented to perform the operation on the Fashion MSNIT dataset and the accuracy results obtained were good in terms of training the dataset and predicting the output after model was trained over the training dataset.

References:

[1] Neural Network
https://en.wikipedia.org/wiki/Neural_network
https://www.investopedia.com/terms/n/neuralnetwork.asp
[2] Class slides and project description documents