Deep Learning Project Report on Fashion MNIST

Ilayda Gurbak
ilayda.gurbak@ozu.edu.tr

Abstract

In this project, I have used three different neural network architectures in Tensorflow using Fashion MNIST dataset. First neural network had only one hidden layer that contains 100 nodes, second one has two hidden layers each contains 100 nodes and the last neural network has two hidden layers each contains 256 nodes. In addition, their performances are compared with each other.

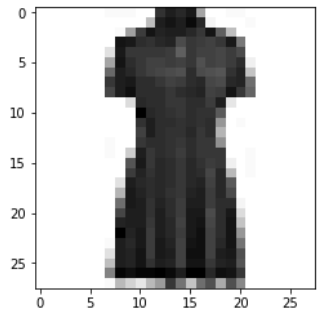
1. INTRODUCTION

Focus of this project was gaining experience in neural network by using tensorflow and solving supervised image classification problem with fashion MNIST dataset. Main focus in neural network on this dataset is observing the change in accuracy depending on number of layers and number of nodes in architecture.

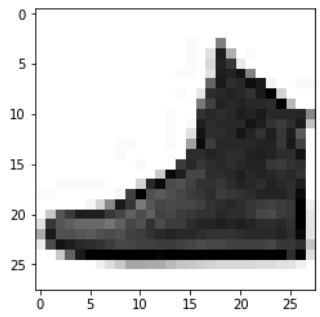
2. DATASET INFORMATION

```
sample_1 = fashion_mnist.train.images[47].reshape(28,28)
# Get corresponding integer label from one-hot encoded data
sample_label_1 = np.where(fashion_mnist.train.labels[47] == 1)[0][0]
# Plot sample
print("y = {label_index} ({label})".format(label_index=sample_label_1, label=label_dict[sa
plt.imshow(sample_1, cmap='Greys')
plt.show()

# Sample 2

# Get 28x28 image
sample_2 = fashion_mnist.train.images[23].reshape(28,28)
# Get corresponding integer label from one-hot encoded data
sample_label_2 = np.where(fashion_mnist.train.labels[23] == 1)[0][0]
# Plot sample
print("y = {label_index} ({label})".format(label_index=sample_label_2, label=label_dict[sa
plt.imshow(sample_2, cmap='Greys')
plt.show()
```

```
y = 3 (Dress)
```




```
y = 7 (Sneaker)
```



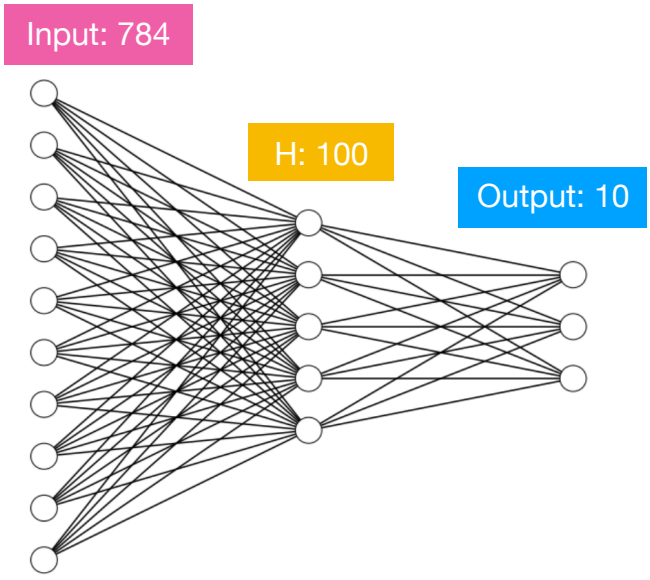
Fashion MNIST is a dataset consisting of 28*28 grayscle images, associated with a label from 10 classes.

```
# Shapes of training set
print("Training set (images) shape: {shape}".format(shape=fashion_mnist.train.images.shape))
print("Training set (labels) shape: {shape}".format(shape=fashion_mnist.train.labels.shape))

# Shapes of test set
print("Test set (images) shape: {shape}".format(shape=fashion_mnist.test.images.shape))
print("Test set (labels) shape: {shape}".format(shape=fashion_mnist.test.labels.shape))
```

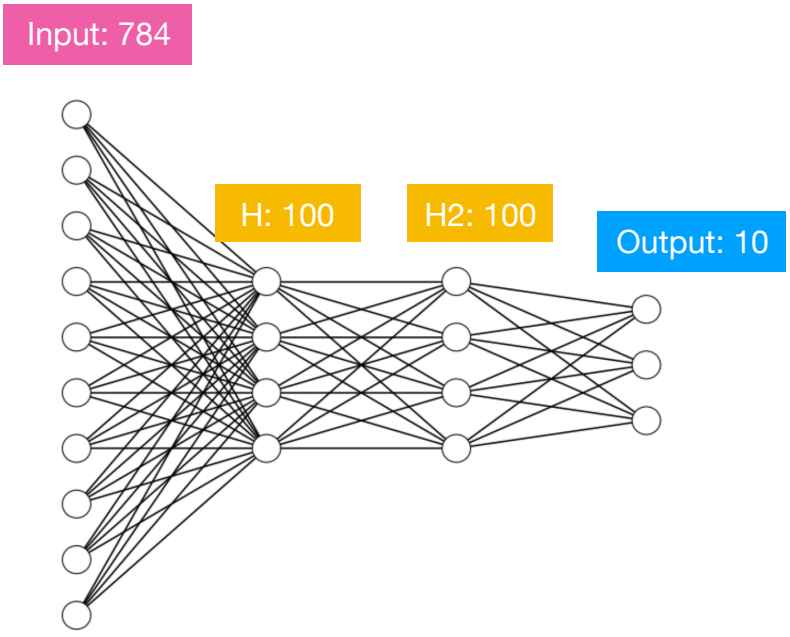
```
Training set (images) shape: (55000, 784)
Training set (labels) shape: (55000, 10)
Test set (images) shape: (10000, 784)
Test set (labels) shape: (10000, 10)
```

3. VISUALIZATION OF THREE DIFFERENT NEURAL NETWORKS

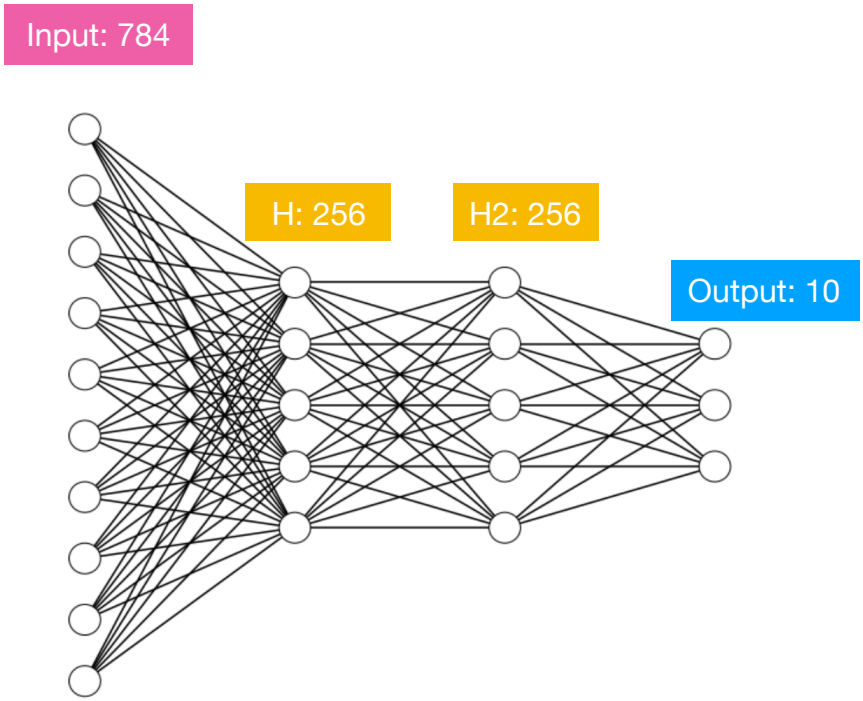
1 hidden layer with 100 nodes :



2 hidden layers with 100 nodes:

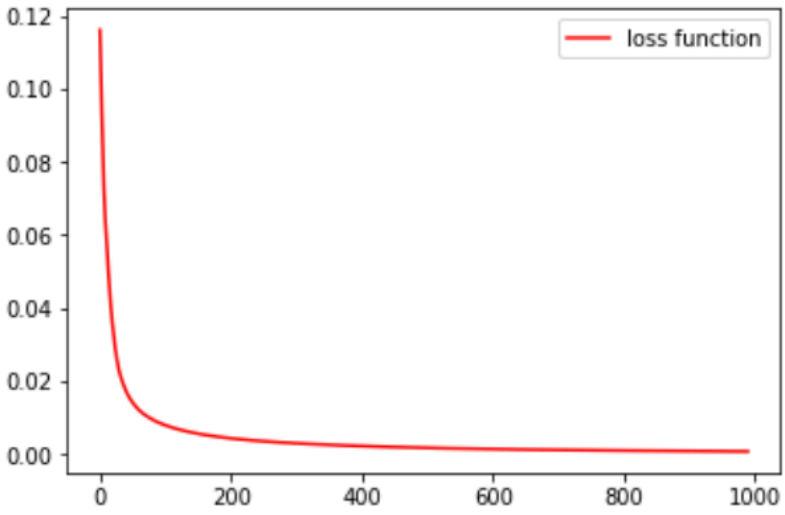


2 hidden layer with 256 nodes:

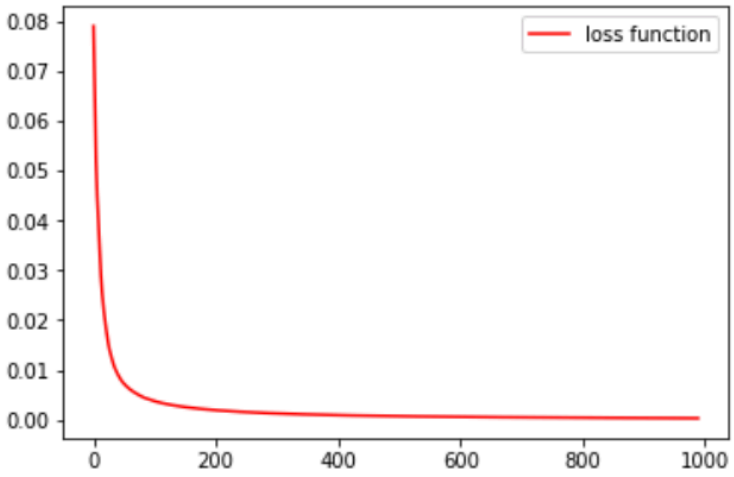


4. COMPARISON OF NEURAL NETWORKS

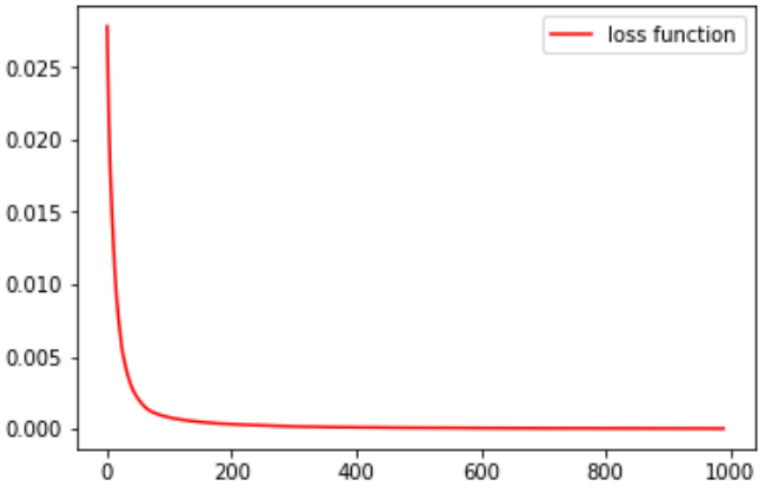
Their loss functions over iterations as following:




```
0) loss: 0.2475
100) loss: 0.0086
200) loss: 0.0045
300) loss: 0.0031
400) loss: 0.0023
500) loss: 0.0018
600) loss: 0.0014
700) loss: 0.0012
800) loss: 0.0010
900) loss: 0.0009
```




```
0) loss: 0.2217
100) loss: 0.0041
200) loss: 0.0020
300) loss: 0.0013
400) loss: 0.0009
500) loss: 0.0007
600) loss: 0.0006
700) loss: 0.0005
800) loss: 0.0004
900) loss: 0.0003
```




```
0) loss: 0.1078
100) loss: 0.0009
200) loss: 0.0004
300) loss: 0.0002
400) loss: 0.0001
500) loss: 0.0001
600) loss: 0.0001
700) loss: 0.0001
800) loss: 0.0000
900) loss: 0.0000
```

Their accuracy on train and test sets :

Train acc: 0.9197636 Test acc: 0.8791	Train acc: 0.93576366 Test acc: 0.8892	Train acc: 0.9483455 Test acc: 0.892

Loss function and accuracy on training & testing show that when our neural network have more nodes and layers its a better fit to our data.