

**CSE 574 : Introduction to Machine Learning**

**Programming Assignment 1**

# **Handwritten Digits Classification Using Neural Networks**

**Submitted by**

**Group #30**

Amey Mhaskar

Gideon Paul

Julien Kann

## SECTION 01:

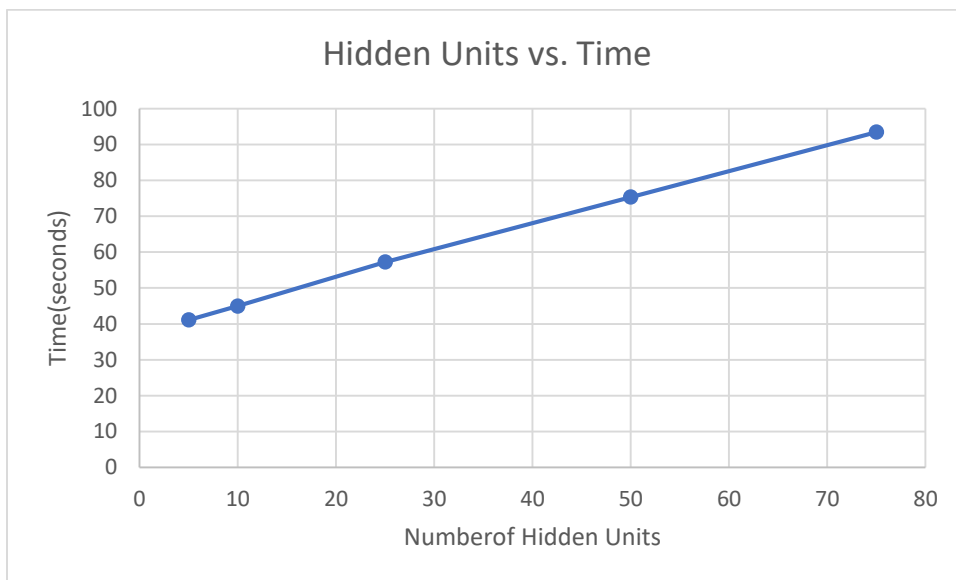
### Hyper-Parameter Selection:

We implemented the Neural Network for handwritten digit recognition and trained it using the training data. A model for the classification of digits was created using the Neural Network. We then made use of this model to work on the validation data trying different values for the hyper-parameters. The hyper-parameters used are the lambda value for regularization and the number of units in the hidden layer of the network. Then based of all the parameters we choose the one with the maximum accuracy as our hyper-parameters

### RESULTS:

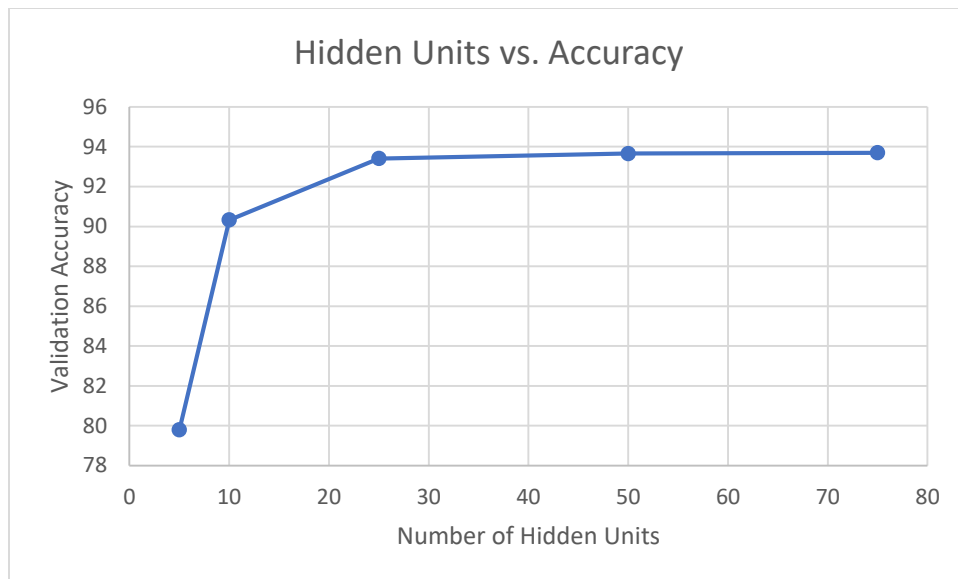
The results of our validation are shown below with the help of some graphs:

#### Hidden Units vs Time:



The above figure show the relation between the number of hidden units in the hidden layer and the execution time. As expected the execution time increases as the number of hidden units do.

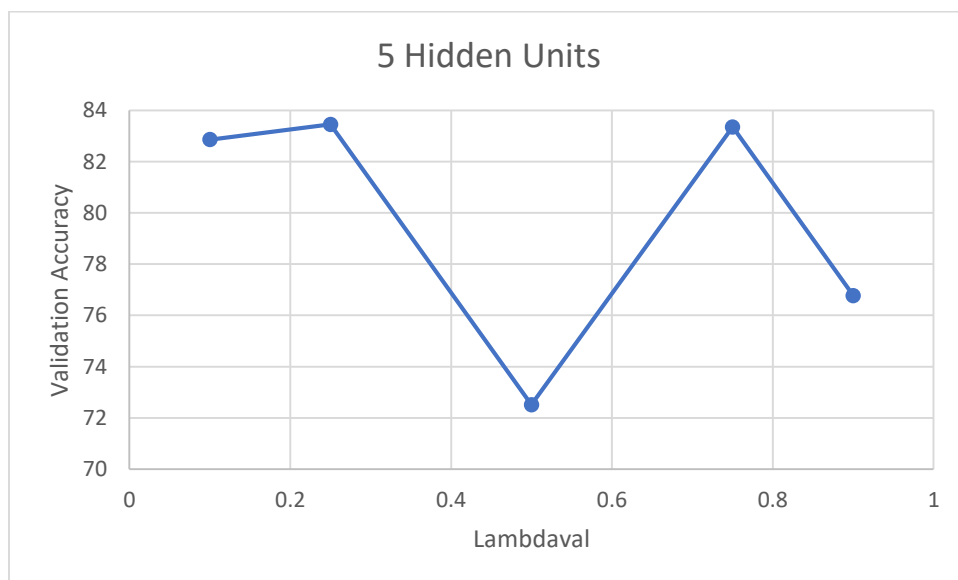
### Hidden Units vs Accuracy:



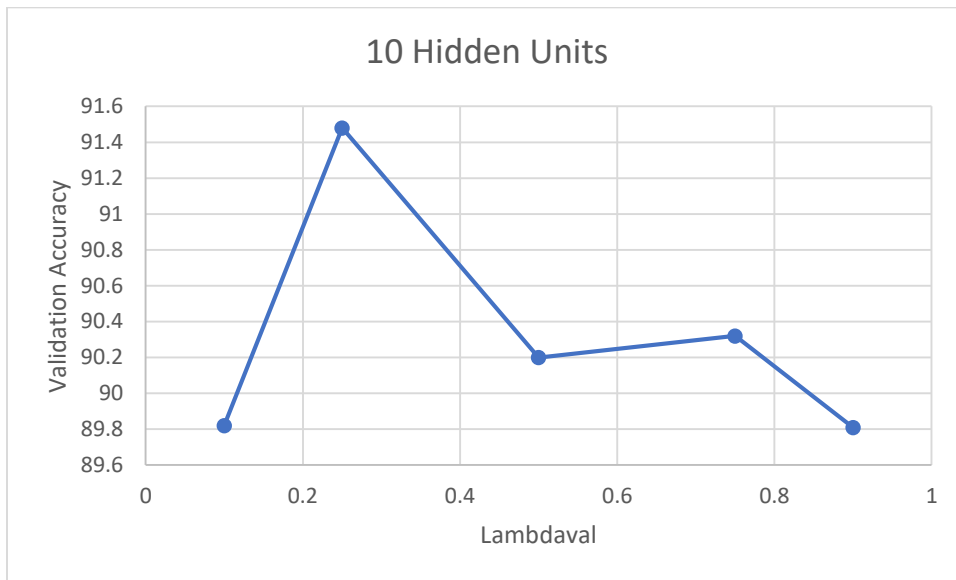
The relation between hidden units and validation accuracy is less linear than vs time. Expectantly the accuracy improves when the number of hidden units is increases, but after a certain number of hidden units it is observed that there is no increase in the accuracy.

The following section shows graphs of **validation accuracy for different lamba values (between 0 and 1) by keeping the number of hidden units fixed** for a set of lambda.

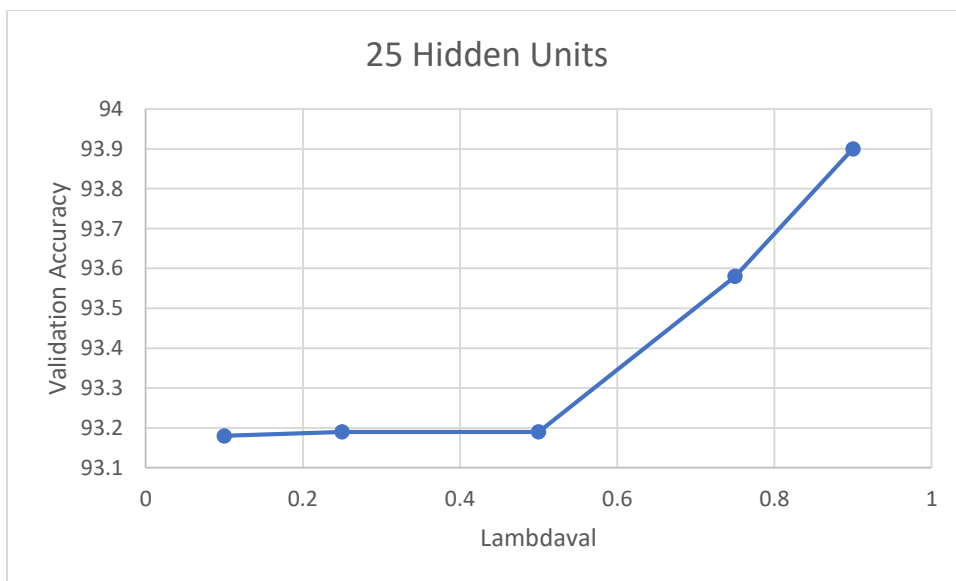
### Number of Hidden Units=5



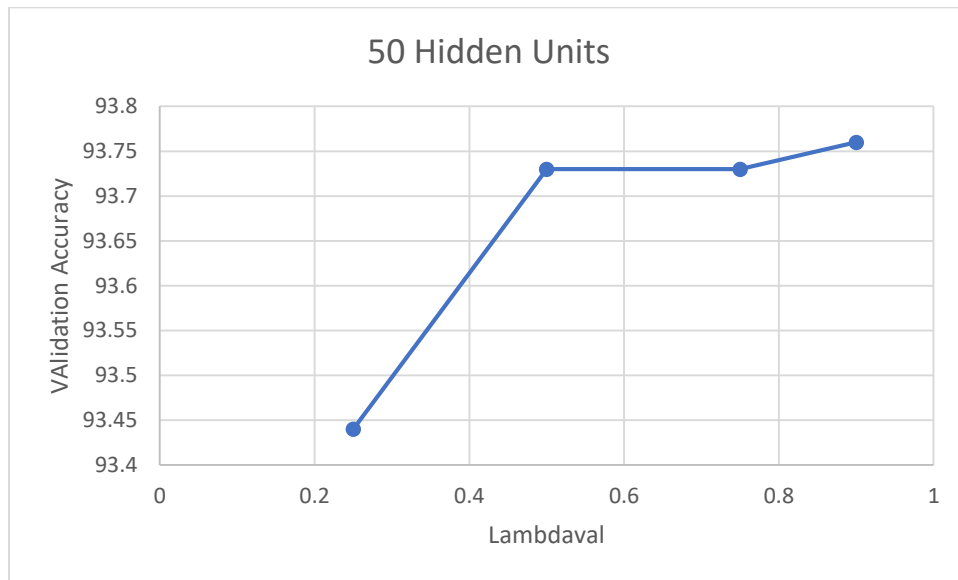
**Number of Hidden Units=10:**



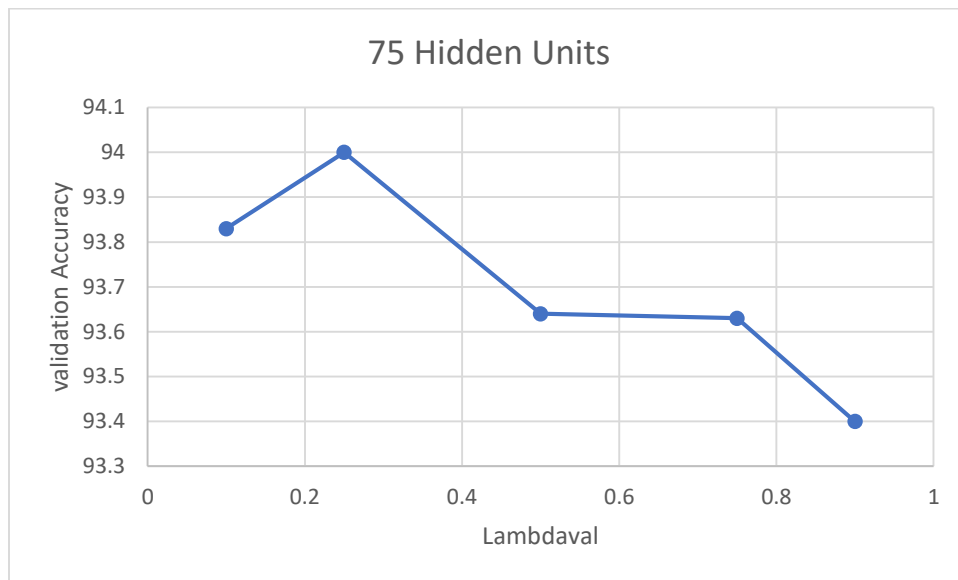
**Number of Hidden Units=25:**



**Number of Hidden Units=50:**



**Number of Hidden Units=75:**



Summary of Results:

	A	B	C	D	E	F
1	hidden	lambdaval	accuracy	train_start	train_end	diff
2	5	0.1	82.86	29:03:00	29:43:00	0:40:00
3	5	0.25	83.45	29:43:00	30:22:00	0:39:00
4	5	0.5	72.51	30:23:00	31:03:00	0:40:00
5	5	0.75	83.35	31:03:00	31:45:00	0:42:00
6	5	0.9	76.77	31:46:00	32:31:00	0:45:00
7						
8						
9	hidden	lambdaval	accuracy	train_start	train_end	diff
10	10	0.1	89.82	32:31:00	33:15:00	0:44:00
11	10	0.25	91.48	33:15:00	33:59:00	0:44:00
12	10	0.5	90.2	33:59:00	34:45:00	0:46:00
13	10	0.75	90.32	34:46:00	35:31:00	0:45:00
14	10	0.9	89.81	35:31:00	36:17:00	0:46:00
15						
16						
17	hidden	lambdaval	accuracy	train_start	train_end	diff
18	25	0.1	93.18	36:18:00	37:15:00	0:57:00
19	25	0.25	93.19	37:15:00	38:12:00	0:57:00
20	25	0.5	93.19	38:13:00	39:08:00	0:55:00
21	25	0.75	93.58	39:08:00	40:07:00	0:59:00
22	25	0.9	93.9	40:08:00	41:07:00	0:59:00
23						
24						
25	hidden	lambdaval	accuracy	train_start	train_end	diff
26	50	0.1	93.64	41:08:00	42:22:00	1:14:00
27	50	0.25	93.44	42:22:00	43:36:00	1:14:00
28	50	0.5	93.73	43:37:00	44:52:00	1:15:00
29	50	0.75	93.73	44:52:00	46:12:00	1:20:00
30	50	0.9	93.76	46:12:00	47:27:00	1:15:00
31						
32						
33	hidden	lambdaval	accuracy	train_Start	train_end	diff
34	75	0.1	93.83	47:27:00	49:02:00	1:35:00
35	75	0.25	94	49:02:00	50:38:00	1:36:00
36	75	0.5	93.64	50:38:00	52:15:00	1:37:00
37	75	0.75	93.63	52:15:00	53:50:00	1:35:00
38	75	0.9	93.4	53:51:00	55:17:00	1:26:00

Based on the validation done on the data we got the maximum accuracy (94%) for 75 hidden units and 0.25 as the regularization term or lambda value. We have thus made use of these two hyper-parameter values for training of our neural network.

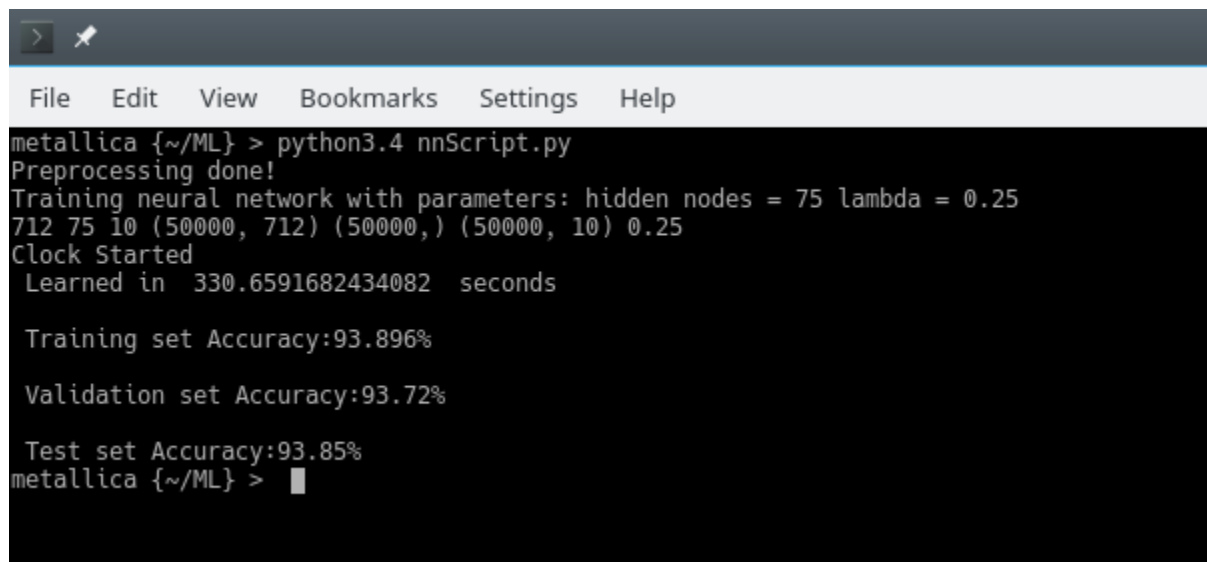
**Therefore, we have used the following values for hyper-parameters:**

- 1) Number of hidden units= 75
- 2) Regularization term (lambda) = 0.25

## SECTION 02:

**Accuracy of neural network on handwritten digits' test data:**

In this section we trained our neural network in the nnScript.py file and tried to predict handwritten digits from the MNIST database. The output node has 10 units corresponding to the digits 0 to 9. The forward pass and the back-propagation functions were coded by us.



```
metallica {~/ML} > python3.4 nnScript.py
Preprocessing done!
Training neural network with parameters: hidden nodes = 75 lambda = 0.25
712 75 10 (50000, 712) (50000,) (50000, 10) 0.25
Clock Started
Learned in 330.6591682434082 seconds

Training set Accuracy:93.896%

Validation set Accuracy:93.72%

Test set Accuracy:93.85%
metallica {~/ML} > █
```

*Fig: Screenshot of nnScript.py execution*

The accuracies are as follows:

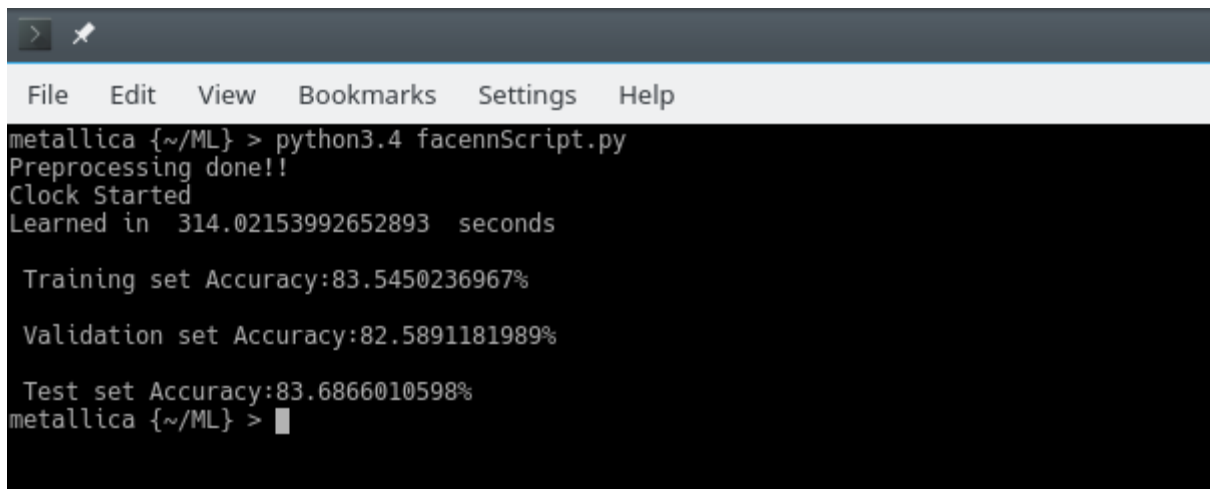
<b>Training Set</b>	93.89 %
<b>Validation Set</b>	93.72 %
<b>Test Set</b>	93.85 %

## SECTION 03:

### Accuracy of neural network on CelebA dataset:

For this section we executed the facennScript.py file for the CelebA database and tried to predict the outputs. In case of the CelebA database the outputs were binary either 0 or 1. The forward pass and the back-propagation functions were the same as the ones we used for the handwritten digit's classification.

The accuracy of facennScript.py on CelebA dataset is as follows:



```
metallica {~/ML} > python3.4 facennScript.py
Preprocessing done!!
Clock Started
Learned in 314.02153992652893 seconds

Training set Accuracy:83.5450236967%

Validation set Accuracy:82.5891181989%

Test set Accuracy:83.6866010598%
metallica {~/ML} > █
```

*Fig: Screenshot of facennScript.py execution*

<b>Training Set</b>	83.545 %
<b>Validation Set</b>	82.589 %
<b>Test Set</b>	83.686 %

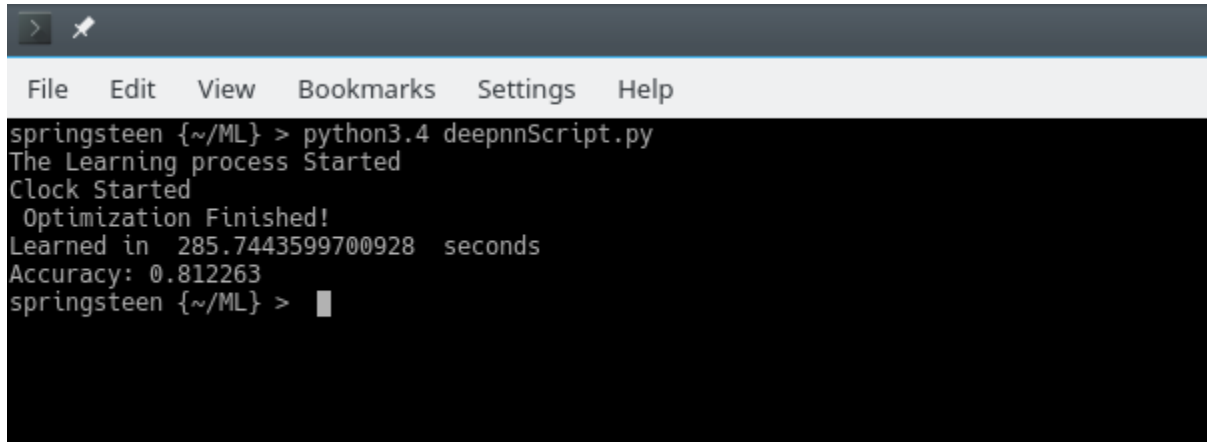
Totally, it took around 314 seconds for the Neural Network to learn.




## SECTION 04:

### Comparison of neural network with deep neural network (Tensorflow):

In this section we ran the `deppnnScript.py` file provided to us to compare the accuracy for CelebA dataset with the accuracy achieved for the dataset with the `facennScript` that doesn't make use of the Tensorflow library.



```
> 
File Edit View Bookmarks Settings Help
springsteen {~/ML} > python3.4 deepnnScript.py
The Learning process Started
Clock Started
Optimization Finished!
Learned in 285.7443599700928 seconds
Accuracy: 0.812263
springsteen {~/ML} > █
```

*Fig: Screenshot of `deepnnScript.py` execution*

As shown in the above image, on the CelebA dataset, by using Tensor Flow library we get an accuracy of 81% as opposed to the 83% we get on using the `facennScript`. The results fall really close. However, there is a huge difference in the runtimes. The tensor flow library trained the neural network with hidden nodes greater than the number of hidden nodes of the neural network that we coded. This shows that the Tensor Flow library is highly efficient. Generally, it should be that the deep neural network has higher accuracy than a neural network. But in some cases, using higher number of hidden units will damage the accuracy of the system.