CSE 574 Introduction to Machine Learning

Assignment #1

Multilayer Perceptron Neural Networks

on

Hand written Digit Classification

&

Facial data classification against a Deep Neural Network

Group #41

Pruthvi Mulagala – 50208595

Madhav Jakkampudi - -50206563

Vinod Kumar Veparala – 50208038

Objective:

To implement a Multilayer Perceptron Neural Network and evaluate its performance in handwritten digits. Furthermore use the same network to analyse a more challenging face dataset and compare the performance of the neural network against a deep neural network using the Tensor Flow library.

Implementation of Multilayer Perceptron Neural Network:

For the implementation of Multilayer Perceptron Neural Network, we segregated the training data of 60,000 into two sets – a data set of 50,000 to train our neural network and the remaining 10,000 as a validation set to check the veracity of the training results. And then the trained neural network is verified against another testing data set of 10,000.

Pre-processing:

Out of the 784 features available for each image, we did pre-processing to remove some of the features which doesn't vary across the data set. In order to achieve this we computed the variance for each feature across all the data sets (includes both the training and the testing data) and removed the features which has a variance less than 1. It is worth noting here that the feature values ranges from 0 t0 255 with 0 being black and 255 being white with the values between representing different shades of grey. So variance less than 1 would imply that the feature doesn't vary much across the data. By doing this, our feature count reduced to 662 and we used only these features to train the data.

Classification Accuracy:

The various factors which impact the classification accuracy of the testing data are as follows:

Hyper parameters:

By varying the regularization parameters from 0 to 60 with an increment of 5 we tried to identify the lamda for which we would obtain the maximum accuracy of our neural network on the test data set. From our observations, with a hidden layer node count of 50 and with 50 iterations, the maximum test accuracy obtained is **94.97%** when the regularization parameter is **10**. However this parameter is not varying much with a variation of less than 1% for all the lambda values in our range.

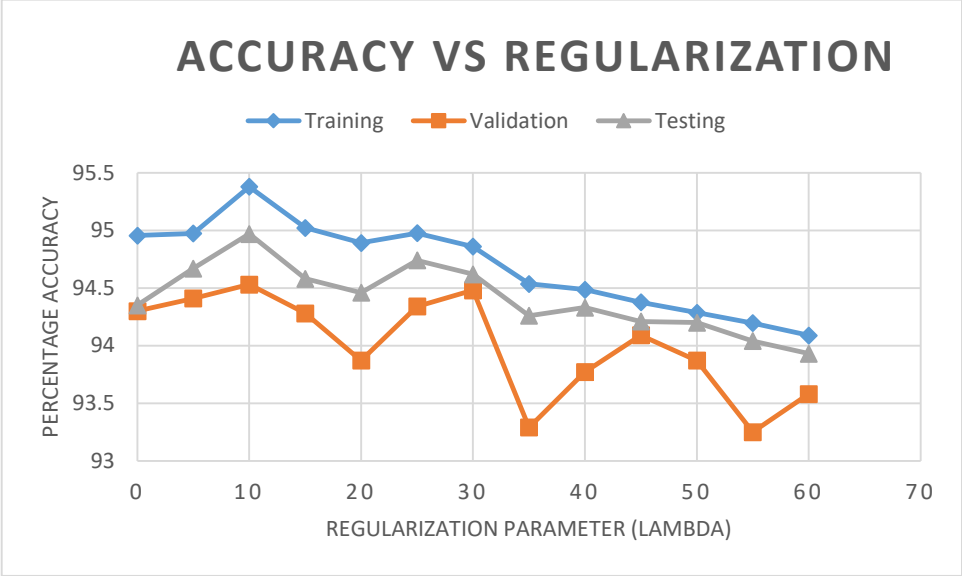


Table representing the variation of accuracy with regularization parameter (lambda)

			Accuracy		
Max Iterations	**Hidden Nodes**	**Lamda**	**Training**	**Validation**	**Test**
50	50	0	94.956	94.3	94.35
50	50	5	94.974	94.41	94.67
50	50	10	95.38	94.53	94.97
50	50	15	95.022	94.28	94.58
50	50	20	94.892	93.87	94.46
50	50	25	94.976	94.34	94.74
50	50	30	94.86	94.48	94.62
50	50	35	94.536	93.29	94.26
50	50	40	94.486	93.77	94.33
50	50	45	94.376	94.09	94.21
50	50	50	94.286	93.87	94.2
50	50	55	94.196	93.25	94.04
50	50	60	94.088	93.58	93.93

Nodes in hidden layer:

By increasing the number of nodes in the hidden layer from 4 to 20 with an increment of 4 every step with 50 iterations and the regularization parameter (lambda) as 10, we observed that the accuracy increases almost with every increment and attains a maximum value of **93.61%** at **20**. This observation implies that there is a scenario of under fitting till the node count is raised to **20** as the accuracy was quite low. By increasing the nodes beyond 20 till 100 with an increment of 10, the accuracy increases slightly with a maximum of **94.56%** with **80** hidden layers.

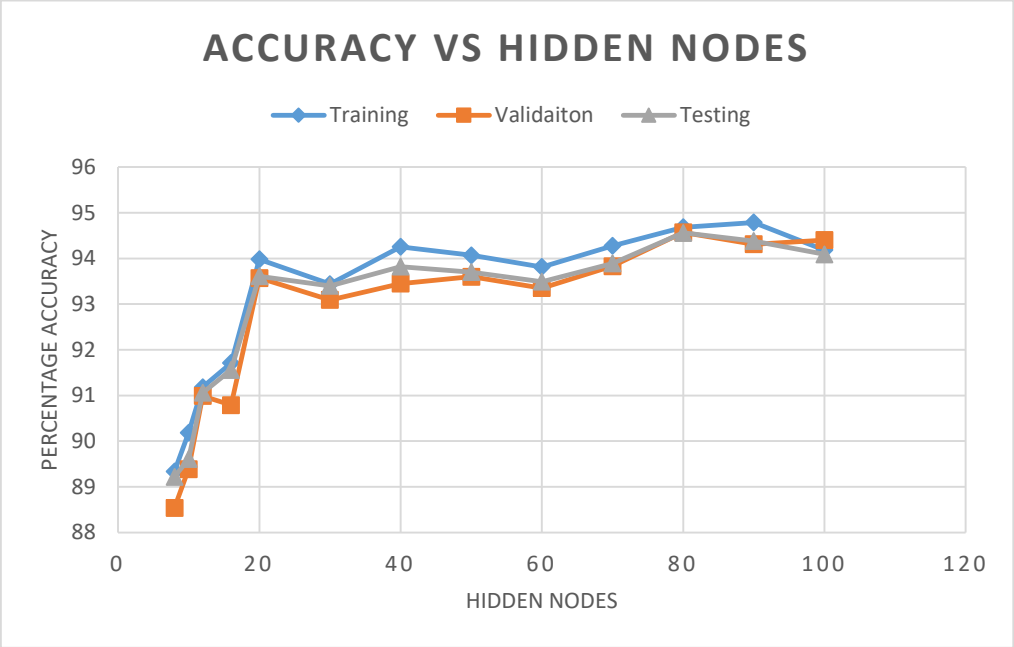


Table representing the variation of accuracy with number of hidden layer nodes

			Accuracy		
Max Iterations	**Hidden Nodes**	**Lamda**	**Training**	**Validation**	**Test**
50	2	10	31.322	29.29	31.03
50	4	10	50.35	48.28	49.92
50	8	10	89.34	88.54	89.22
50	10	10	90.182	89.39	89.61
50	12	10	91.182	90.99	91.06
50	16	10	91.712	90.79	91.56
50	20	10	93.982	93.57	93.61
50	30	10	93.438	93.09	93.4
50	40	10	94.254	93.45	93.82
50	50	10	94.07	93.6	93.7
50	60	10	93.812	93.35	93.49
50	70	10	94.274	93.83	93.89
50	80	10	94.682	94.57	94.56
50	90	10	94.788	94.31	94.38
50	100	10	94.184	94.4	94.09

Number of iterations:

With a hidden layer node count of 50 and a regularization parameter value(lambda) of 10, on increasing the iterations from 10 to 100 with an increment of 10 at each step, it is observed that with more number of iterations the accuracy tend to increase. By doing so, we achieved the maximum test accuracy of 96.54% with 100 iterations.

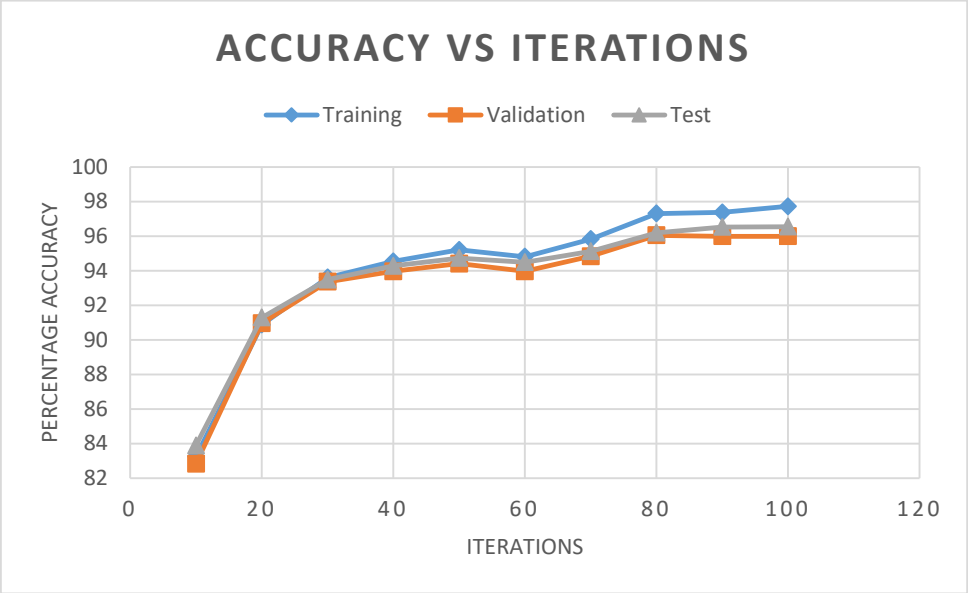
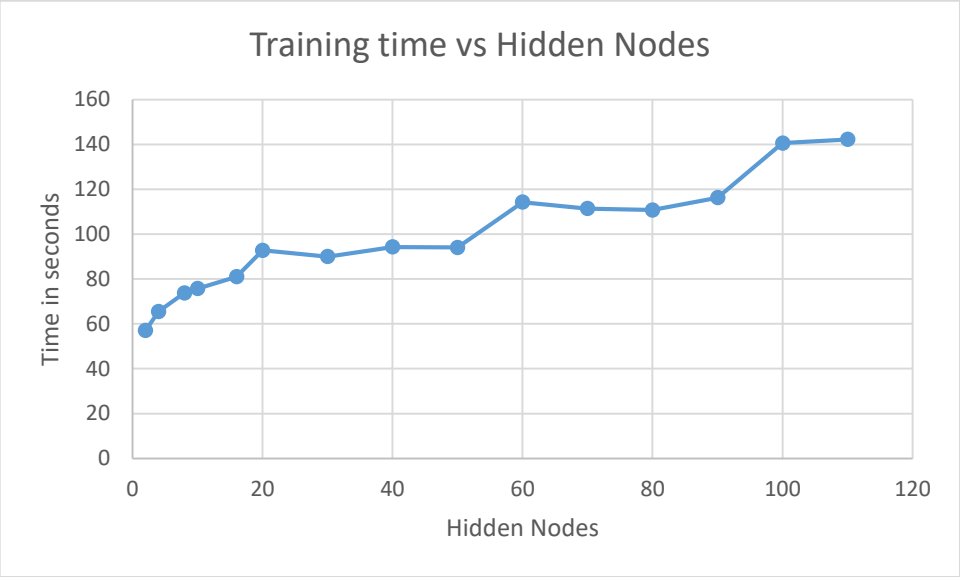


Table representing the variation of accuracy with number of iterations

			Accuracy		
Max_Iterations	Hidden Nodes	Lamda	Training	Validation	Test
10	50	10	83.254	82.83	83.89
20	50	10	90.876	90.96	91.29
30	50	10	93.604	93.36	93.48
40	50	10	94.532	93.96	94.28
50	50	10	95.204	94.41	94.73
60	50	10	94.802	93.96	94.49
70	50	10	95.82	94.83	95.12
80	50	10	97.306	96.04	96.19
90	50	10	97.366	95.99	96.53
100	50	10	97.72	95.99	96.54

In terms of the time taken for training the neural network, we observed that the training time increase with an increase in the number of nodes in the hidden layer. A plot representing the time rise with the addition of hidden nodes is as below:



Conclusion:

From our experimentation we observed that for a neural network with one hidden layer, the maximum accuracy is achieved when the hidden nodes are set to 80 nodes with the regularization parameter around 10 (however there is no direct relationship with this parameter with a value similar at almost all other values of lambda) and its value is 94.97% with 50 iterations. With an increase in the number of iterations we were even able to achieve an accuracy of 96.54% on the testing data. Scenarios of under-fitting was observed until the hidden node count is raised till 20.

Analysis of Facial data classification with Deep Neural Network:

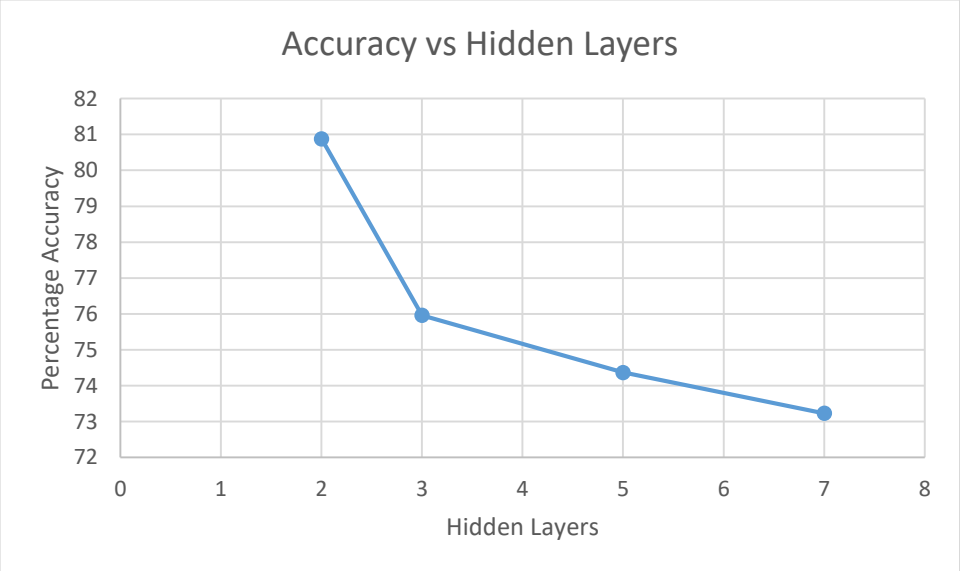
Using Custom Neural Network:

The facennScript.py was modified to include the neural network that was developed in nnScript.py for digit classification and appropriate changes were made to accommodate for 2 output classes and 256 hidden nodes. We obtained an accuracy of **85.77%** and took **183.3** seconds using this setup.

Using TensorFlow:

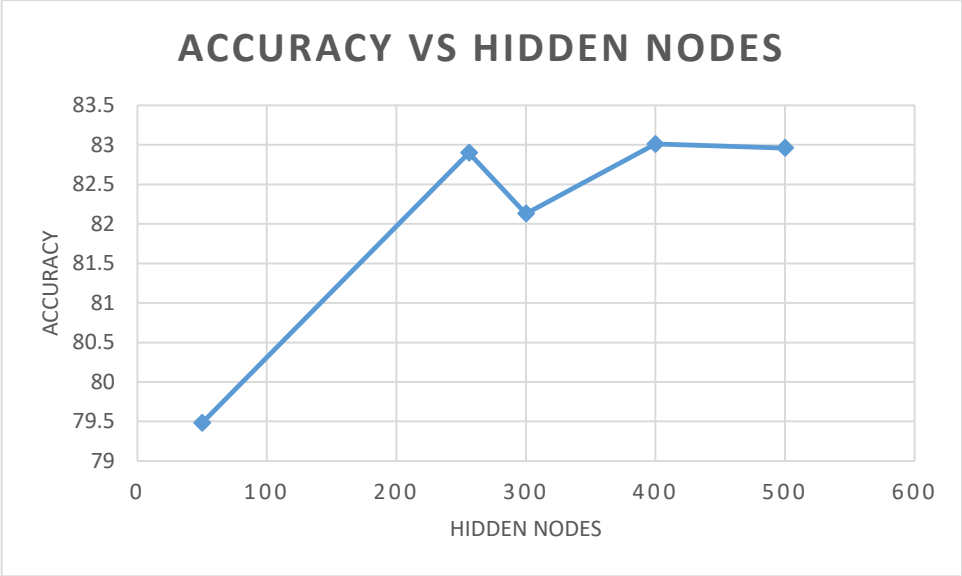
All test were conducted on a machine having 2.3Ghz i5 processor, 8GB RAM and TensorFlow was compiled without GPU support. To make the timing results comparable with those obtained by our custom Neural Network we first ran the deepnnScript with a single hidden layer and 256 hidden nodes which yielded an accuracy of **79.8259%** and took **286.67** seconds.

Further, the following graph and table shows how increasing the number of layers affects the accuracy



Hidden layers	Accuracy (%)
2	80.88
3	75.96
5	74.37
7	73.23

Also, the table below show how increasing the number of nodes in the hidden layers affects the accuracy



Hidden layer Nodes	Accuracy (%)
50	79.48
256	82.9
300	82.13
400	83.01
500	82.96

Conclusion:

From the above tables we can conclude that increasing the number of layers resulted in a decrease in accuracy possibly due to over-fitting. And as we increase the number of nodes and keep the number of layers constant there was an increase in accuracy but after having 400 nodes the accuracy stabilized.