CSE 574 Fall 2021 Introduction to Machine Learning

Programming Assignment 2

Handwritten Digits Classification

Group 72

abhawsar@buffalo.edu, 50416025

jayeshki@buffalo.edu, 50417114

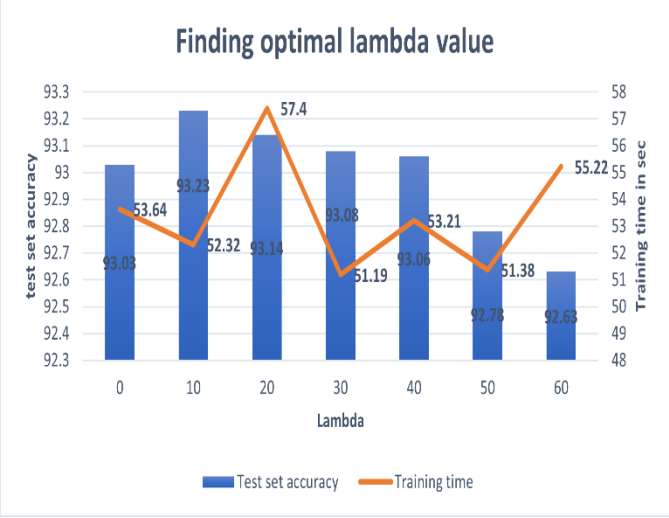
akumar56@buffalo.edu, 50412887

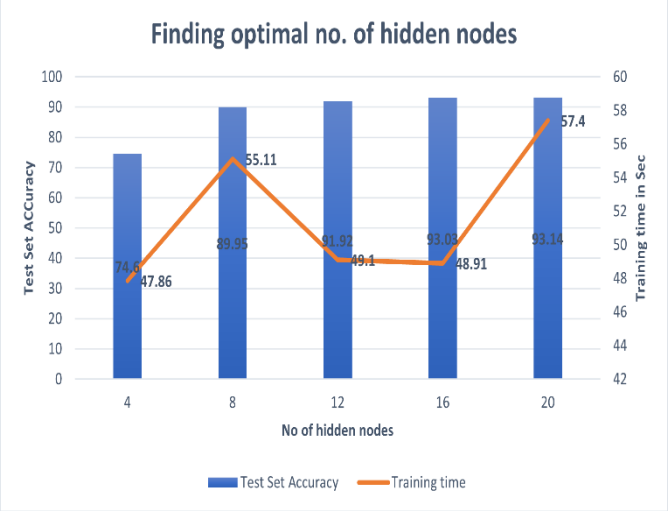
1: Introduction

This assignment is about implementation of a Multilayer Perceptron Neural Network and evaluation of its performance in classifying handwritten digits on the MNIST Dataset. We used the same network to analyze a face dataset (celebA) to recognize who wore spectacles and compare the performance of the neural network against a deep neural network and a convolutional neural network using the TensorFlow Library.

2: Choosing Hyper-Parameters for Neural Network

The regularization coefficient (lambda) and the number of hidden units are the hyper parameters to be chosen for the neural network. Following are the results for obtaining an optimal value of lambda and number of hidden nodes:





It can be observed that for a fixed lambda value, the test accuracy increases almost linearly with the number of hidden nodes. Training time also increases with increased complexity with a greater number of hidden units. Test accuracy and training time almost increase linearly with number of hidden units. Highest accuracy of 93.14 is obtained for 20 hidden units. Also, it can be observed that highest accuracy is obtained for lambda value 10. Increasing lambda value more doesn't help in regularization. The highest test accuracy of 93.23 is observed for lambda value 10.

3: Accuracy on Handwritten Digits Classification

We have implemented the functions preprocess(), sigmoid(), nnObjFunc() and nnPredict() in the nnScript.py script and ran it on MNIST dataset that gave us the following results:

- The highest accuracy observed on handwritten digits data is 93.23%
- Optimal lambda value = 10
- Optimal number of hidden nodes = 20

4: Accuracy on celebA data set

As we have already implemented functions of neural network in nnScript.py, we use same functions in facennScript.py for obtaining the results on CelebA data set which are as follows:

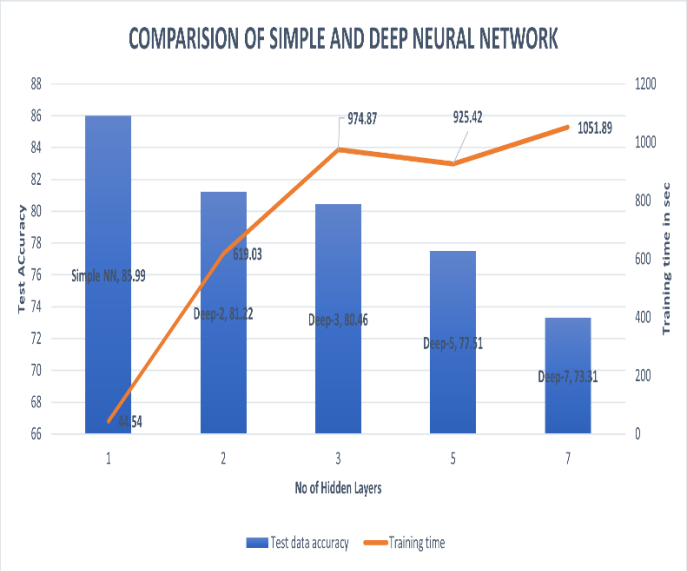
- Testing Data Accuracy is 85.99% at lambda value = 10
- Training time is 44.5 seconds

5: Comparison of Neural Network with a Deep Neural Network (using TensorFlow)

We compare and analyze the results in the celebA dataset between single layer Neural Network and the multilayer Deep Neural Network. We increase the number of layers from 2 to 7 (2, 3, 5, 7) and got the following results:

Type	No of Hidden layers	Test Data Accuracy	Training Time (in sec)
Simple NN	1	85.99	44.54
Deep NN	2	81.22	619.03
Deep NN	3	80.46	974.87
Deep NN	5	77.51	925.42
Deep NN	7	73.31	1051.89

The graph plot helps better understand and conclude the results as shown below:



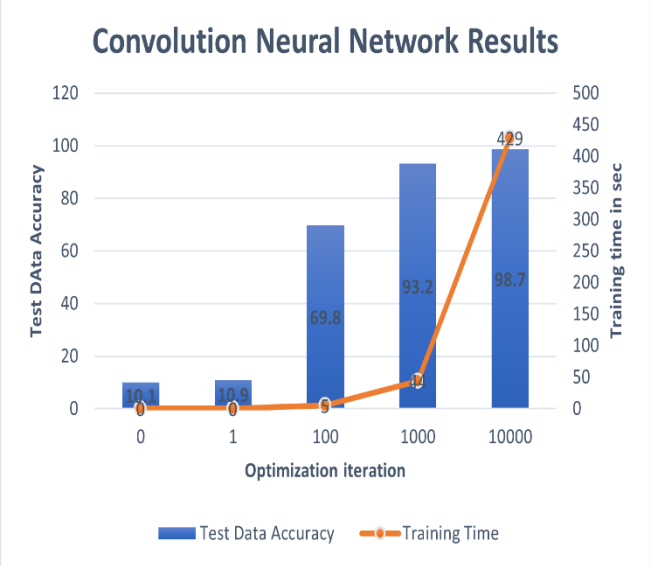
The plot above shows the Test accuracy and Training time for different number of hidden layers. It can be observed that highest test accuracy is obtained for simple neural network with just one layer. As the number of layers increase, accuracy decreases due to over-fitting. Also, training time is the lowest with Simple neural network. With deep networks, as the number of layers increase, training time increases linearly due to increased cost and computation. The summary of observations include:

- By increasing the no of hidden layers, training time increases due to increase in computation and complexity.
- Adding a greater number of hidden layers causes overfitting, thus resulting in lower accuracies.
- Simple neural networks do the work with better accuracy and less training time than Deep neural networks with multiple layers.

6: Results for Convolutional Neural Networks

We run the convolutional neural network on the given dataset and get the confusion matrix, our observations are as follows:

- We compare the accuracy of the CNN model, and we see that the accuracy is much better than the single layer neural network and multilayer deep neural network. **The accuracy of the CNN model is 98.7% and training time required is 429 seconds for approx. 10,000 iterations**.
- We infer that this is because of the way CNN configures the model. CNN changes the hyper parameters depending upon the requirement of the image and independence of prior knowledge in selecting the features.
- The primary reason for this accuracy is the use of filter matrix. As when create the filter by convolution we divide the input image into many parts of lower dimension and multiply with the filter of same dimension. So, we can change the image at every stage.



Confusion Matrix generated after approximately 10,000 iterations is:

```
[[ 975    0    1    0    0    1    0    0    2    1]
 [   0 1132    1    0    0    0    0    2    0    0]
 [   1    4 1019    2    1    0    0    2    3    0]
 [   1    0    1 1002    0    3    0    1    2    0]
 [   0    0    1    0  974    0    0    0    1    6]
 [   2    0    0    4    0  885    1    0    0    0]
 [   8    3    0    1    2   15  929    0    0    0]
 [   0    3    6    2    0    0    0 1011    1    5]
 [   4    0    6    6    1    2    0    2  949    4]
 [   1    3    0    2    3    6    0    2    0  992]]
```

Matrix plot of the confusion matrix: