CSE 574 INTRODUCTION TO MACHINE LEARNING

MARCH 6, 2017

PA1: HANDWRITTEN DIGITS CLASSIFICATION

SUBMITTED BY: ABHISHEK YEOLA

ROSHAN BABY REJI

CSE574 Machine Learning Programming Assignment 1

Group 6
Abhishek Yeola
Roshan Baby Reji

March 6, 2017

Contents

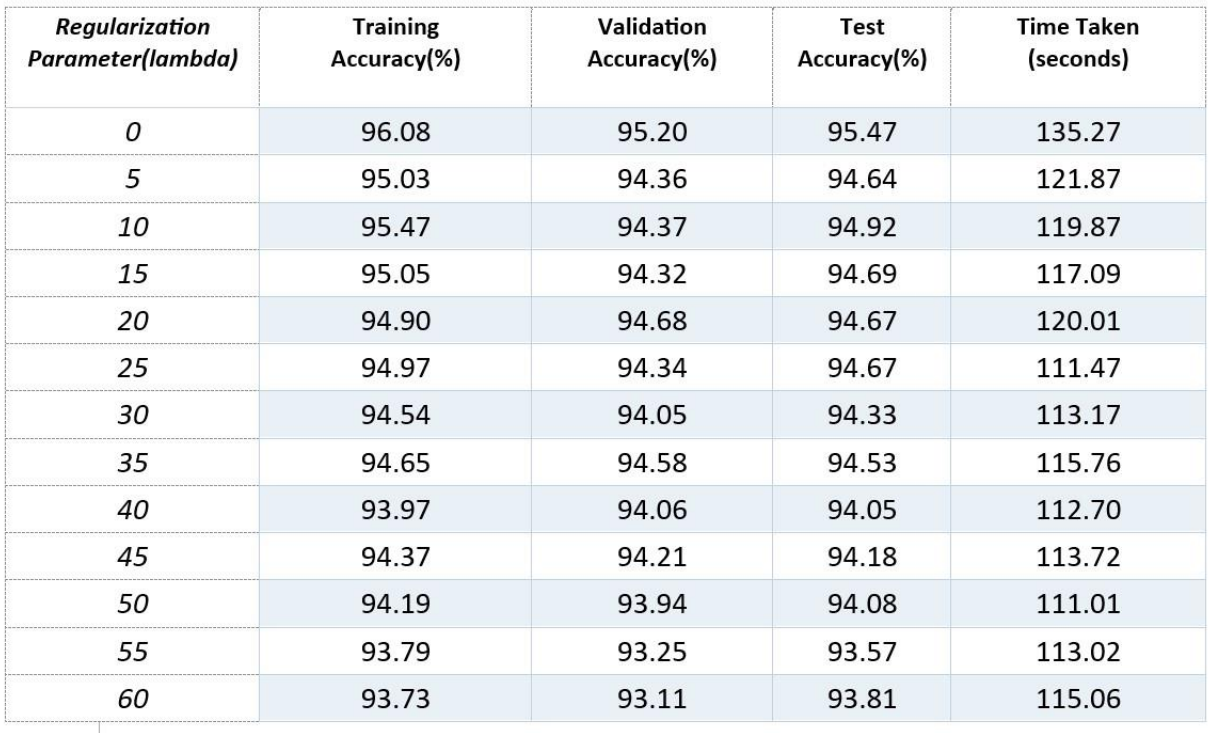
1 Introduction

In this project, a single layer Perceptron Neural Network was implemented to classify handwritten digits from 0 to 9. A negative log likelihood error function was minimized using the Conjugate Gradient method to learn the concept on MNIST data set. During feature selection, a total of 67 features out of 784 were removed. An optimum value of regularization parameter was estimated.

2 Handwritten Digits Classification

2.1 Finding Optimal Value of Lambda

Parameters
Number of hidden units: 50
Range of the regularization parameter (λ): 0 − 60 Step
Size: 5



Regularization Parameter(lambda)	Training Accuracy(%)	Validation Accuracy(%)	Test Accuracy(%)	Time Taken (seconds)
0	96.08	95.20	95.47	135.27
5	95.03	94.36	94.64	121.87
10	95.47	94.37	94.92	119.87
15	95.05	94.32	94.69	117.09
20	94.90	94.68	94.67	120.01
25	94.97	94.34	94.67	111.47
30	94.54	94.05	94.33	113.17
35	94.65	94.58	94.53	115.76
40	93.97	94.06	94.05	112.70
45	94.37	94.21	94.18	113.72
50	94.19	93.94	94.08	111.01
55	93.79	93.25	93.57	113.02
60	93.73	93.11	93.81	115.06

figure (1) : *Effect of* λ *on accuracy and time*

Most of the time the cause of the poor performance in a machine learning algorithm is either overfitting or underfitting of the data. A good machine learning algorithm generalizes well on training data and gives accurate prediction on new unseen data. In case of overfitting, the algorithm learns the concept too well on the training data set, which results in an increase in the error on the test data. Underfitting happens when the model performs poorly on the training data and it learns a more generalized concept. This also leads to decrease of prediction accuracy on the test data.

A regularization factor is used to add a penalty term to the objective function and control the complexity of the concept learned. A proper selection of the regularization parameter (λ) helps reduce the problem of overfitting. A popular technique to evaluate overfitting in a machine learning algorithm is to use validation data set, which comprises of randomly selected subset of the training data set. After the algorithm has finished learning the concept we can evaluate it on the validation data set to get an idea of how the model will perform on an unseen test data.

We have to choose a regularization parameter (λ) which leads to the algorithm performing well on the training, validation and test data sets.

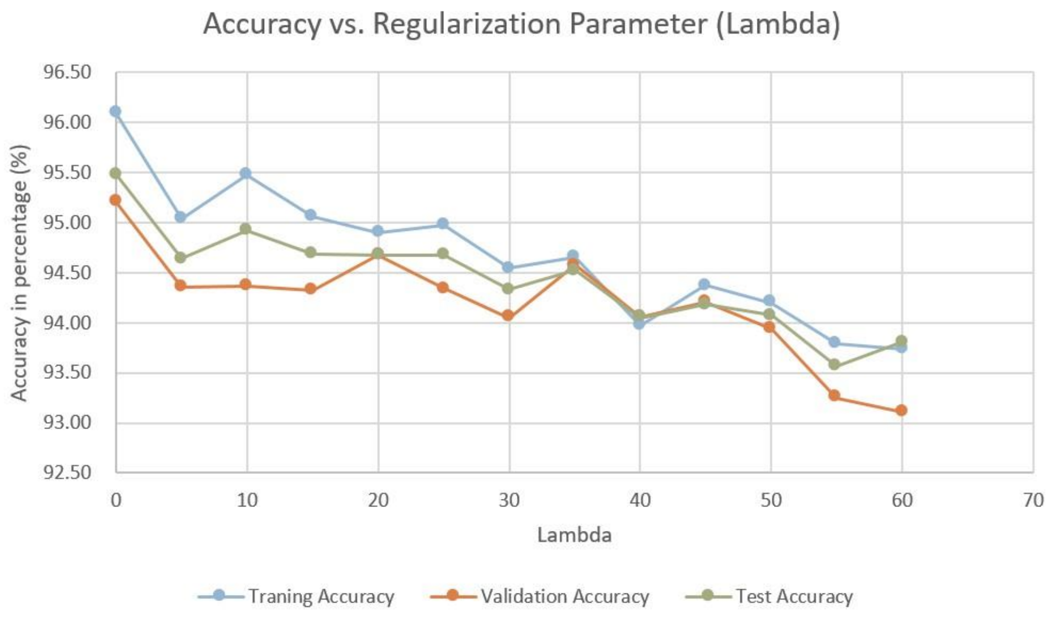


figure (2) : *Plot of λ vs. accuracy*

From figure(2) we can observe that in the region surrounding λ = 10, the accuracy of the training data set is very high. But the accuracy on the validation data set is comparatively very low. This is a typical example of overfitting problem.

At λ = 40, the performance on the training data set is poor while that on validation and test data set are comparatively more accurate. This is a typical example of underfitting problem.

Therefore we choose a value of λ in between these two points on which the performance is good on all three data sets. In this project, we have chosen λ = 35 as the optimum regularization parameter.

2.2 Varying the number of hidden units

Parameters

Regularization parameter (λ): 35
Range of hidden units: 4 – 80
Step size: 4

We keep the regularization parameter constant and increase the number of hidden nodes and check its effect on accuracy and execution time.

Number of Hidden Nodes (n_hidden)	Training Accuracy (%)	Validation Accuracy (%)	Test Accuracy (%)	Time (seconds)
4	69.356	68.43	69.7	34.57
8	88.47	88.53	88.26	45.273
12	91.65	91.29	91.08	50.328
16	92.7	92.05	92.63	57.7797
20	93.132	92.58	93.07	68.467
32	94.016	93.25	93.7	83.748
40	94.536	93.91	94.4	97.515
48	94.804	93.83	94.62	108.412
60	94.912	94.15	94.73	126.0498
72	94.932	94.15	94.78	142.909
80	94.77	94.19	94.35	151.606

figure (3) : *Effect of hidden nodes on accuracy and time*

We can see from figure(3) that as the number of hidden nodes are increased, the Machine Learning algorithm gives better accuracy over the data and after a limit it saturates.

Also, as the number of hidden nodes is increased, the time taken for execution also increases.

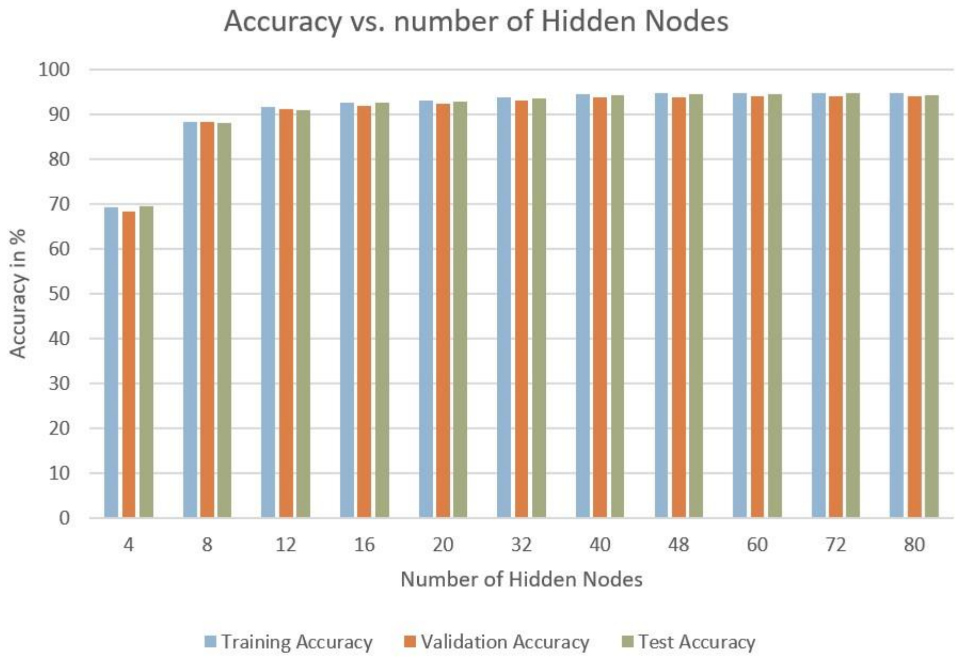


figure (4) : *Plot of hidden nodes vs. accuracy*

From figure(4), we can observe that the maximum accuracy is achieved when the number of hidden nodes is around 50.

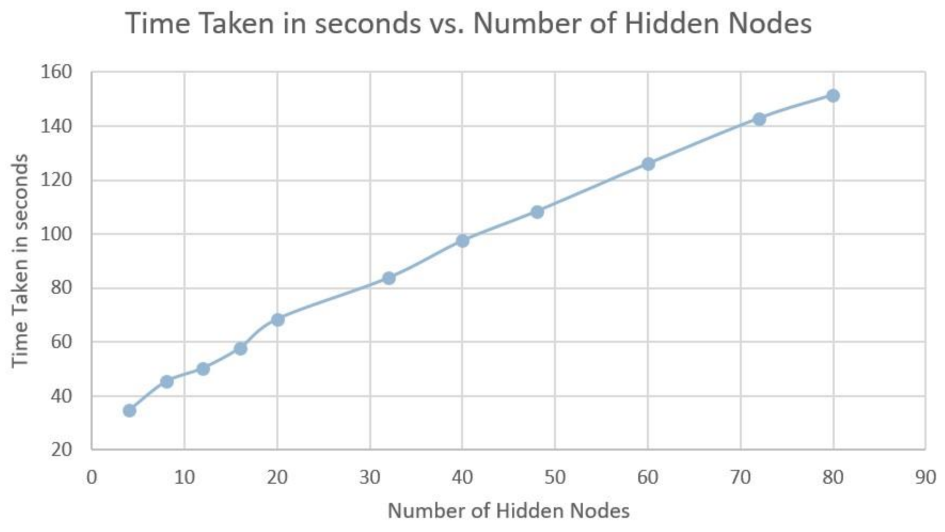


figure (5) : *Plot of hidden nodes vs. time*

We can see that as we increase number of hidden nodes, the time taken by the model to learn the concept increases. This is expected behavior, as the number of computations increase with an increase in the number of hidden nodes.

3 Deep Neural Network

3.1 Image classification on CelebA data set on single layer

Implement a single layer neural network.
Distinguish between two classes: Wearing glasses and not wearing glasses Use facennScript.py to obtain these results.

We have achieved an accuracy of 85.81% on the test data set after running the the facennScript.py on the CelebA data set.

The time taken was 32.53*sec*

3.2 Varying the number of hidden layers using TensorFlow

Try 3, 5, and 7 hidden layers on CelebA data set. The number of hidden nodes was kept constant at 256 Use deepnnScript.py to obtain these results.

Number of hidden layers	Test Accuracy in %	Time Taken in seconds
2	80.3936	145.7308
3	77.6684	152.1816
5	75.3974	167.2792
7	72.8615	189.9058

figure (6) : *Effect of hidden layers on test accuracy and time*

We can see that as the number of hidden layers is increased, the accuracy on test data decreases.

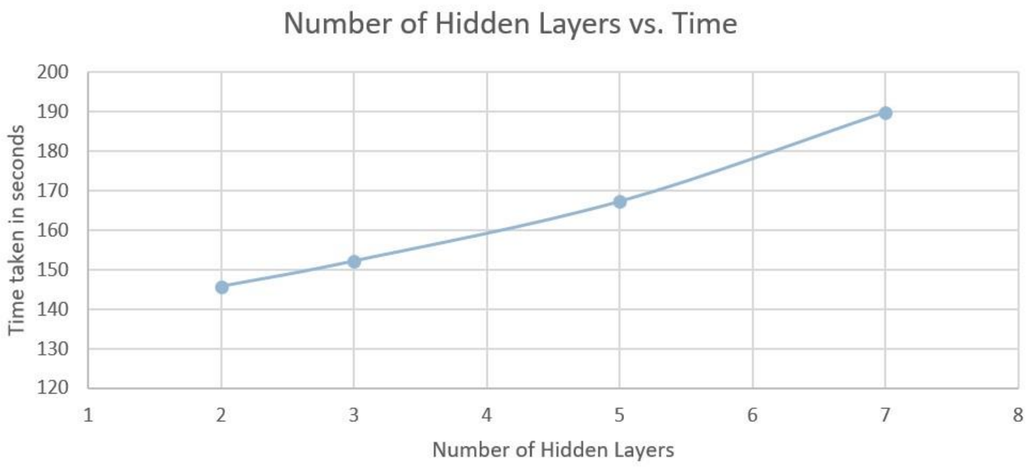


figure (7) : *Plot of hidden layers vs. time*

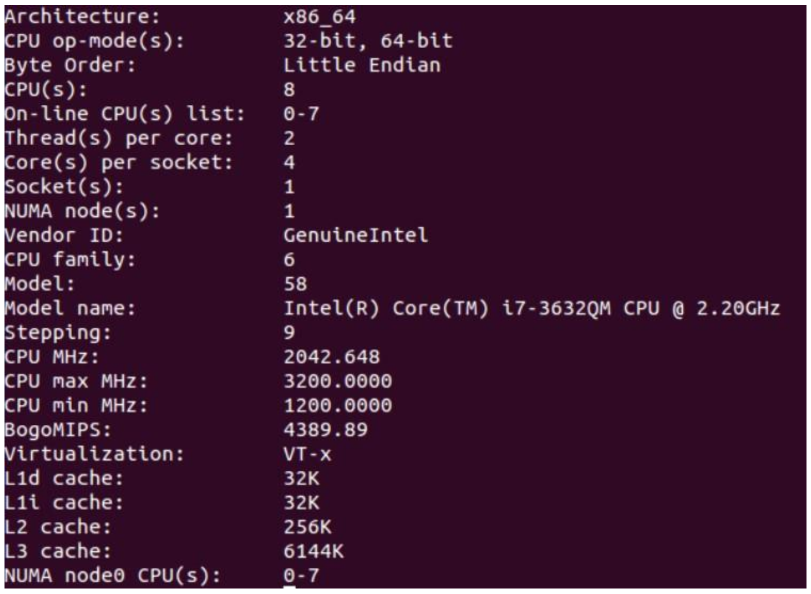
We can see that as the number of hidden layers is increased, the execution time increases. This is expected behavior as the number of computations increases drastically.

3.3 Comparison between single layer and multilayer neural network

We can see from the results of single layer neural network(facennScript) and multilayer neural network (deepnnScript) that the accuracy on test data decreases from 85.81% to 80.39% as we add one more hidden layer. On the other hand, the time taken for execution increases. The accuracy drops further as we increase the number of hidden layers.

4 Hardware and Software Specifications

All the scripts were run using **python 3.5** on Ubuntu 16.04



```
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                8
On-line CPU(s) list:   0-7
Thread(s) per core:    2
Core(s) per socket:    4
Socket(s):             1
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 58
Model name:            Intel(R) Core(TM) i7-3632QM CPU @ 2.20GHz
Stepping:              9
CPU MHz:               2042.648
CPU max MHz:           3200.0000
CPU min MHz:           1200.0000
BogoMIPS:              4389.89
Virtualization:        VT-x
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              6144K
NUMA node0 CPU(s):     0-7
```