Programming Assignment-2 Report

Group number 22

Course number 574

Members Meghana Raj Kalidindi(50385493)

Prashant Jitendra Kalyani(50388408)

Venkata Naga Pramod Bikkina (50366406)

Neural-networks

While we can use 'basis function expansion' or 'kernel methods' for building a non-linear modelling, in real life scenarios it is very difficult to choose what functions to use for either of these methods. On the other hand, Neural networks captures the non-linear relationship between Input data and the output labels naturally, and thus they are very powerful.

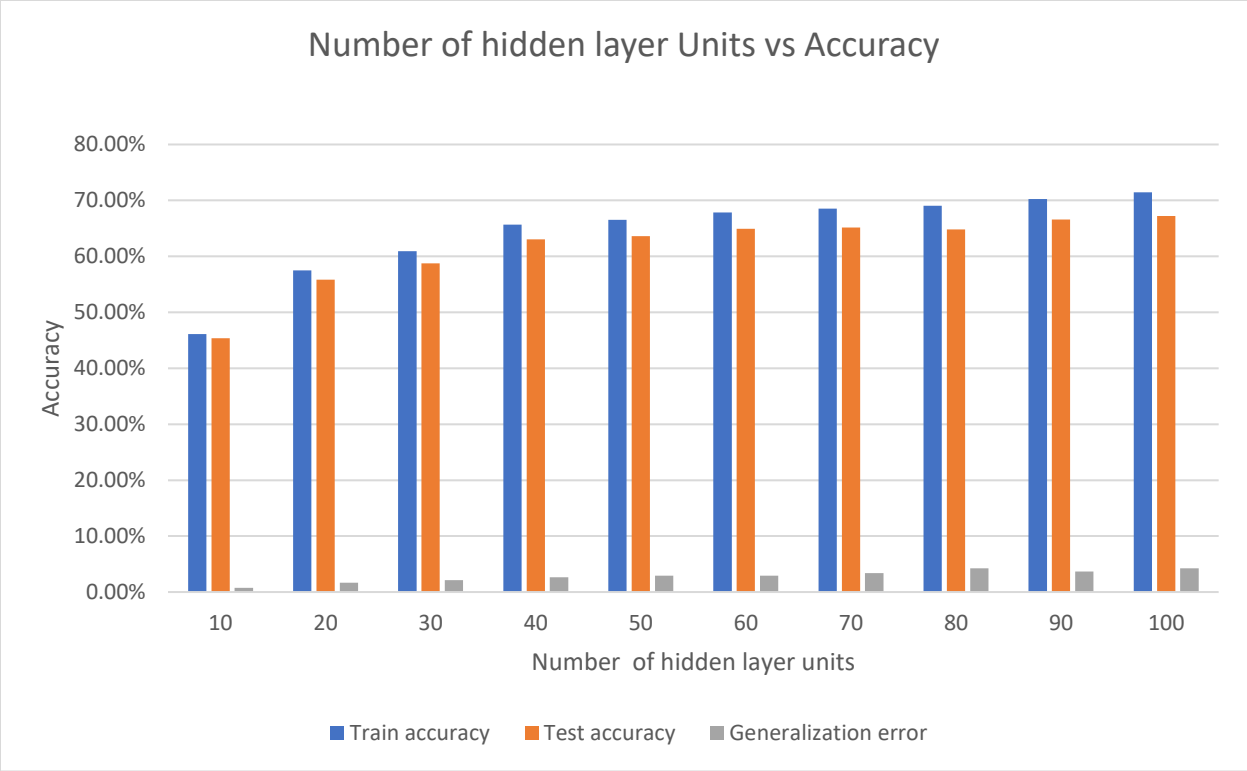
Report-1

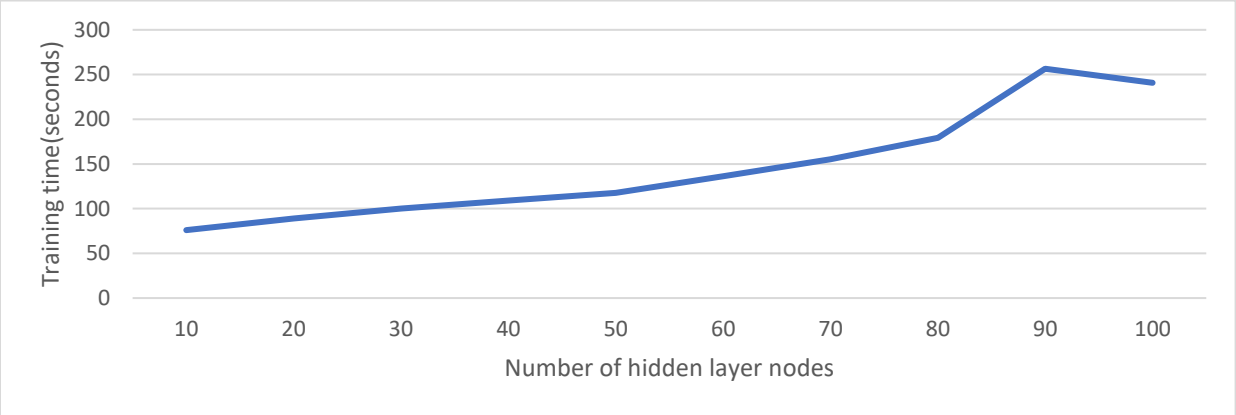
1. the evaluation of the implemented neural network with hidden layer having 50 nodes is as below (for this specific run).

Train accuracy	**65.16%**
Test accuracy	**63.04%**
runtime	**110.81 seconds**

2. Below are the test and train accuracies for various number of hidden layer units ranging from 10 to 100 with a step value of 10.

Number of Nodes	Train accuracy	Test accuracy	Generalization error
10	46.09%	45.36%	0.73%
20	57.50%	55.84%	1.66%
30	60.90%	58.76%	2.14%
40	65.68%	63.04%	2.64%
50	66.55%	63.62%	2.93%
60	67.87%	64.95%	2.92%
70	68.52%	65.14%	3.38%
80	69.02%	64.79%	4.23%
90	70.26%	66.60%	3.66%
100	71.45%	67.19%	4.26%





Observation:

a) From the plot, it can be observed that the optimal number of hidden layer units(**M)** is **100** as both test and train accuracies are highest when compared with rest of the values.

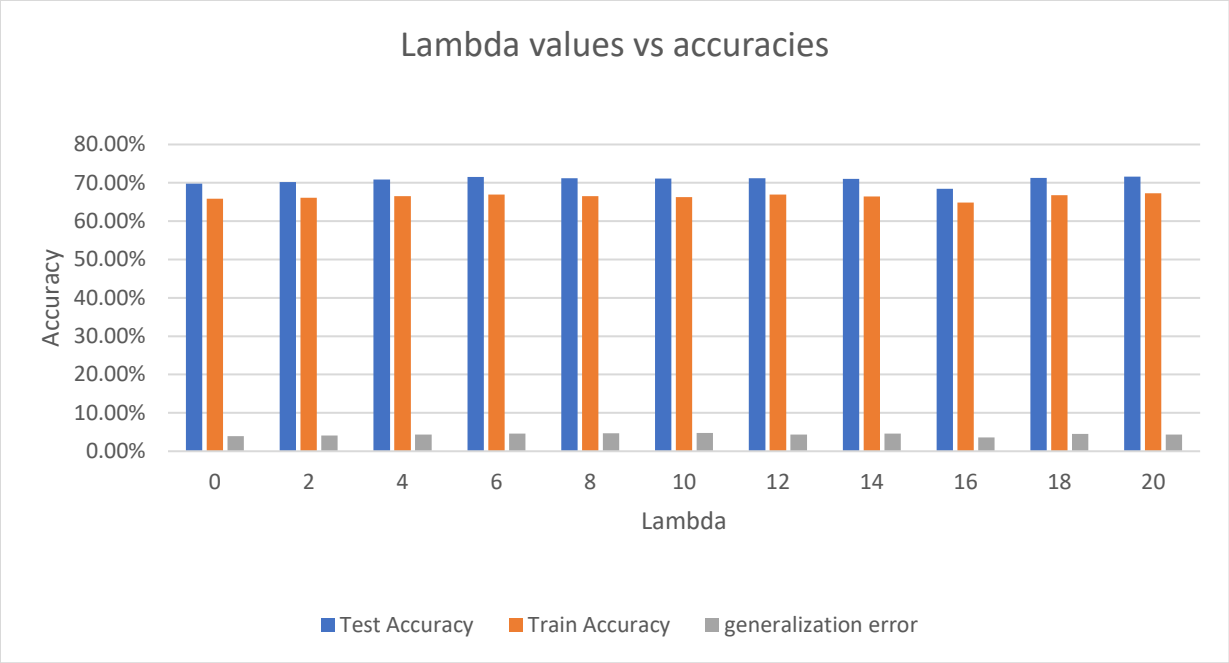
Optimal value of M	**100**

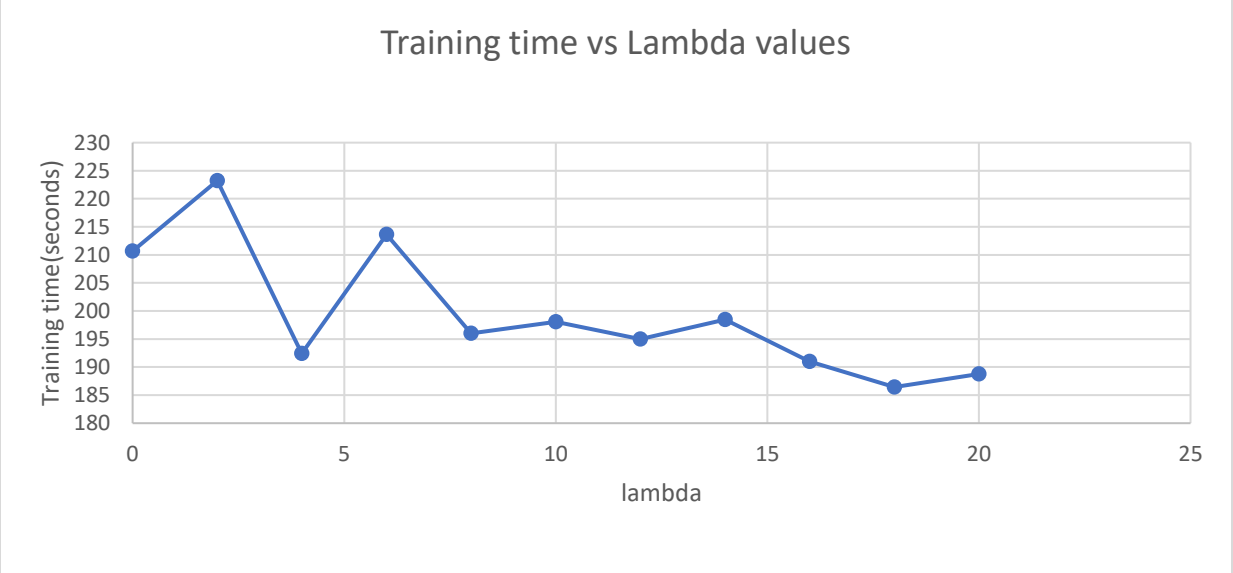
b) It is worth to mention that, for M=90, also has similar accuracies as M=100 with less generalization error. Since the difference in generalization error is not significant, M=100 is chosen as an optimal value. *In case if the accuracies for any two models are same, it is better to choose a simpler model to avoid over-fitting*.

*since the initial weights are selected by using NumPy random function, the values above may vary during each run due to the fact that while finding the optimal weights, gradient descent may end up at different local optimum based on the initial weights. However, a similar trend has been observed on multiple runs.

3.The train and test accuracies for different values of lambda are shown below along with generalization error where number of hidden layer units **M =100**.

Lambda	Test Accuracy	Train Accuracy	generalization error
0	69.78%	65.82%	3.96%
2	70.21%	66.10%	4.11%
4	70.90%	66.55%	4.35%
6	71.56%	66.93%	4.63%
8	71.18%	66.52%	4.66%
10	71.10%	66.31%	4.79%
12	71.21%	66.90%	4.31%
14	71.06%	66.45%	4.61%
16	68.48%	64.87%	3.61%
18	71.24%	66.75%	4.49%
20	71.62%	67.31%	4.31%





Observation: for this particular run, Lambda=20 gave the highest test and train accuracy with a generalization error of 4.31%. For this reason, it is chosen as an optimal value. For different set of runs, optimal lambda value is varying. We observed Lambda values {6,18,20} as optimal for multiple runs where optimal lambda to be 20 is observed in most of the cases.

Optimal value of Lambda	**20**

4. For optimal values, M=100 and lambda=20, **We observed that our model makes more mistakes on the class 'arm'.** On Visual inspection, we understand that model is confusing between the classes **'banana'** and **'arm'** because of the similarity in the drawings.

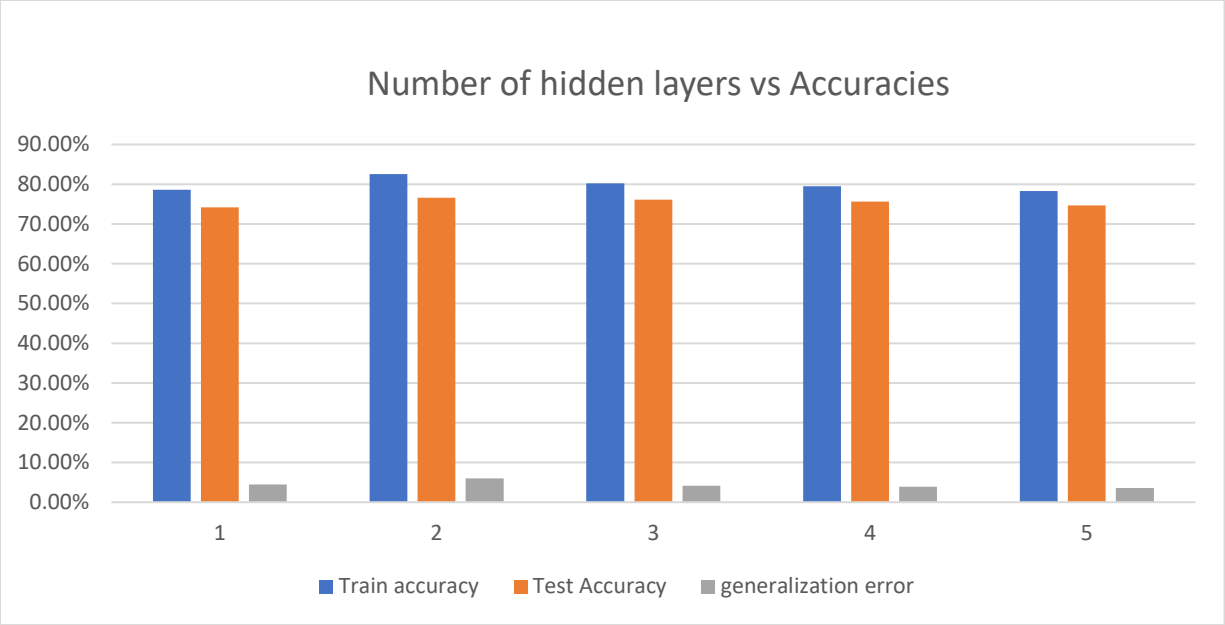
We can further improve the performance of our model by the following:

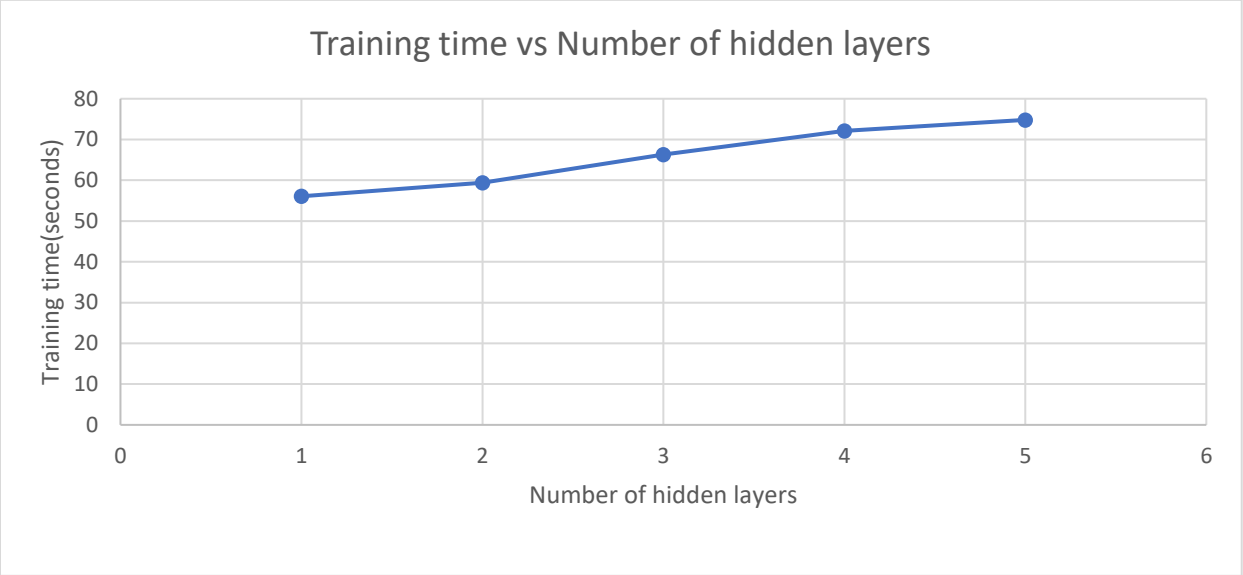
- ➔ Increasing the number of hidden layers
- ➔ Changing the activation function at either hidden layer or at output layer or both
- ➔ Using a different architecture for neural network instead of multi-layer perceptron

Report-2

1.Fixing the number of hidden layer units at 100, below are the accuracies we got by varying the number of hidden layers from 1 to 5

Number of hidden layers(L)	Train accuracy	Test Accuracy	Training time	generalization error
1	78.64%	74.22%	56.07 seconds	4.42%
2	82.55%	76.59%	59.41 seconds	5.96%
3	80.24%	76.08%	66.29 seconds	4.16%
4	79.46%	75.60%	72.10 seconds	3.86%
5	78.28%	74.69%	74.81 seconds	3.59%





Observation: It is observed that for L=2, the test and train accuracies are the highest as well with a less training time. So, Number of layers equals 2 looks like an optimal value

Optimal number of hidden layers	**2**

2.

activation function	Train accuracy	test accuracy
Sigmoid	82.55%	76.59%
Tanh	79.92%	76.05%
Relu	74.71%	63.78%

The best activation function to use for this problem is ”**Sigmoid”**