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 model, 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.

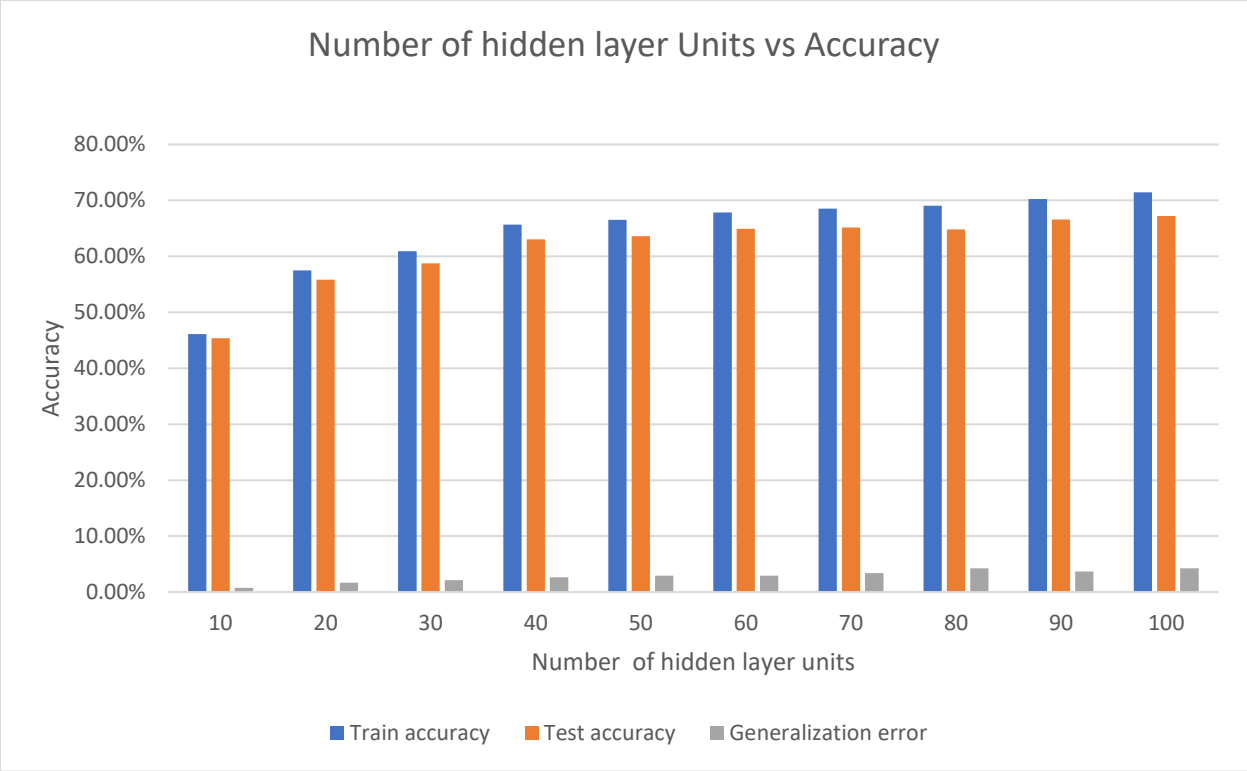
Report-1

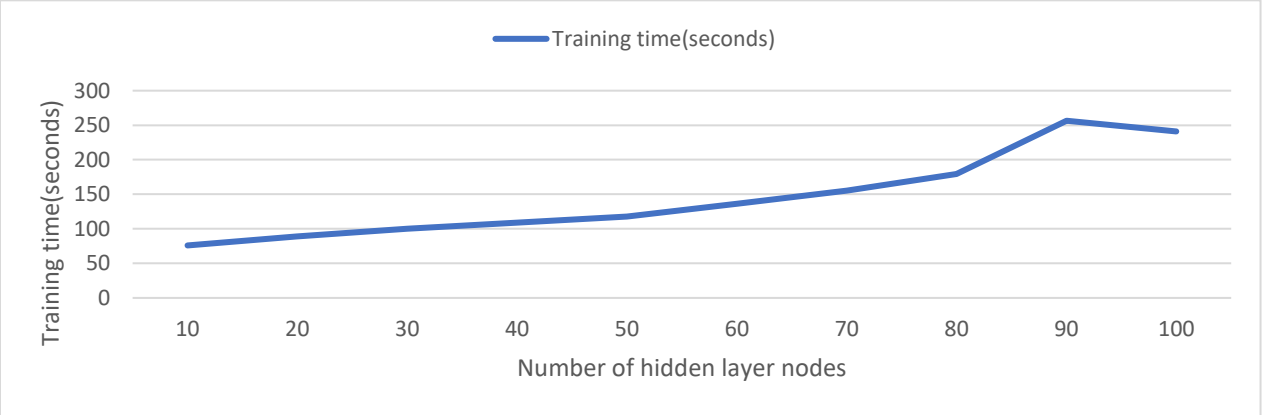
1. the evaluation of the implemented neural network with hidden layer having 50 nodes is as below (for this specific run).

Train accuracy	66.55%
Test accuracy	63.62%
Training time	136.18 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%





Analysis:

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.

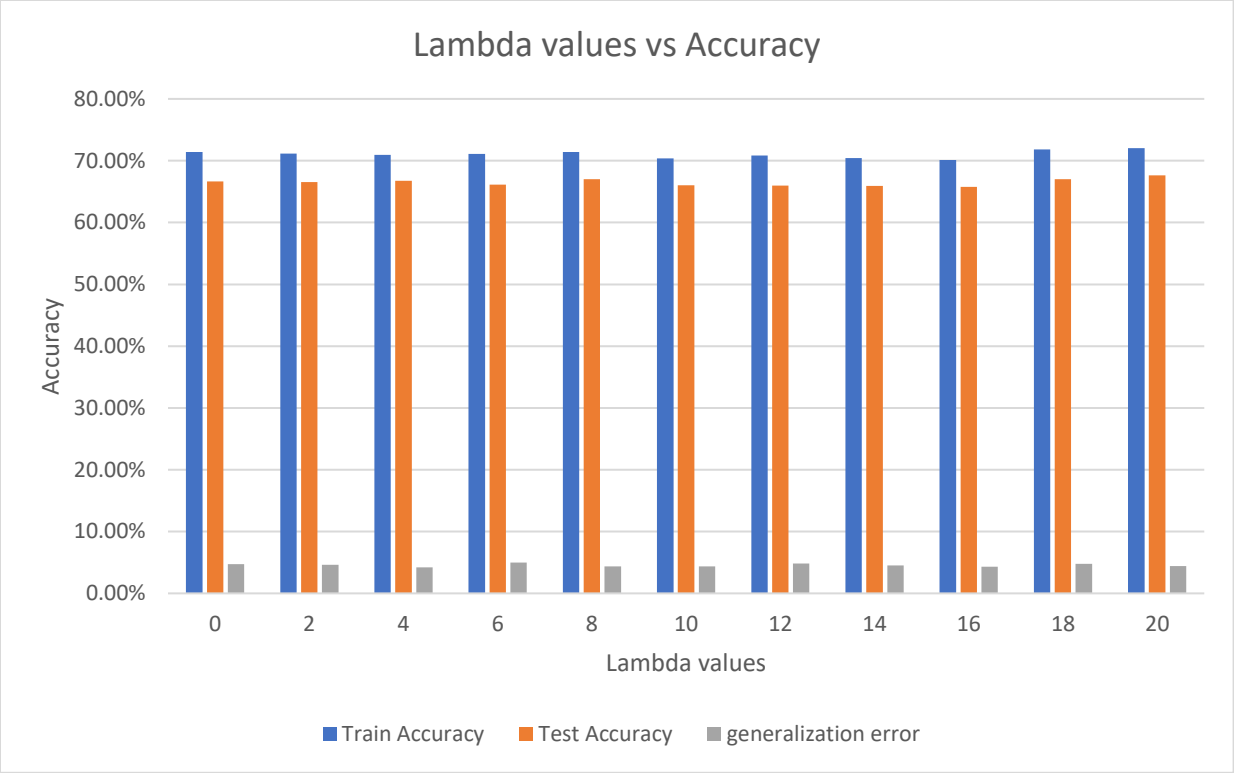
Optimal value of M	**100**

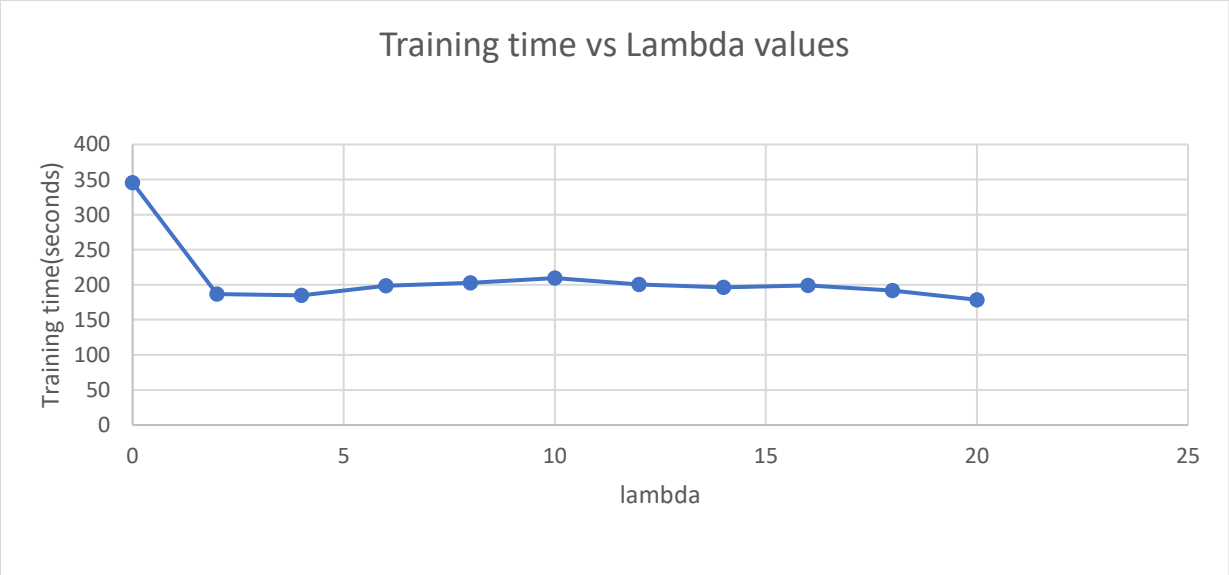
b) It is worth to mention that, for M=90, also has similar accuracies as M=100 with less generalization error. Since the difference in generalization error is not significant, M=100 is chosen as an optimal value. *In case if the accuracies for any two models are same, it is better to choose a simpler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

Note: 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	Train Accuracy	Test Accuracy	generalization error
0	71.41%	66.67%	4.74%
2	71.14%	66.52%	4.62%
4	70.93%	66.73%	4.20%
6	71.10%	66.13%	4.97%
8	71.40%	67.02%	4.38%
10	70.40%	66.03%	4.37%
12	70.83%	65.99%	4.84%
14	70.43%	65.92%	4.51%
16	70.10%	65.77%	4.33%
18	71.80%	67.03%	4.77%
20	72.02%	67.61%	4.41%





Observation: for this particular run, Lambda=20 gave the highest test and train accuracy with a generalization error of 4.41%. For this reason, **Lambda =20 is chosen as an optimal value**.

Reason: For this problem, we just used accuracy as a metric to decide the optimal parameters hence we chose Lambda=20 as optimal. However, in real life situations,

along with accuracy we might have used required computational resources as metric and thereby selected L=8/18 as optimal as these values also produce similar accuracy but require lesser computational power.

4. For optimal values, M=100 and lambda=20, **We observed that our model makes most number mistakes on the class '<u>arm</u>'.** Below are the number of mistakes the selected model is making on each class (in the descending order, for a specific run)

Class	**Number of mistakes**
arm	1731
ant	1321
alarm clock	931
horse	909
airplane	768
bed	709
banana	522
basketball	497
axe	471
apple	232

Number of mistakes chosen model makes on test data

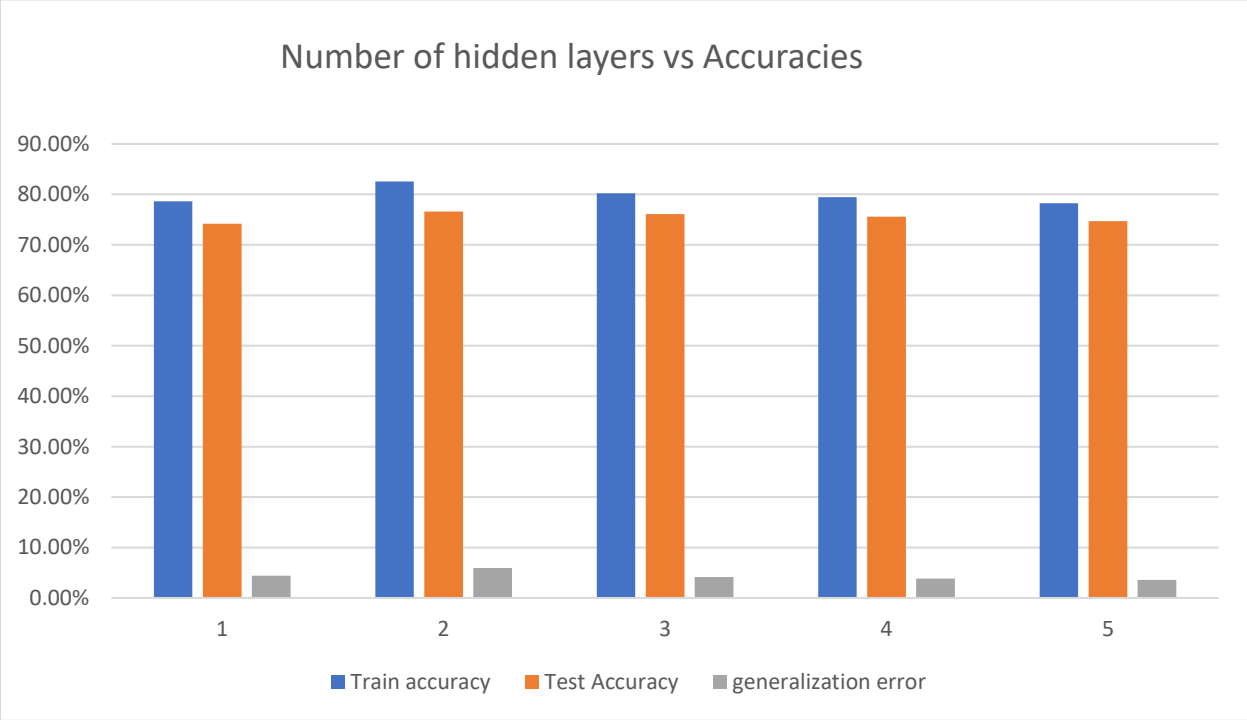
<u>We can further improve the performance of our model by the following</u>:

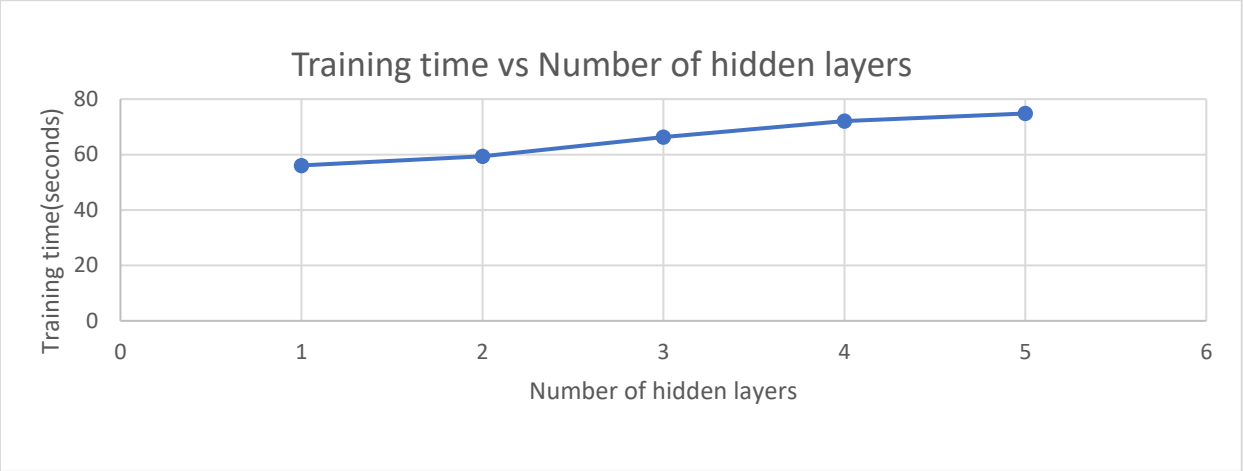
- ➔ Increasing the number of hidden layers
- ➔ Changing the activation function at either hidden layer or at output layer or both
- ➔ Using a different architecture for neural network instead of multi-layer perceptron
- ➔ By using more effective preprocessing methods on the Input data.

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**, test and train accuracies are the highest and hence it is decided that this value is optimal (for this specific run)

Reason: Just by increasing the number hidden layers the accuracy of the model may not increase as it is also depending on the complexity of the problem. Also, we might run into the problem of over-fitting by increasing the number of hidden layers. In this particular case, optimal accuracies are obtained at L=2 and for higher L values, accuracy is shown a downward trend. Hence, we chose L=2 as optimal value.

2.

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

For this particular problem, "**Sigmoid"** is the best activation function as the model able to achieve the highest accuracy with this function.
