Programming Assignment 1 Report – Handwritten Digits Classification

Subject: CSE574 (Machine Learning)

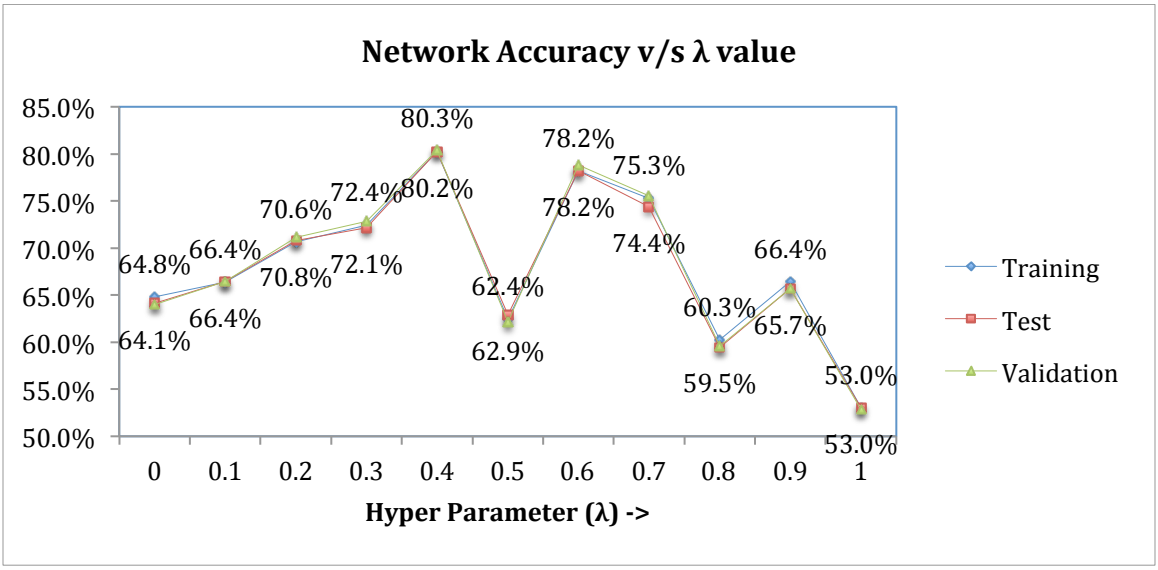
Team Member 1: Ashish Gupta

Team Member 2: Girish Suri Venkataraman

Team Member 3: Sanjay Natraj Ashok

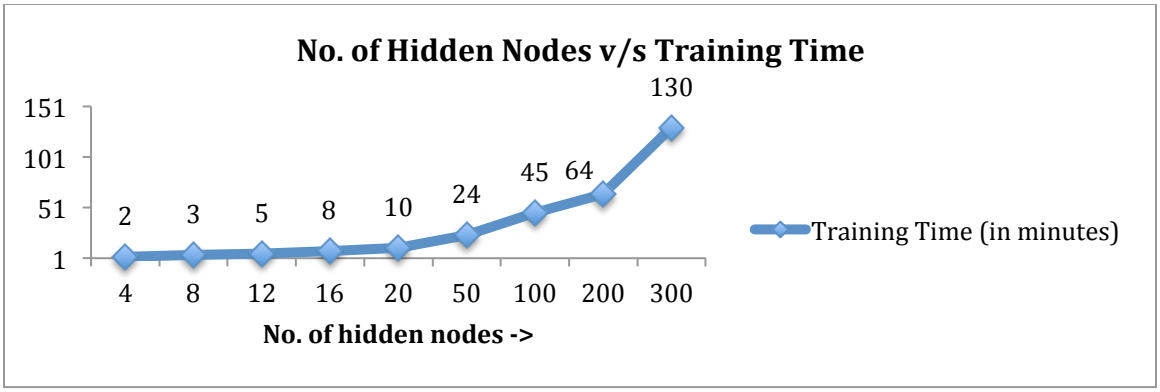
Group Number: Group 14

1. Effect of regularization coefficient (λ) on the performance of the Neural Network



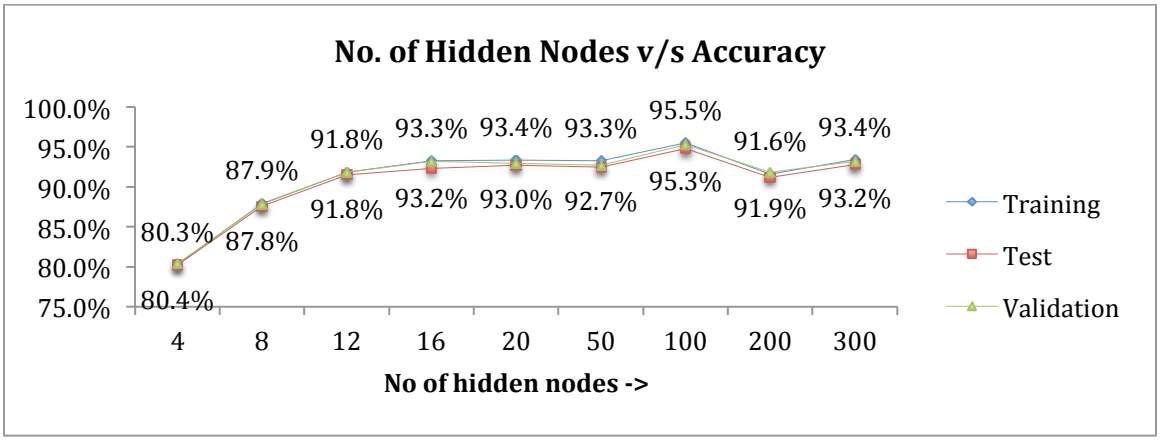
In general, we find that there is not much problem of over-fitting or under-fitting in our dataset as the accuracy is relatively the same throughout (between training, validation and test data). However, what we did find is that increasing the regularization coefficient to 0.4 gives the best accuracy and also has almost complete convergence between training and test data. **MAX_ITER** was set to **50**.

2. Effect of number hidden nodes on training time of the Neural Network



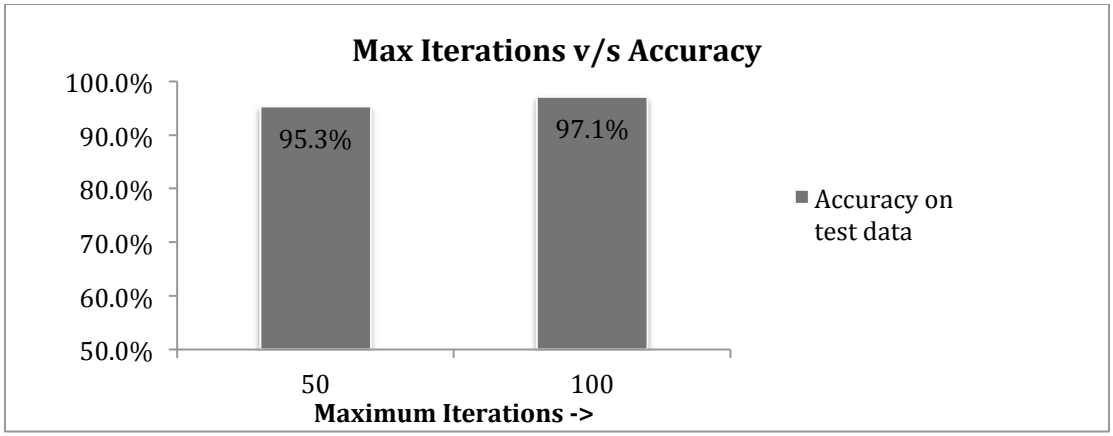
As you can see from the graph, more hidden nodes implies more internal calculations are to be performed and hence the time taken also increases. For 4 hidden nodes, it takes 2 minutes to train. This time was optimized from 1.5 days to 2 minutes through the clever use of python numpy's inbuilt matrix multiplications and dot product function (* and np.dot respectively)!! Also, generally doubling the number of nodes, doubles the amount of time taken to train the network. **MAX_ITER** was set to **50**.

3. Effect of number of hidden nodes on performance of Neural network



All the results are for the λ = 0.4 (optimum value for lambda). We found that until 100 hidden nodes, the accuracy keeps on increasing after which it starts dropping. This implies that for our model, a hidden nodes number of around 100 produces the best results. The decision to increase the number of hidden nodes was made based on the result of the validation dataset. **MAX_ITER** was set to **50**

4. Effect of number of maximum iterations (conjugate gradiant descent) on the performance of the Network (for no. of hidden nodes = 100, λ = 0.4)



Increasing the maximum iterations that the minimize function takes helps produce better accuracy. We found this after testing on **100 max iterations** v/s 50 iterations while keeping the number of hidden nodes as 100 and a λ of 0.4. This gave us a best accuracy of **97.1%** on **test data** and **98.3%** on our **training data** on trying to classify digits.