Programming Assignment 1
Handwritten Digits Classification

Mohamed Ismail, Nikhil Pillai, Krithika Krishnan
CSE 574; Group 24

The purpose of this assignment was to classify Handwritten Digits by implementing a Multilayer Perceptron Neural Network. The process involved the following tasks,

- Implement a three layer (one hidden layer) Neural Network (forward and back propagation)
- Incorporation of regularization constant on the weights, λ
- Tune hyper-parameters for Neural Network (number of units in the hidden layer and λ)

The same network was used to analyze a more complex face dataset so as to gauge the performance of the constructed neural network against a deep neural network using the TensorFlow library.

MNIST Dataset

The MNIST dataset contains 60,000 training examples of which 10,000 examples were used for testing. The format of the MNIST database is such that the first value is the "label" of the digit the handwriting is representative of. The remaining values are the pixel values of the handwritten digit. Size of pixel array is 28 x 28 after the provided digits have already been normalized and centered.

Choosing the hyper-parameters for Neural Network:

Regularization in Neural Network: The learning model built fits well with the training data set. However, it is unlikely to generalize well with new data(validation data). Therefore, to overcome this problem of over-fitting, regularization can be applied to avoid complexity in the neural network even if it leads to a less accurate learning rule.

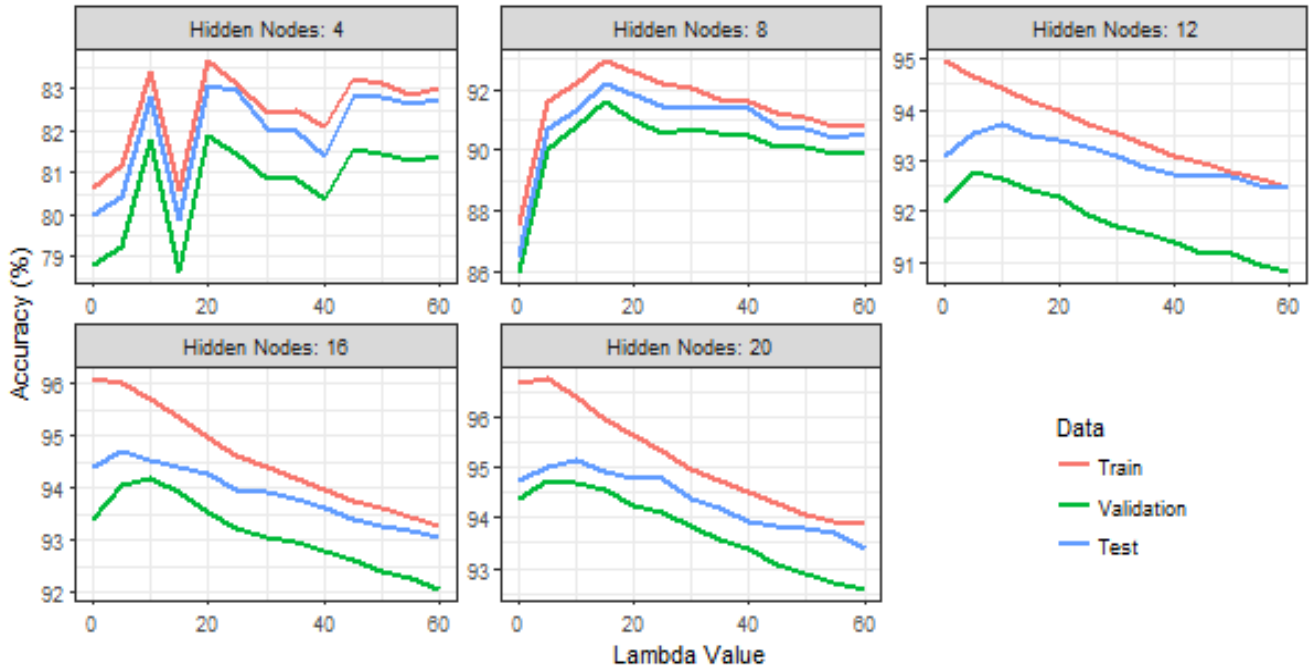

Figure 1: Accuracy expressed as a function of λ for constant values of hidden nodes(4,8,12,16,20)

Therefore, we add a regularization term λ into our error function to control the magnitude of parameters in the neural network. Regularization modifies the objective function by adding a term that

penalizes large weights. The value we choose for λ determines how much we want to protect against over-fitting. A λ value of 0 would mean that we are not taking any measure against the possibility of over-fitting while an extremely large value of λ might lead to an issue of under-fitting. It can be observed from Figure 1 that the accuracy first rises to a peak and after that it drops with increasing λ.

The plot in Figure 1 represents the effect of change in the hyper-parameter, λ on the accuracy of the Neural Network. λ was varied from 0 to 60 in increments of 5. The change was observed for fixed values of the number of hidden nodes. In order to obtain the optimal hyper-parameters, we chose a range of values for λ and number of hidden nodes and recorded the results. It can be concluded from the above shown plots that the validation accuracy, which is of interest to us, has a peak accuracy of 94.72%(fifth plot). This occurs with 20 nodes and a λ value of 5.

The positive effect of including λ in the model can strongly be seen at smaller number of nodes (4,8). As the number of nodes goes up, λ has a less positive impact on the model, however it does appear to slightly improve the model. To choose hyper parameters, (number of nodes, λ) a grid search method was implemented. All possible combinations of number of nodes (4, 8, 12, 16, 20) and λ (0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60) were tested, and the parameter combination that gave the best fit (in terms of accuracy on validation data) was chosen. As indicated in the plot below (Figure 2), the number of hidden units was gradually increased from 4 and raised until 20. It is clear from the graph that the accuracy is reaching a plateau with increase in hidden nodes and increasing number of nodes further is not likely to increase accuracy substantially.

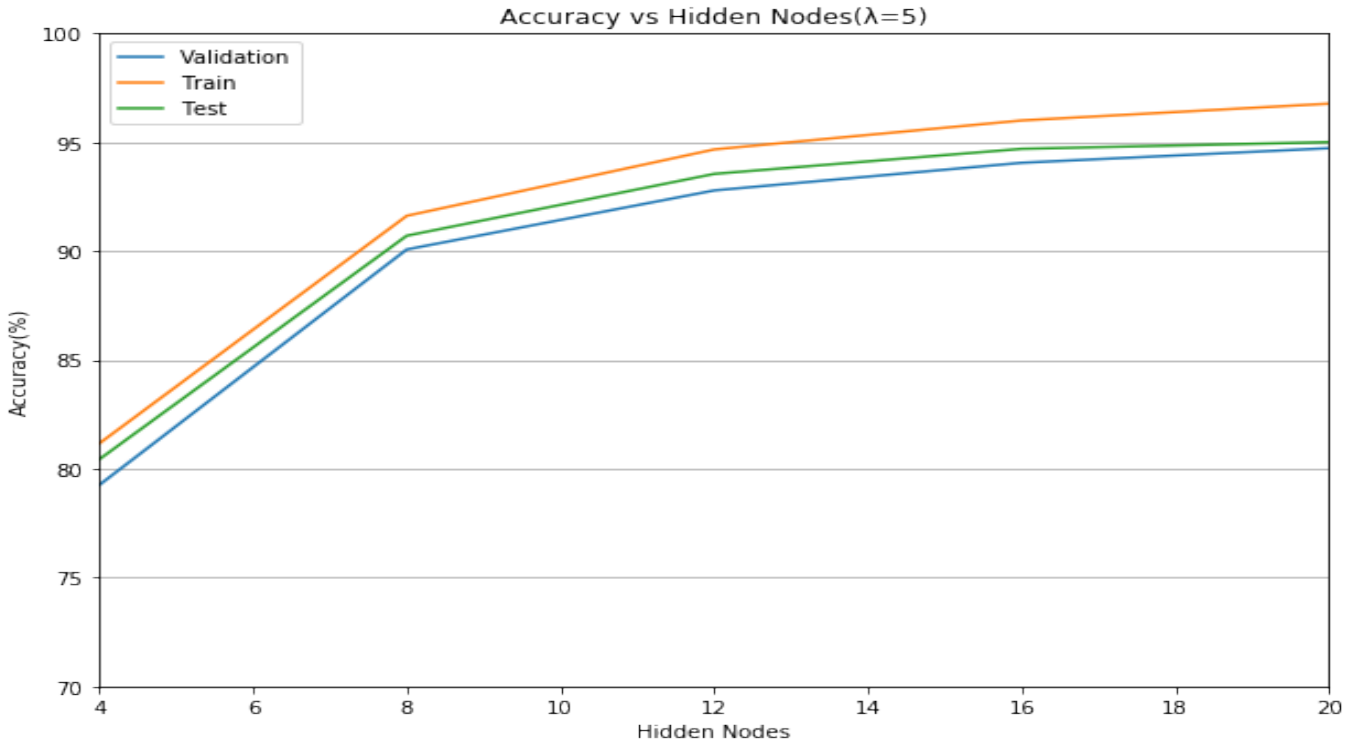

Figure 2: Accuracy rates expressed as function of Hidden Nodes

Relation between number of hidden units and training time: The number of hidden units was gradually increased to observe its effect on the performance of Neural Network. It can be observed from the plot(Figure 3) below that the training time increases linearly as a function of the number of hidden units. An accuracy of 94.72% was obtained in the validation dataset which is of particular interest to us. In the test data, 95% accuracy was obtained.

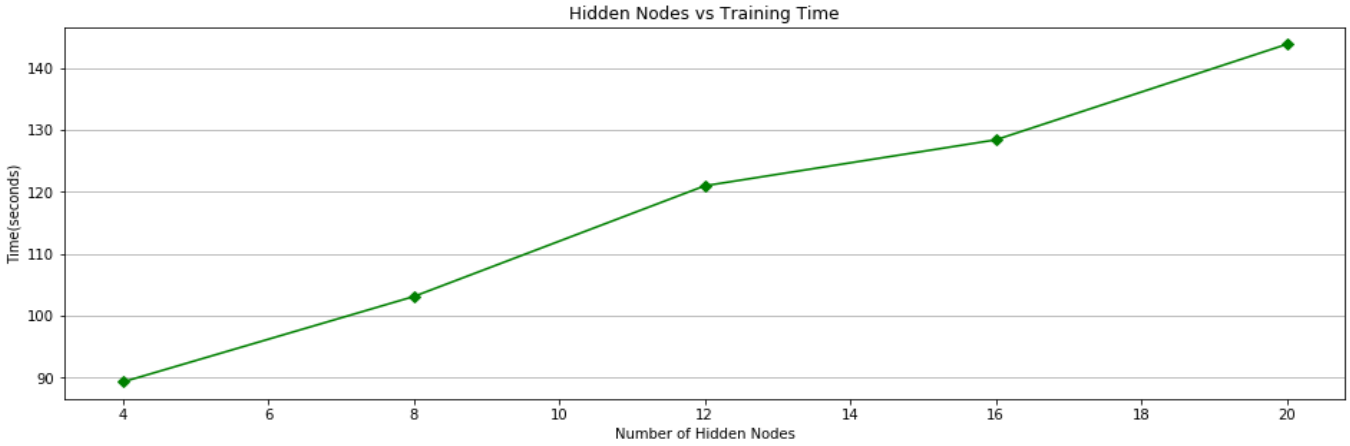

Figure 3: Training time expressed as a function of Hidden Nodes

CelebFaces Attributes Dataset (CelebA)

The task required to complete with the CelebA dataset is to analyze the improvement in performance of the Neural Network by adding more layers using Google's TensorFlow library.

The Validation Accuracy of a single hidden layer Neural Network on CelebA dataset was found to be 83.64% for 50 iterations with a learning time of 191.514 seconds. The training and test accuracies of the CelebA data set were 85.35% and 85.162% respectively. The objective for the neural network was to determine whether celebrities were wearing glasses or not. The file *facennScript.py* was used to run this model.

The accuracy of deep Neural Network for hidden layers, 2,3,5,7 was evaluated on CelebA dataset and it was noted as following, 80.05% for 2 hidden layers with a learning time of 171.598 seconds, 78.27% for 3 hidden layers with a learning time of 198.151 seconds, 75.13% for 5 hidden layers with a learning time of 221.392 seconds, and 72.03% for 7 hidden layers with a learning time of 248.329 seconds. This was performed by running *deepnnScript.py* on a system with 16.0 GB and 2.21GHz, i7 processor. The computational time increases with increase in hidden layers since now there are more parameters to optimize. The reason for a higher computational time in case of facenn script having 1 layer compared to 2 layer deepnn script can be accounted due to the modern libraries being used in the deep neural network script which works more efficiently.

Comparison of the performance of single vs. deep Neural Networks in terms of accuracy on test data and learning data:

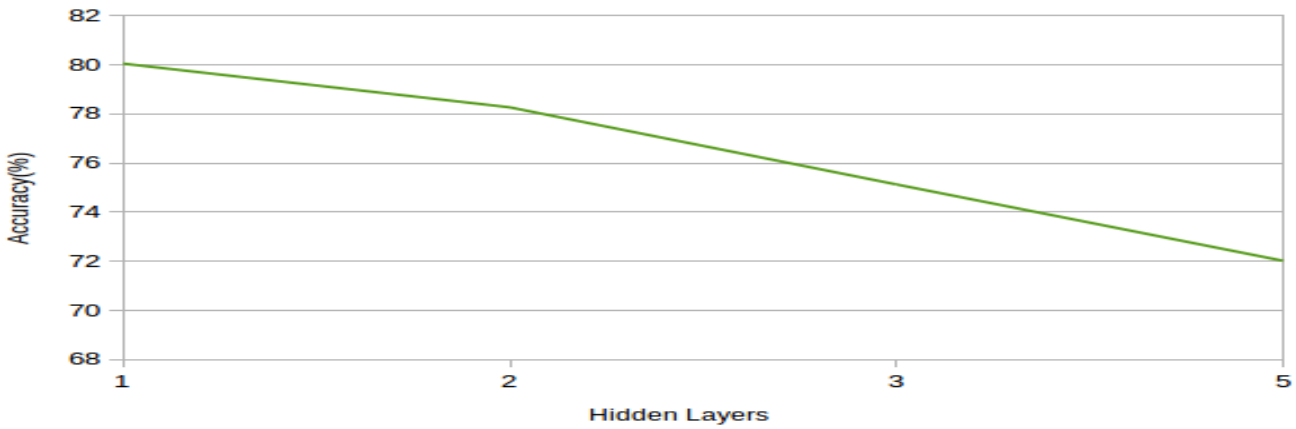

Figure 4: Accuracy of Deep Neural Network on CelebA dataset for different number of hidden layers

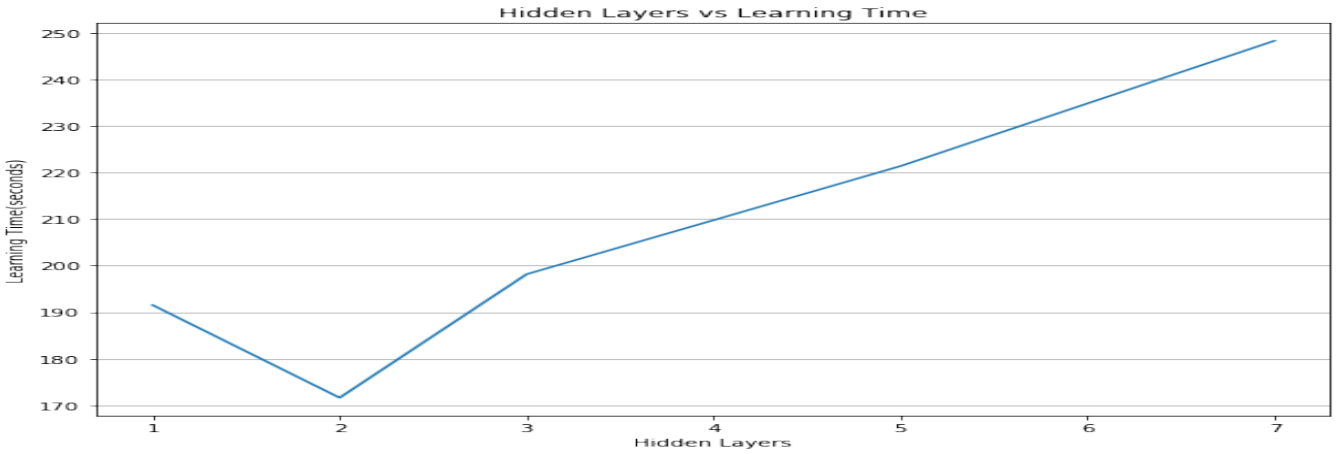

Figure 5: Deep Neural Networks in terms of Learning time

It is evident from the accuracy rates on the test data that as more layers are added in the deep Neural Network, it does not necessarily increase the accuracy rate. This can be accounted due to increase in complexity of the model (more parameters to optimize). As you increase the number of hidden layer, effectiveness of back propagation decreases; plus you need to deal with overfitting problem i.e. your network will perform very well on the training data set but it cannot generalize to new data it has not seen and gives poor performance on the new data. The performance of the single layer Neural Network is comparatively better and with a lower learning time than those with hidden layers consisting of 2, 3, 5 and 7. The learning time is observed to increase linearly with the addition in hidden layers due to increase in complexity of the model.