HANDWRITTEN DIGITS CLASSIFICATION

CSE 574 : INTRODUCTION TO MACHINE LEARNING

Programming Assignment 1

GROUP 29

Prashanth Seralathan(pseralat)

Shreya Ravi Kumar(sr264)

Shreya Ravi Hegde(shegde3)

INTRODUCTION

The first component of the assignment involved implementing a Multilayer Perceptron neural network and evaluating its performance in classifying handwritten digits. The second component involved analyzing a more complicated dataset that contains celebrity images with this Neural network and classify them to determine whether the given image wear glasses or not, finally a comparison is made on performance of single layer neural network against that of a deep Neural Network that uses the TensorFlow library.

OBSERVATIONS AND RESULTS

Choosing hyper-parameters for neural network (number of hidden units, regularization term λ) :

The neural network hyper parameters depends upon Learning Rate, Number of Training Iterations and Momentum.

Learning rate needs to be chosen carefully since a larger learning rate would cause the convergence to happen too soon and it would miss the actual minima and a slower learning rate would take a lot of time in training, hence it is necessary to choose an optimal learning Rate. Having a large number of training iterations ensures that there is sufficient iterations for the convergence to occur. Momentum involves a custom way of smoothing the gradient updates, harmonically decreasing learning rate is always considered optimal.

We conducted several experiments to determine how the hyper parameters influence the accuracy and training time of the neural network. In general choosing a hyper parameter can be done on the basis of random-search, grid search, Model based method, etc. In this programming assignment we try to vary the hyper-parameter and number of hidden nodes to study the consistency/ improvements we can get in training the model.

Experiments conducted:

Relationship between number of hidden nodes and accuracy:

We varied the number of hidden nodes from 4 to 20 in increments of 4 and from 20 to 50 in increments of 10.

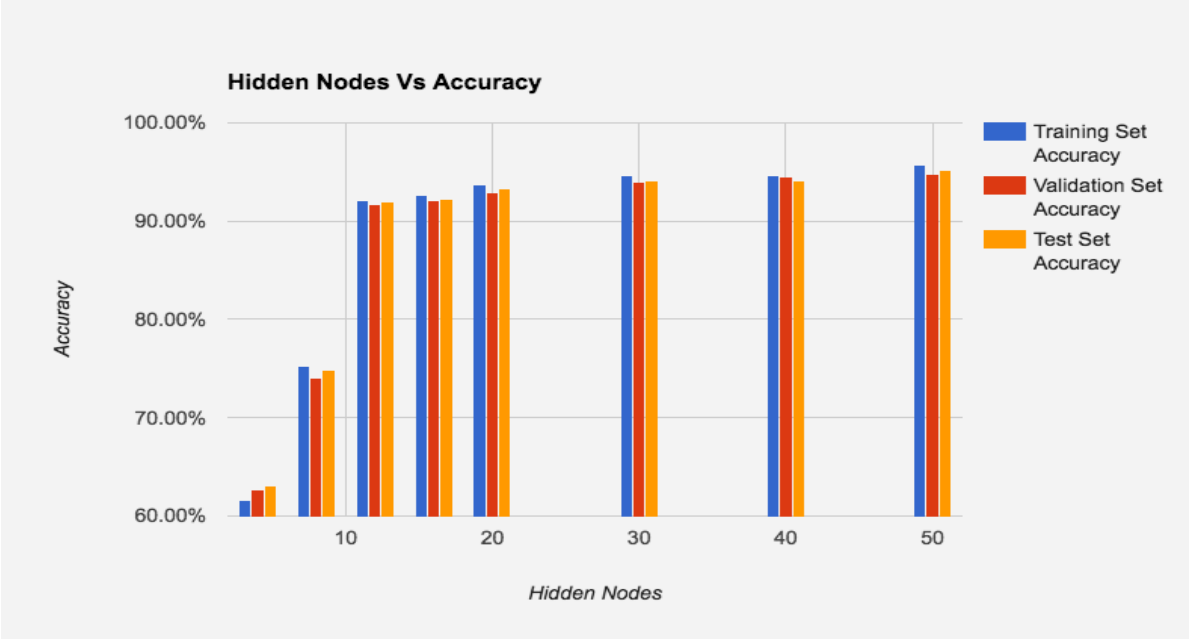

Figure : Hidden nodes Vs Accuracy

As seen from the above graph, the **accuracy** of the results obtained **improved as the number of hidden nodes increased** keeping the hyper-parameter value to zero. The concept of higher nodes giving better accuracy on the results as it avoids the problem of generalization is the reason for the above behavior. The training dataset, test dataset and validation dataset had the best performance when the number of nodes was set to 50. We also tried doing a random experiment of increasing the number of hidden nodes till 250 to see the trend of accuracy and how the overfitting problem is handled. We found that the accuracy values peaked at 50 and 100 hidden nodes and started to drop off after that, this is probably because of the overfitting problem.

Relationship between number of hidden nodes and time taken to train the network:

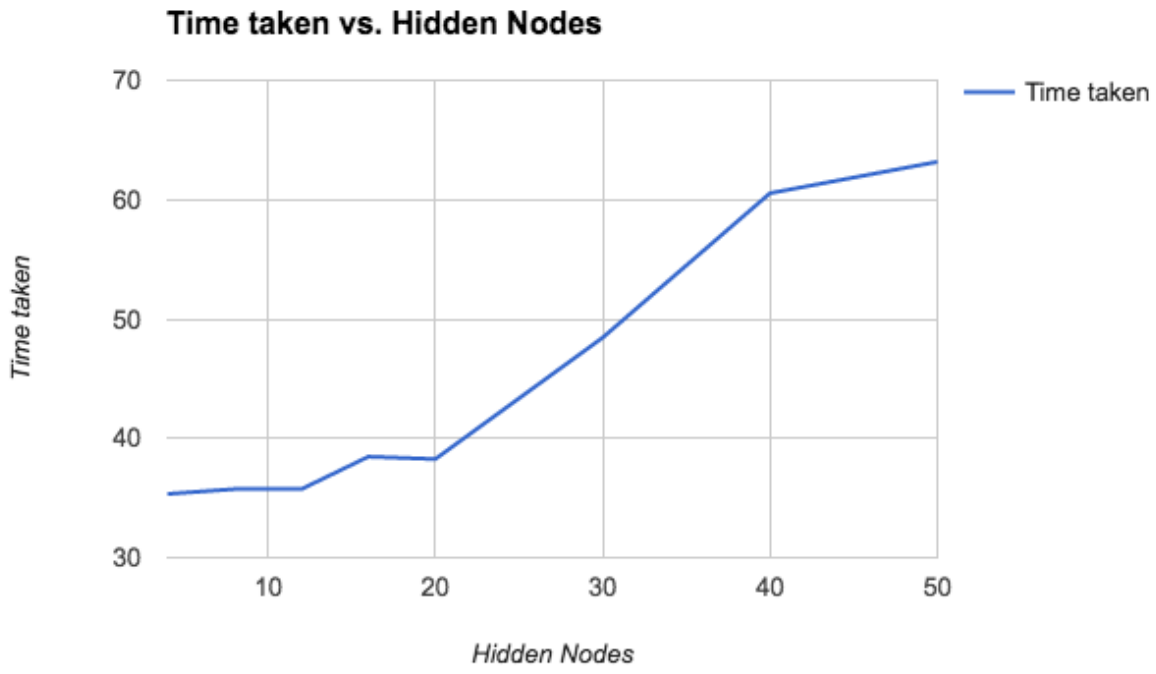

Figure : Hidden nodes vs Time taken

We checked how the time taken to train the network varies based upon the number of hidden nodes present in neural network. We see that, **as the hidden nodes were increased the time taken to train the network also increased**. This linear increase is purely performance oriented as it would take a lot of time to perform the cycle of forward-Propagation and backward-propagation that involves weights update, gradient calculations over a cycle of iterations. This would take significantly larger time to converge considering a larger number of nodes.

Relationship between regularization parameter λ and accuracy:

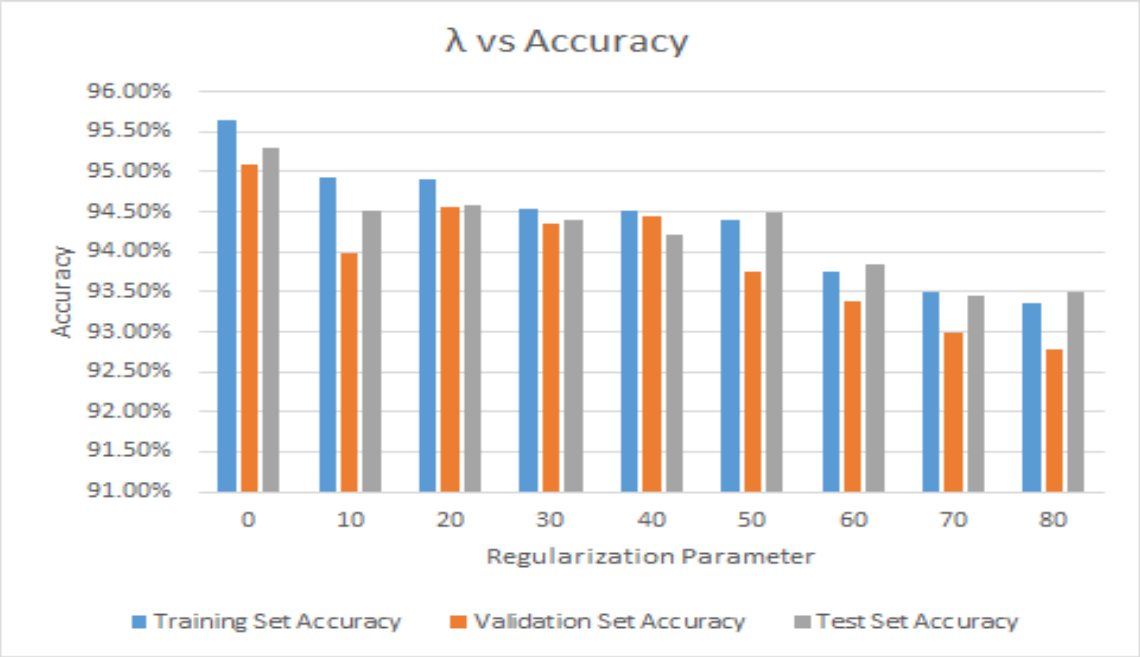

Figure : Plotting λ Vs Accuracy for # of hidden nodes = 50

From the first two experiments, We observed that the results are optimal when the number of hidden nodes is set to 50. We then varied the λ values and and tried to determine how it influences the accuracy of the training, validation and test data.

We observed that **as the regularization parameter λ increases the accuracy of the neural network model decreases**. This is due to the fact that regularization involves learning rate, momentum while calculating gradient descent, etc. We were able to evidently see the decrease in accuracy once the hyper parameter crossed the value of 10.

Regularization helps to choose preferred model complexity, so that model is better at predicting. It adds a penalty term to the objective function and controls the model complexity. If it is made too large, it might lead to overfitting and if it is too small then it does not adequately fit the data, hence we have to choose a value that lies in between the two. According to our analysis, we found that $\lambda < 10$ gave us consistent results on accuracy. Also, increasing the values of λ to larger values lead to reduction in the accuracy.

We collected data on how the neural network behaves when varying the hyper-parameter values. We didn't find a matching pattern of a peak at any particular combination. The neural network behaved perfectly when we used smaller values for the regularization parameter.

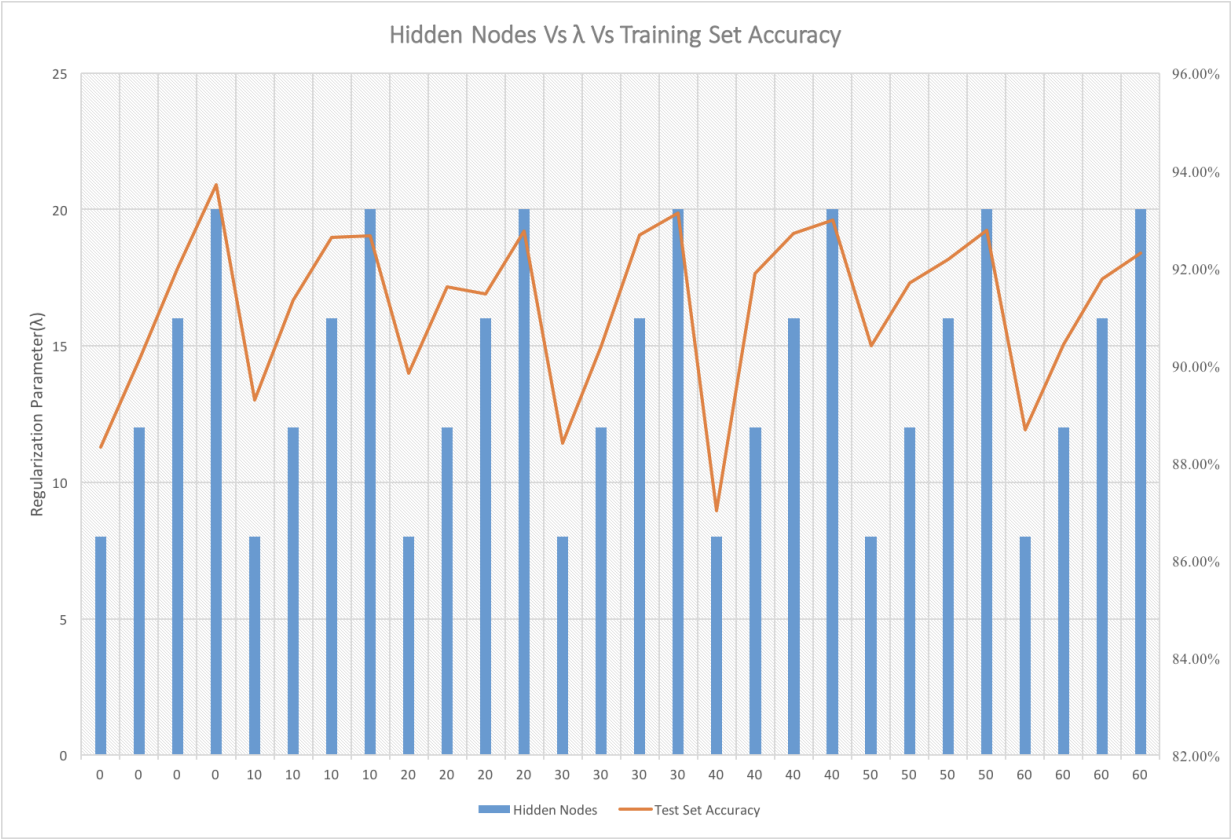

Figure : Plotting values of hidden nodes and λ along with accuracy

The above graph shows the variation of the hidden nodes and the regularization parameter against the test set accuracy. The hidden nodes ranges over 0 to 60 with increments of 10 and the regularization parameter (λ) ranges over 4 to 20. We obtained the peak accuracy of 93.71% with the number of hidden nodes as 20 and the value of regularization parameter (λ) equal to 0.

Comparison of neural network and deep neural network on the Celebrity data set:

In this phase of the project, there were mainly two tasks. Firstly, we ran the file FaceNN.py by implementing the sigmoid function, objective function and predict function of the previous implementation for the Celebrity data set.

And the second part was to run the DeepNN.py by changing the number of hidden layers using the Tensorflow library. We ran the same for the Celebrity dataset for 3,5 and 7 layers.

The obtained results of accuracy and time taken for the experiments were as follows :

Number of layers	Accuracy	Time Taken (Seconds)
Single Layer NN	85.92%	146.7
3 layers- Deep NN	78.72%	151.3
5 layers- Deep NN	75.05%	191.1
7 layers- Deep NN	72.59%	222.5

We notice that the **accuracy decreases as the number of layers were increased**. This is because it is hard to train deep neural networks, and when there are many layers this leads to a problem known as vanishing gradients. The backpropagation computes the derivatives for all weights using the chain rule and the for deep layers the chain becomes too long and the derivatives become hard to estimate and hence the algorithm breaks down giving lower accuracy values.

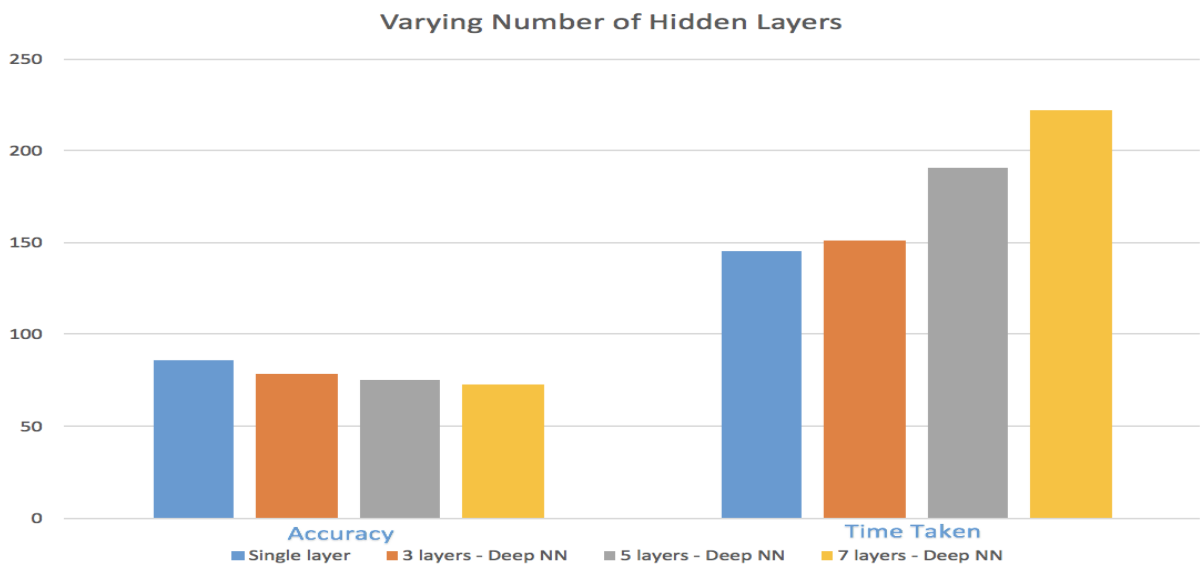

Figure : Varying the number of hidden layers, in single layer NN and Deep NN

The above shows the graphical representation of the accuracy and time taken with respect to different number of layers.

Conclusion:

We tried to determine the best combination of hyper-parameter and number of hidden nodes that gives the best accuracy in determining the handwritten digits, we observed that the neural networks worked best when the number of hidden nodes was 50 for lower values of regularization parameter. Considering a network of hidden nodes ranging from 4 to 20, the regularization parameter value of 0 with 20 hidden nodes gives a good trade off with accuracy around ~93%.

Test data Accuracy on Digits Classification - 95.31% (λ = 0, Hidden nodes = 50)

Test data Accuracy on Facial - Glasses recognition - 85.92%

References :

[1]https://www.mathworks.com/help/nnet/ug/improve-neural-network-generalization-and-avoid-overfitting.html?s_tid=gn_loc_drop

[2]http://colinraffel.com/wiki/neural_network_hyperparameters