CSE 574 Introduction to Machine Learning

Programming Assignment 2

Handwritten Digits Classification

Group 5

Prachi Ajay Sawant (50247675)
Likhith Kumar Miryala (50249280)

1 Introduction:

We have implemented a Multilayer Perceptron Neural Network and evaluated its performance in classifying handwritten digits. We have used the same neural network implementation in order to analyse a more challenging face dataset and have compared the performance of the neural network against a deep neural network using the TensorFlow library.

In this project assignment we have implemented feed Forward and back propagation methodology to adjust weights of model for predictive learning.

We have tried different values of the hyper-parameters to find optimal values for the performance of the neural network. We used different values of λ ranging from 0 to 60 in increments of 10. We also tried different hidden values between 4, 8, 12, 16, 20 and 50. Accordingly, we measured the accuracy and training time of the neural network on the given input datasets and used these values to choose the hyper-parameters to get the best performance.

2. Feature Selection in Pre-processing step:

This step is the most important step as we are trying to remove unwanted data and keeping only those that are relevant to the predictive modelling.

There are many such features which has exactly same value and has no significance. Hence, we can safely ignore those.

3. Simulation Process:

In this process we are trying to find the optimum parameters. There are various attributes that we need to take into consideration such as the number of hidden layers, regularization co-efficient, training, validation and test accuracy along with the time it takes to execute.

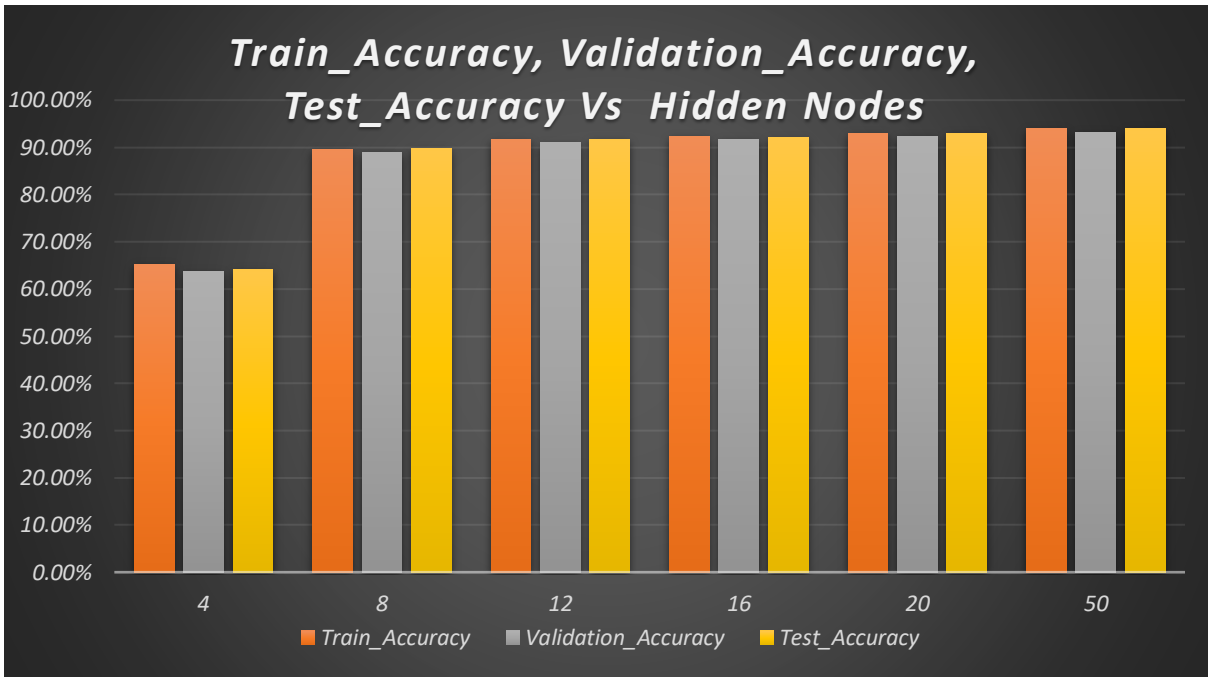
Variation of different Accuracies with Number of Hidden Nodes

4 Best accuracy value:
Regularization parameter=15
Hidden nodes=50

5.Accuracy Analysis of Neural Network for Handwritten Digits

Accuracy V/s Hidden Nodes

Hidden	Train_Accuracy	Validation_Accuracy	Test_Accuracy
4	65.13%	63.74%	64.03%
8	89.60%	88.86%	89.78%
12	91.71%	90.95%	91.75%
16	92.27%	91.72%	92.04%
20	92.95%	92.27%	93.03%
50	**94.00%**	**93.25%**	**93.99%**



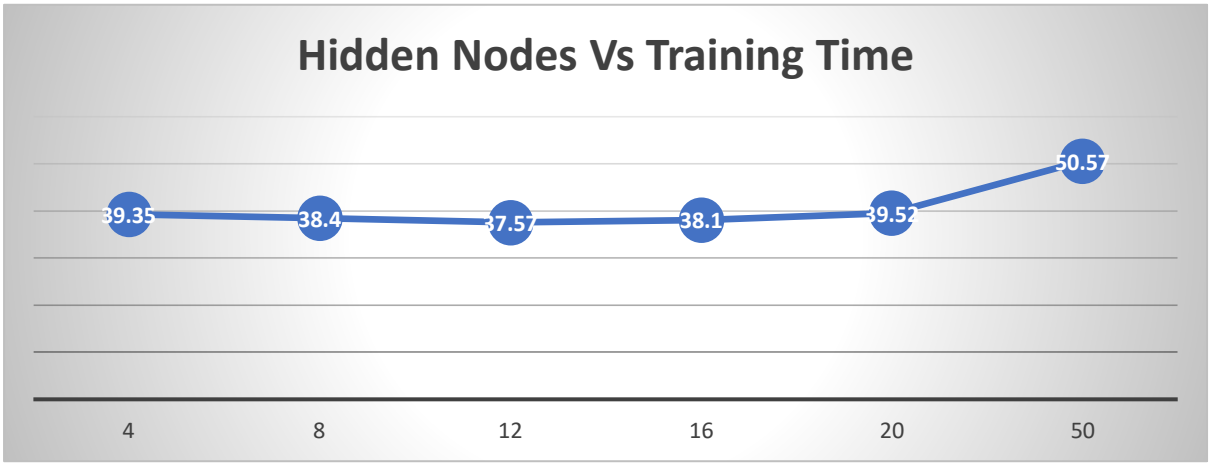
From the above graph we get to know that the training accuracy, validation accuracy and test accuracy increases as the number of hidden nodes increases.

6. Training time analysis of Neural network for handwritten digits

Hidden Nodes Vs Training Time

Hidden	Execution time	Train_Accuracy	Validation_Accuracy	Test_Accuracy
4	39.35	69.84%	68.64%	68.99%
8	38.4	87.95%	87.00%	87.81%
12	37.57	92.53%	92.24%	92.36%
16	38.1	92.80%	91.89%	92.57%

20	39.52	93.78%	93.10%	93.26%
50	50.57	95.33%	94.17%	95.04%



From the above graph we can come to conclusion that as we increase the number of hidden nodes the training time also increases.

Too many hidden units may lead to the slowness of training phase. This holds true as more number of hidden units imply greater size of hidden layer and output weight matrix and hence that adds to the complexity of the system.
Hence any Minimization/Optimization algorithm will need more time in computing these increased weights and spitting out converge weight matrix with low error function value.

7. Accuracies with different values of Lambda

Hidden	Lambda	Train_Accuracy	Validation_Accuracy	Test_Accuracy
50	0	95.53%	94.95%	94.86%
50	10	95.16%	94.78%	94.75%
50	15	95.21%	93.85%	94.55%
50	20	94.75%	94.33%	94.66%
50	30	94.45%	94.12%	94.32%
50	40	94.55%	93.8%	94.31%
50	50	94.10%	92.90 %	94.05%
50	60	94.00%	92%	93.71%

We use regularization in Neural Network to avoid overfitting problem. When we try different values of lambda the accuracy tends to decrease as the lambda value increases.

It appears that training, validation and testing data are closely related and hence applying more regularization coefficient backfires in this experiment. While we had expected it to increase it turn the other way around.

8. Comparative Analysis of Single Hidden layer implementation and Deep Neural Network on the CelebA data set.

Performance of face Dataset using implemented algorithm

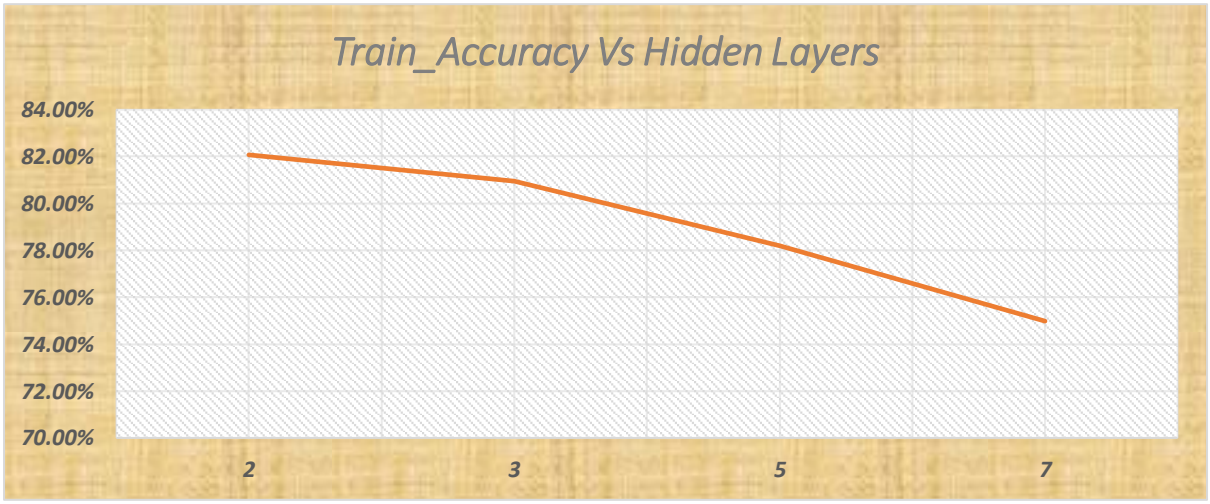
We used the same forward feed and back propagation algorithm/code for CelebA dataset and below are the observations:

Training set Accuracy	85.17%
Validation set Accuracy	83.71%
Test set Accuracy	85.46%
Evaluation time	198.642 s

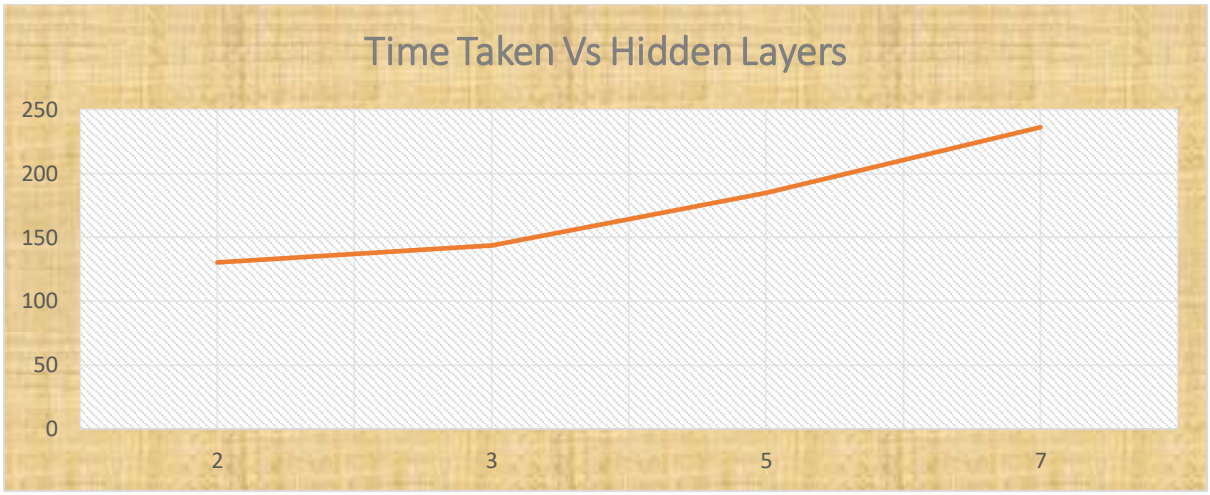
Performance of face Dataset using deep neural network(TensorFlow)

We have used the neural network with single hidden layer on a complex face dataset to distinguish between two classes - wearing glasses and not wearing glasses. We have used deep neural network on the same dataset and compared the results.

Number of Hidden Layers	Execution Time (Seconds)	Accuracy
2	130.35	82.07%
3	143.83	80.96%
5	184.95	78.19%
7	196.14	74.98%



In the above graph we can see that as the number of hidden nodes increases the accuracy decreases. This is because certain threshold addition of many hidden nodes can lead to decreasing accuracy.



If we compare our implemented algorithm of Single Layer Forward feed and backward propagation surpasses Tensorflow library in terms of accuracy percentage. Our implementation algorithm gives accuracy of 84.48% on test data whereas Tensorflow library gives best accuracy of 82.07% with 2 Hidden Layers. In terms of time deep neural network seems to be faster.

9. Convolutional neural network in terms of accuracy and training time:

```
Accuracy on Test-Set: 11.6% (1162 / 10000)
Optimization Iteration:      1, Training Accuracy:  12.5%
Time usage: 0:00:00
Accuracy on Test-Set: 11.7% (1171 / 10000)
Time usage: 0:00:05
Accuracy on Test-Set: 66.7% (6669 / 10000)
Optimization Iteration:    101, Training Accuracy:  60.9%
Optimization Iteration:    201, Training Accuracy:  85.9%
Optimization Iteration:    301, Training Accuracy:  76.6%
Optimization Iteration:    401, Training Accuracy:  93.8%
Optimization Iteration:    501, Training Accuracy:  95.3%
Optimization Iteration:    601, Training Accuracy:  92.2%
Optimization Iteration:    701, Training Accuracy:  95.3%
Optimization Iteration:    801, Training Accuracy:  84.4%
Optimization Iteration:    901, Training Accuracy:  93.8%
Time usage: 0:00:46
```

```
Accuracy on Test-Set: 93.1% (9307 / 10000)
Optimization Iteration:   9701, Training Accuracy: 100.0%
Optimization Iteration:   9801, Training Accuracy: 100.0%
Optimization Iteration:   9901, Training Accuracy:  98.4%
Time usage: 0:08:19
Accuracy on Test-Set: 98.7% (9871 / 10000)
[Finished in 580.996s]
```

As we see from the above output that as the number of iteration increases the test-set accuracy also increases. But the more iterations it performs the more time it requires to perform the operations. Here it requires **580.99 seconds** to compute.

The larger the dataset the more accuracy we obtain through convolutional neural network. Here, we have achieved accuracy of **98.7%** which is much larger than deep neural networks.

CNN uses relatively less pre-processing compared to other image classification algorithms. Convolutional Neural Networks are very similar to that of an ordinary Neural Networks they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs,

performs a dot product and optionally follows it with a non-linearity. Hence it makes this algorithm faster over others.