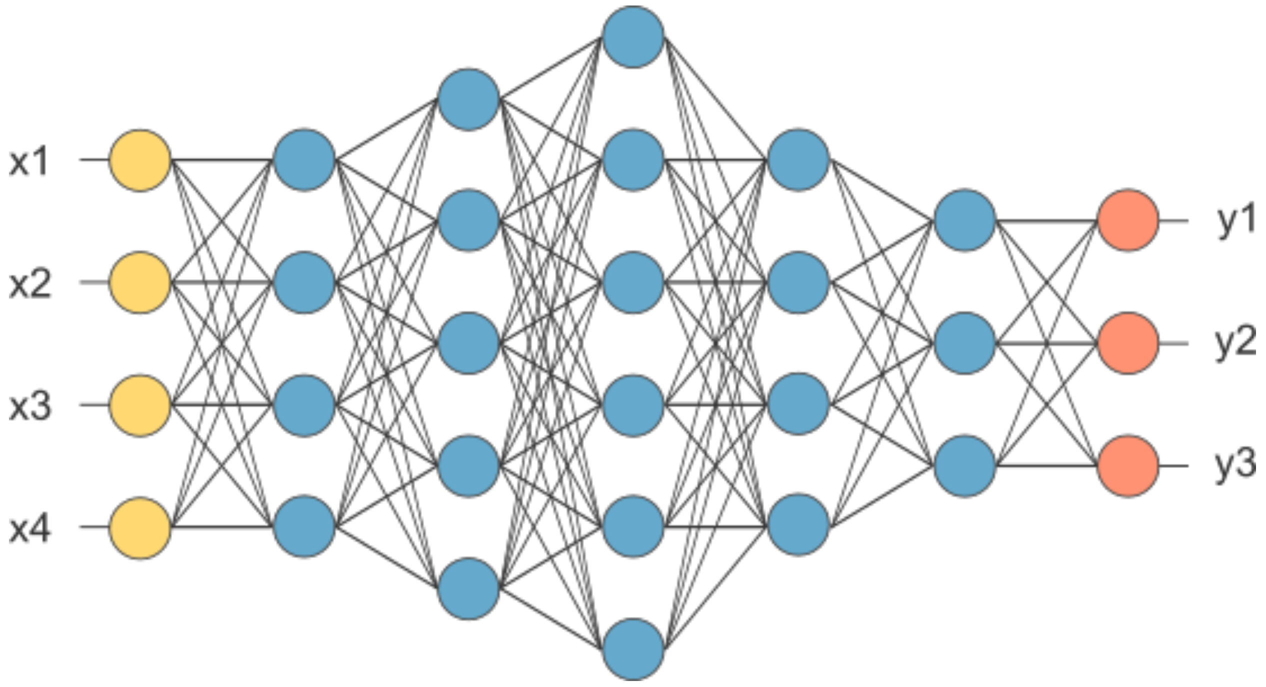
CSE574 Introduction to Machine Learning
Programming Assignment 2
Handwritten Digits Classification
Group 150

Ashutosh Ahmad Alexandar Harish Ganesan

In this assignment, we design and develop a Neural Network with one hidden layer as well as a MultiLayer Perceptron Neural Network and check the performance using handwritten digit data which we have obtained from the MNIST data set. We then use a multi-layer neural network on a more complex Celeb face dataset by making use of TensorFlow.

In this report, we will be discussing the impact of feature selection (removing unnecessary data), the selection of number of hidden nodes, optimal regularization factor (lambda), performance with multiple hidden layers, etc.



Example of a Multi-Layer Perceptron Neural Network

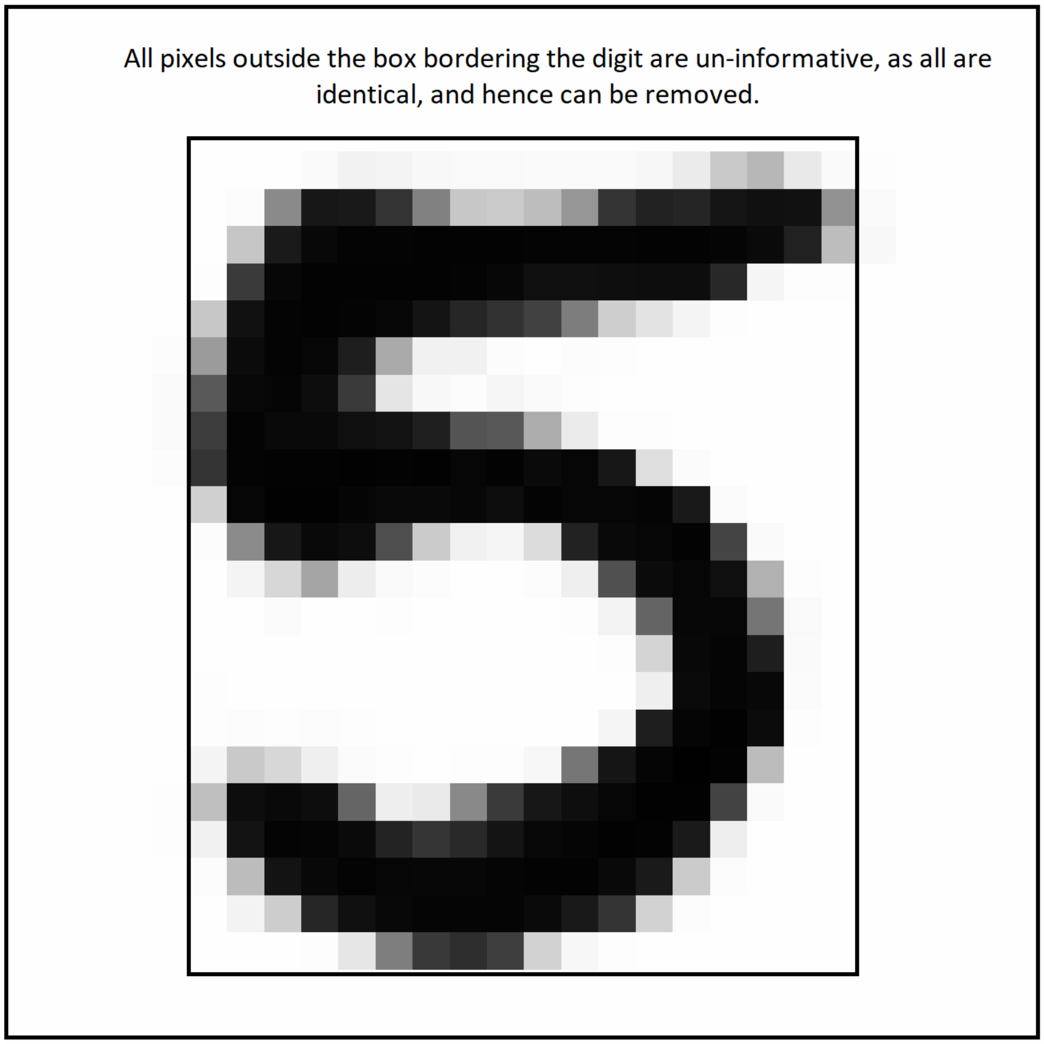
Feature Selection :

This is one of the first steps we need to perform. We need to decide what pixels contain useful information, and which pixels contain redundant information.

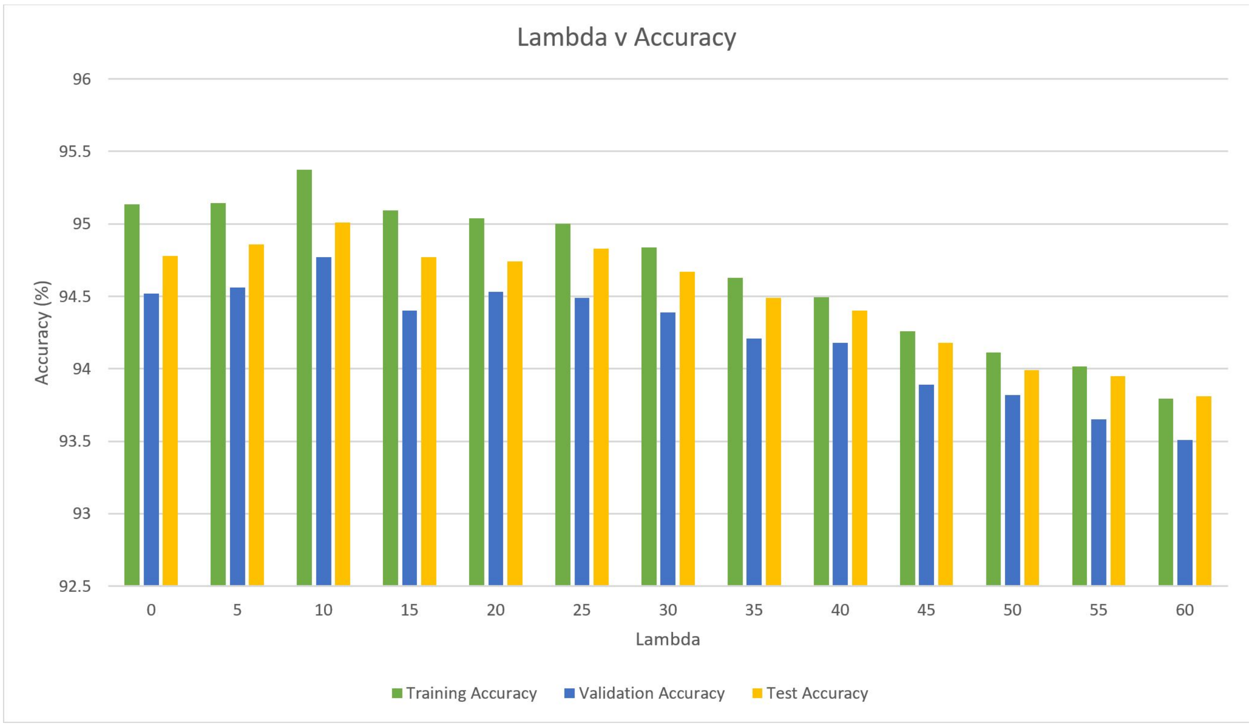
To do this, we see which features are the exact same as the background colour on the edges, which contain the redundant information. We then delete these pixels/features from the dataset.

These are the removed values from the dataset :
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 52, 53, 54, 55, 56 , 57, 59, 82, 83, 84, 85, 111, 112, 140, 141, 168, 476, 560, 644, 645, 671, 672, 673, 699, 700, 701, 727, 728, 729, 730, 754, 755, 756, 757, 758, 759, 780, 781, 782, 783]

Selected features are all indices minus the above indices, and it is provided in the params.pickle file.


All pixels outside the box bordering the digit are un-informative, as all are identical, and hence can be removed.

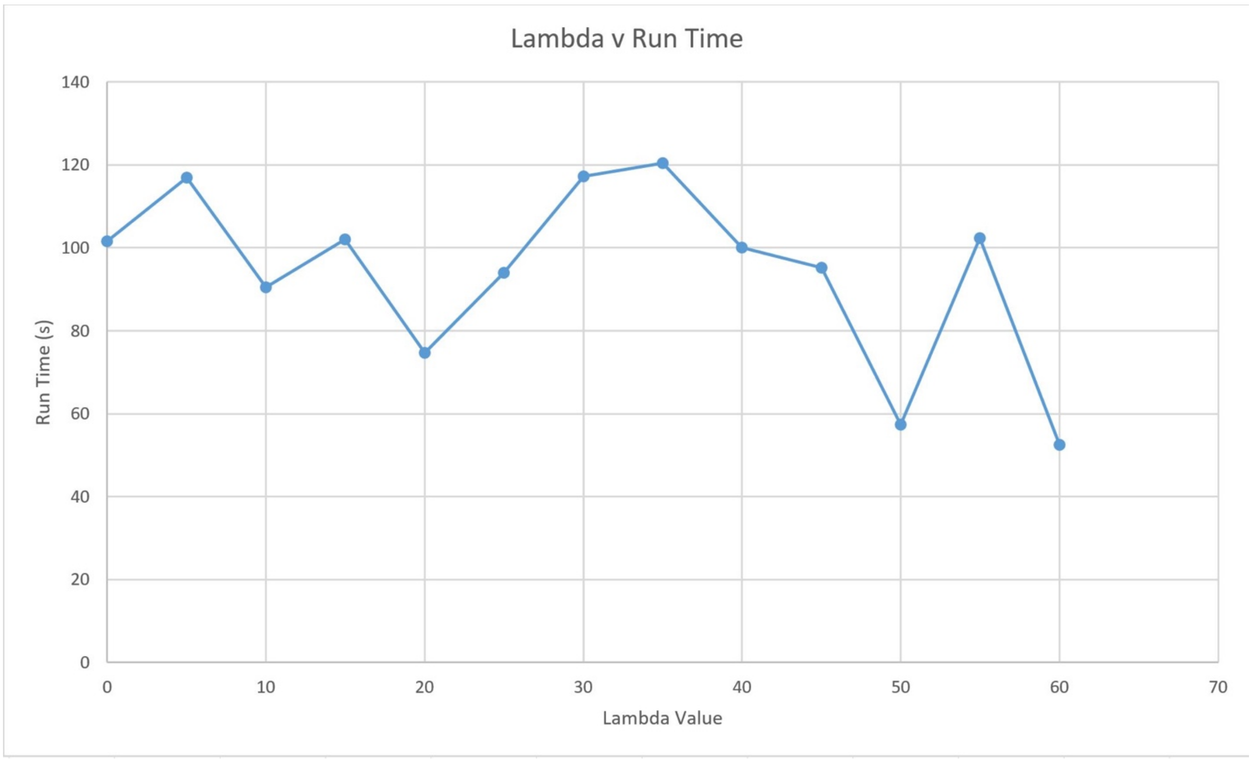
Relationship between Regularization Factor (λ) and Classification Accuracy



As we can see from the bar-chart shown above, we can clearly see that as the lambda increases from 0 to 5 to 10, the training, validation and testing accuracy also increases, and it is **maximum at lambda = 10**. From there on, as we increase the value of lambda, the accuracies keep decreasing, and this is because the values of the weights in the neural network are being changed by too large a value (for larger lambdas) and hence we get lower accuracies.

Now that we have found the optimal value for the regularization parameter, we must use this value of lambda = 10 for further experiments.

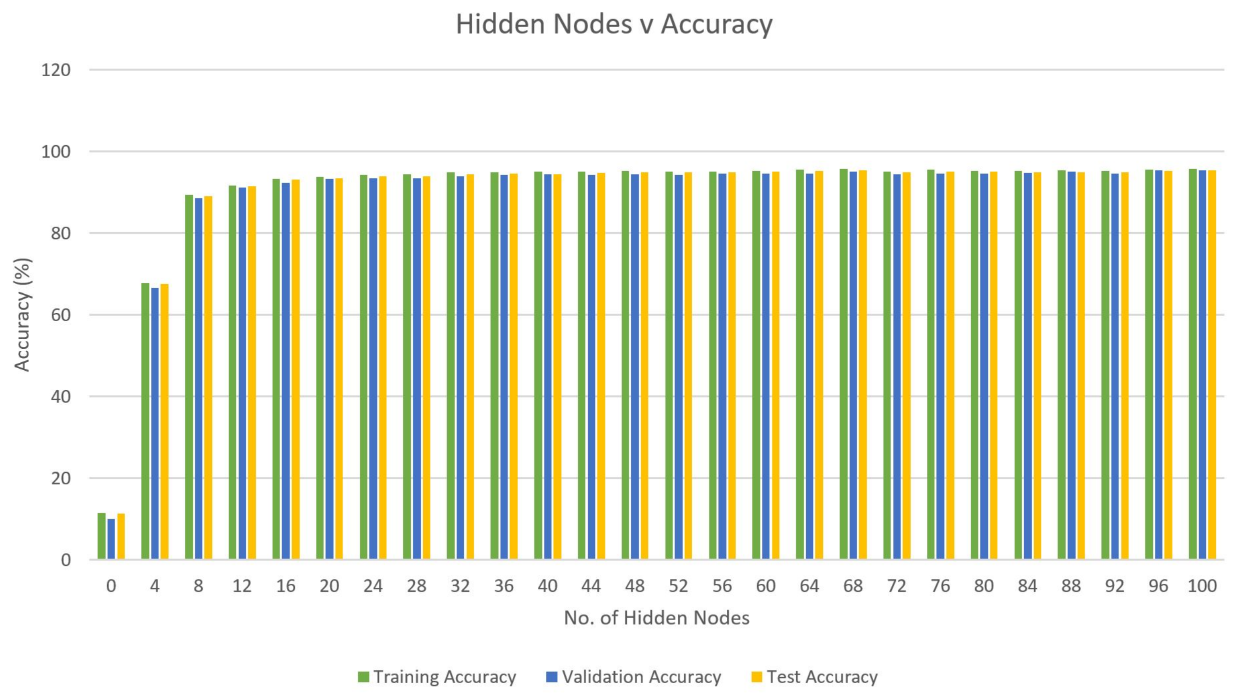
Relationship between Regularization Factor (λ) and Run-time



As shown in the graph above, the minimal times are present for larger values of lambda such as 50, 60. But if we take these higher lambda values into consideration, the accuracy drops sharply.

The fourth lowest run-time is when lambda = 10, and this is our optimal value in terms of accuracy, so we can decide furthermore that this is indeed our optimal lambda value.

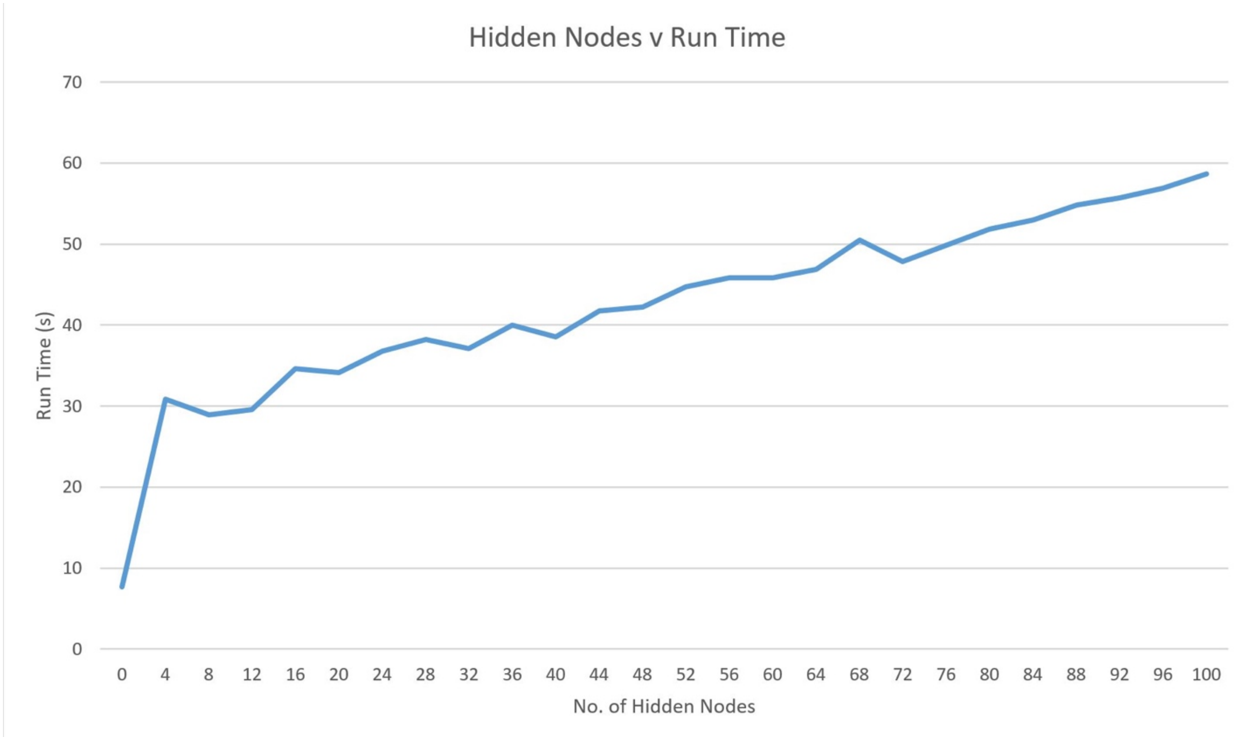
Relationship between No. of Hidden Nodes and Classification Accuracy



In this experiment, we fix the lambda value to 10 (optimal value).

As we can see from the above graph, all 3 accuracies for training, testing and validation increases as we increase the number of nodes in the hidden layer. In the beginning, the accuracies are low for smaller values such as 4 and 8, but then it increases by miniscule amounts thereon. This is because when we have more weight values, we can capture more variation in the input data. **The maximum accuracy achieved is in the case of no. of hidden nodes as 100 = 95.33% .**

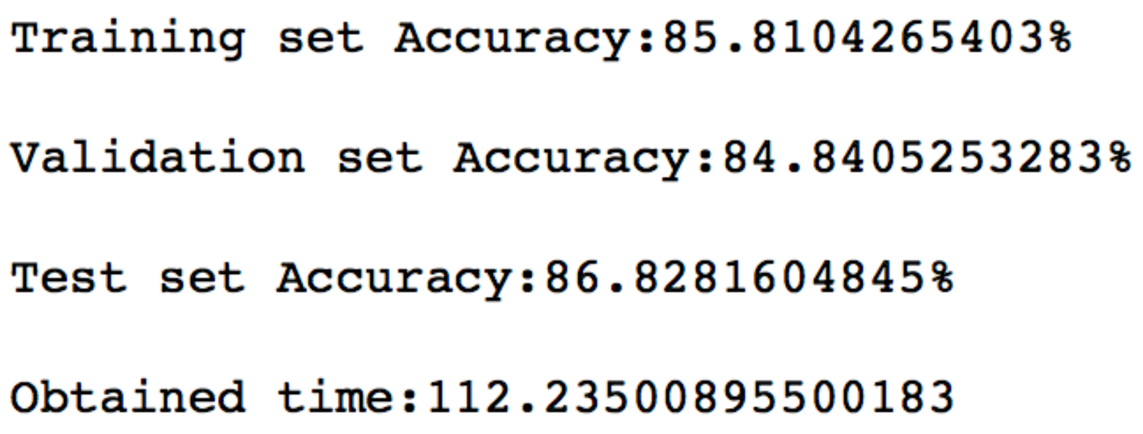
Relationship between No. of Hidden Nodes and Run Time



In this experiment, we fix the lambda value to 10 (optimal value).

From the above graph, we can observe that as we increase the no. of hidden nodes, the run time also increases. This is intuitive, because with the addition of mode hidden nodes, the number of computations will increase and naturally the run time too. **The highest run-time is when no. of hidden nodes = 100.**

Accuracy for Single Layer Neural Network – Celeb Face Data



```
Training set Accuracy:85.8104265403%

Validation set Accuracy:84.8405253283%

Test set Accuracy:86.8281604845%

Obtained time:112.23500895500183
```

As we can see from the above screenshot, when we run the facennScript, and copy our functions from nnScript to facennScript, we get the aforementioned accuracies, with a test accuracy **of 86.82%, and this runs in 112.235 s.**

Accuracy for Deep Neural Network – Celeb Face Data

In this case, we vary the no. of hidden layers between the values : 2,3,5 and 7.

The other parameters considered are :
n_hidden = 256, learning_rate = 0.0001, training_epochs = 100, batch_size = 100

The results for each of them are shown below :

No. of Hidden Layers	**Test Accuracy (%)**	**Run-time (s)**
2	80.5829	116.154
3	79.0689	124.786
5	74.8675	164.668
7	75.2839	227.138

Here, we can observe that as we increase the number of hidden layers, the run-time increases – this is intuitive because there will be more computation. The accuracy is dropping as we increase the no. of hidden layers, and we this is due to overfitting performed by the Deep Neural Network.

The accuracy as well as the run-time is marginally better when we compare the single layer Neural Network with the Deep Neural Network. This is because it takes lesser time due to just one hidden layer, and the accuracy is better because we aren't overfitting the data.

Convolution Neural Network Experiment

In this experiment, we set the dataset size as follows:

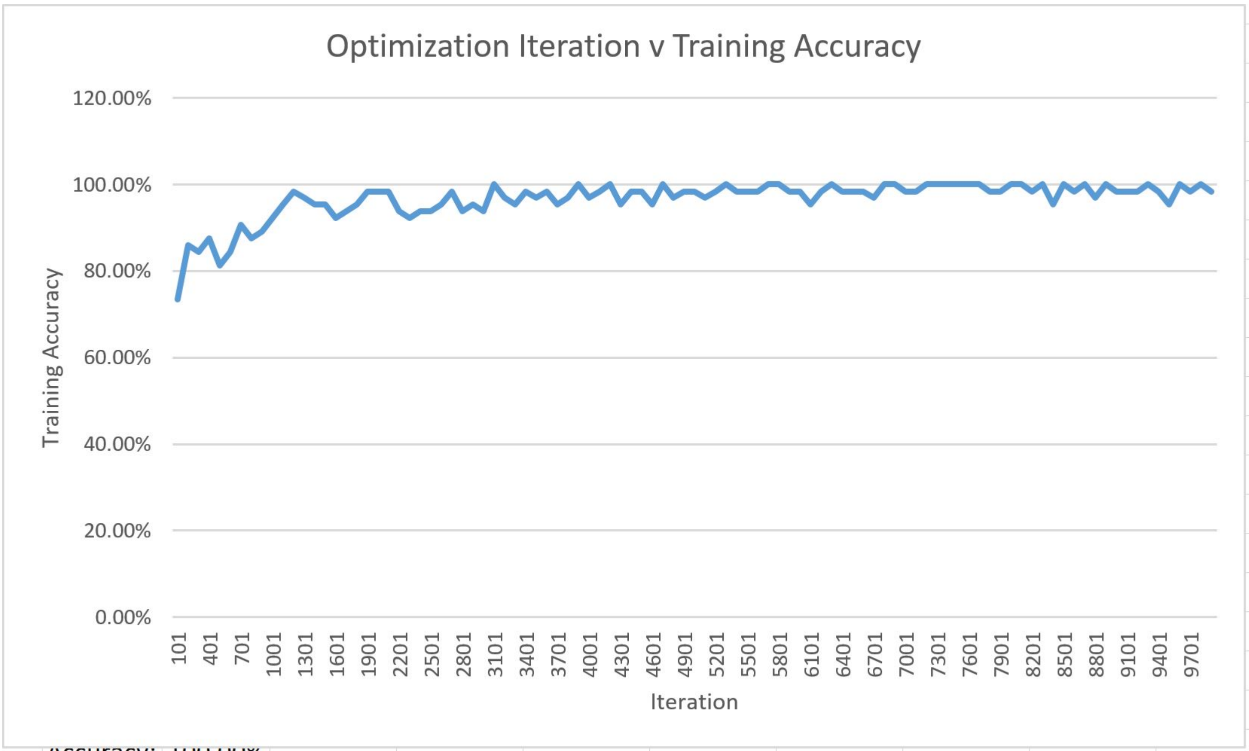
- Training-set: 55000 rows
- Test-set: 10000 rows
- Validation-set: 5000 rows

Shown below is the output of the CNN script.

```
Accuracy on Test-Set: 8.7% (873 / 10000)

Optimization Iteration:      1, Training Accuracy:  10.9%

Time usage: 0:00:00

Accuracy on Test-Set: 9.4% (941 / 10000)

Time usage: 0:00:05

Accuracy on Test-Set: 74.8% (7476 / 10000)

Optimization Iteration:    101, Training Accuracy:  73.4%

Optimization Iteration:    201, Training Accuracy:  85.9%

Optimization Iteration:    301, Training Accuracy:  84.4%

Optimization Iteration:    401, Training Accuracy:  87.5%

Optimization Iteration:    501, Training Accuracy:  81.2%

Optimization Iteration:    601, Training Accuracy:  84.4%

Optimization Iteration:    701, Training Accuracy:  90.6%

Optimization Iteration:    801, Training Accuracy:  87.5%

Optimization Iteration:    901, Training Accuracy:  89.1%
```

```
Time usage: 0:00:45
Accuracy on Test-Set: 93.3% (9327 / 10000)
Optimization Iteration:   1001, Training Accuracy:  92.2%
Optimization Iteration:   1101, Training Accuracy:  95.3%
Optimization Iteration:   1201, Training Accuracy:  98.4%
Optimization Iteration:   1301, Training Accuracy:  96.9%
Optimization Iteration:   1401, Training Accuracy:  95.3%
Optimization Iteration:   1501, Training Accuracy:  95.3%
Optimization Iteration:   1601, Training Accuracy:  92.2%
Optimization Iteration:   1701, Training Accuracy:  93.8%
Optimization Iteration:   1801, Training Accuracy:  95.3%
Optimization Iteration:   1901, Training Accuracy:  98.4%
Optimization Iteration:   2001, Training Accuracy:  98.4%
Optimization Iteration:   2101, Training Accuracy:  98.4%
Optimization Iteration:   2201, Training Accuracy:  93.8%
Optimization Iteration:   2301, Training Accuracy:  92.2%
Optimization Iteration:   2401, Training Accuracy:  93.8%
Optimization Iteration:   2501, Training Accuracy:  93.8%
Optimization Iteration:   2601, Training Accuracy:  95.3%
Optimization Iteration:   2701, Training Accuracy:  98.4%
Optimization Iteration:   2801, Training Accuracy:  93.8%
Optimization Iteration:   2901, Training Accuracy:  95.3%
Optimization Iteration:   3001, Training Accuracy:  93.8%
Optimization Iteration:   3101, Training Accuracy: 100.0%
Optimization Iteration:   3201, Training Accuracy:  96.9%
Optimization Iteration:   3301, Training Accuracy:  95.3%
Optimization Iteration:   3401, Training Accuracy:  98.4%
Optimization Iteration:   3501, Training Accuracy:  96.9%
Optimization Iteration:   3601, Training Accuracy:  98.4%
Optimization Iteration:   3701, Training Accuracy:  95.3%
Optimization Iteration:   3801, Training Accuracy:  96.9%
Optimization Iteration:   3901, Training Accuracy: 100.0%
Optimization Iteration:   4001, Training Accuracy:  96.9%
Optimization Iteration:   4101, Training Accuracy:  98.4%
Optimization Iteration:   4201, Training Accuracy: 100.0%
Optimization Iteration:   4301, Training Accuracy:  95.3%
Optimization Iteration:   4401, Training Accuracy:  98.4%
Optimization Iteration:   4501, Training Accuracy:  98.4%
Optimization Iteration:   4601, Training Accuracy:  95.3%
Optimization Iteration:   4701, Training Accuracy: 100.0%
Optimization Iteration:   4801, Training Accuracy:  96.9%
Optimization Iteration:   4901, Training Accuracy:  98.4%
Optimization Iteration:   5001, Training Accuracy:  98.4%
Optimization Iteration:   5101, Training Accuracy:  96.9%
Optimization Iteration:   5201, Training Accuracy:  98.4%
Optimization Iteration:   5301, Training Accuracy: 100.0%
Optimization Iteration:   5401, Training Accuracy:  98.4%
Optimization Iteration:   5501, Training Accuracy:  98.4%
Optimization Iteration:   5601, Training Accuracy:  98.4%
Optimization Iteration:   5701, Training Accuracy: 100.0%
Optimization Iteration:   5801, Training Accuracy: 100.0%
Optimization Iteration:   5901, Training Accuracy:  98.4%
Optimization Iteration:   6001, Training Accuracy:  98.4%
Optimization Iteration:   6101, Training Accuracy:  95.3%
Optimization Iteration:   6201, Training Accuracy:  98.4%
Optimization Iteration:   6301, Training Accuracy: 100.0%
Optimization Iteration:   6401, Training Accuracy:  98.4%
Optimization Iteration:   6501, Training Accuracy:  98.4%
Optimization Iteration:   6601, Training Accuracy:  98.4%
Optimization Iteration:   6701, Training Accuracy:  96.9%
Optimization Iteration:   6801, Training Accuracy: 100.0%
Optimization Iteration:   6901, Training Accuracy: 100.0%
Optimization Iteration:   7001, Training Accuracy:  98.4%
Optimization Iteration:   7101, Training Accuracy:  98.4%
Optimization Iteration:   7201, Training Accuracy: 100.0%
Optimization Iteration:   7301, Training Accuracy: 100.0%
Optimization Iteration:   7401, Training Accuracy: 100.0%
Optimization Iteration:   7501, Training Accuracy: 100.0%
Optimization Iteration:   7601, Training Accuracy: 100.0%
Optimization Iteration:   7701, Training Accuracy: 100.0%
Optimization Iteration:   7801, Training Accuracy:  98.4%
Optimization Iteration:   7901, Training Accuracy:  98.4%
Optimization Iteration:   8001, Training Accuracy: 100.0%
Optimization Iteration:   8101, Training Accuracy: 100.0%
Optimization Iteration:   8201, Training Accuracy:  98.4%
Optimization Iteration:   8301, Training Accuracy: 100.0%
Optimization Iteration:   8401, Training Accuracy:  95.3%
Optimization Iteration:   8501, Training Accuracy: 100.0%
Optimization Iteration:   8601, Training Accuracy:  98.4%
Optimization Iteration:   8701, Training Accuracy: 100.0%
Optimization Iteration:   8801, Training Accuracy:  96.9%
Optimization Iteration:   8901, Training Accuracy: 100.0%
Optimization Iteration:   9001, Training Accuracy:  98.4%
Optimization Iteration:   9101, Training Accuracy:  98.4%
Optimization Iteration:   9201, Training Accuracy:  98.4%
Optimization Iteration:   9301, Training Accuracy: 100.0%
Optimization Iteration:   9401, Training Accuracy:  98.4%
Optimization Iteration:   9501, Training Accuracy:  95.3%
Optimization Iteration:   9601, Training Accuracy: 100.0%
Optimization Iteration:   9701, Training Accuracy:  98.4%
Optimization Iteration:   9801, Training Accuracy: 100.0%
Optimization Iteration:   9901, Training Accuracy:  98.4%
Time usage: 0:07:30
Accuracy on Test-Set: 98.9% (9888 / 10000)
```

We can make a graph depicting the optimization iteration and training accuracy, and it is shown below :



Accuracy: 100.00%

As we can see, as the iterations increases, the training accuracy hits close to 100%.

The final test accuracy obtained for this is: **98.9%,** which is quite exceptional performance by the convolution network.

The params.pickle file contains 5 variables –

- selected_features – contains indices of useful features
- w1 – weights between input layer and hidden layer
- w2 – weights between hidden layer and output layer
- n_hidden – the number of nodes in the hidden layer
- lambda – optimal regularization parameter