CSE574-ASSIGNMENT 1 – GROUP 52

HANDWRITTEN DIGITS CLASSIFICATION

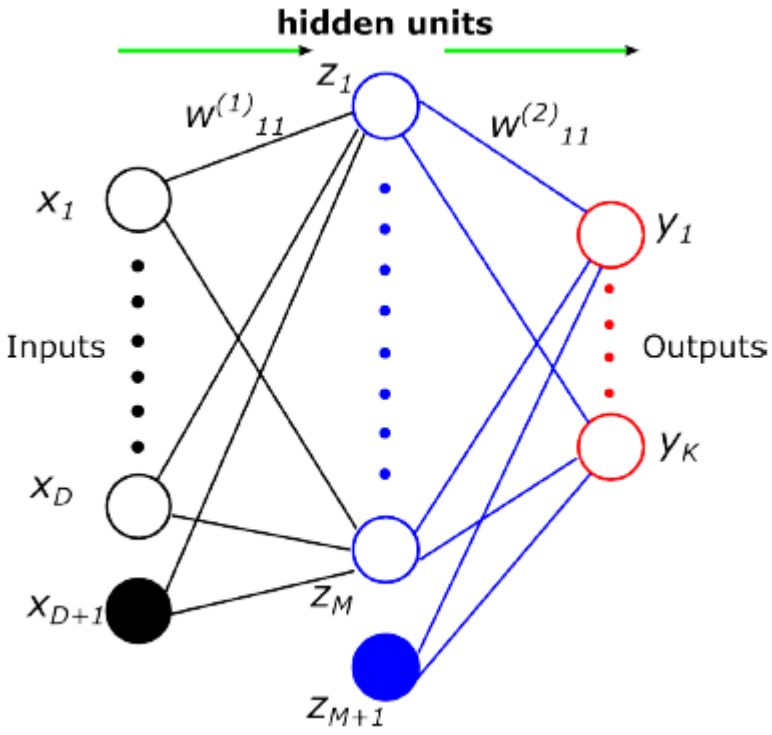
March 8, 2017

Name	UBIT Name	Person #
Sourav Puri	Souravpu	50206918
Shaleen Somani	Shaleens	50207380
Amardeep Virdy	Avirdy	50208831

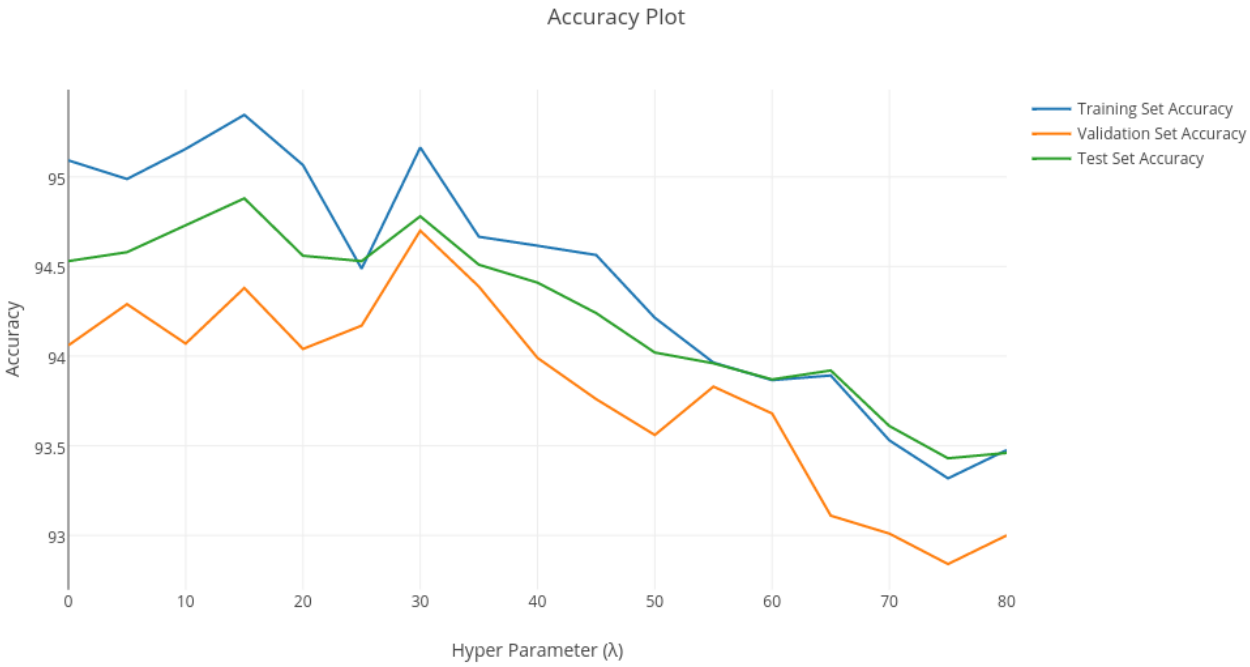
Neural Network - Introduction

A neural network involves a large number of various processes working in parallel and arranged in tiers or levels. A neural network is generally built on three layers. The first part receives the input information. The second layer is generally the hidden layer and the third layer usually is the output layer. Each successive layer or set of nodes receives the output from the layer preceding it, rather than from the starting input. The last layer produces the output of the system. The neural is pretty much similar to the way the human brain works where each neuron processes some information that is transferred to other neurons in the brain.

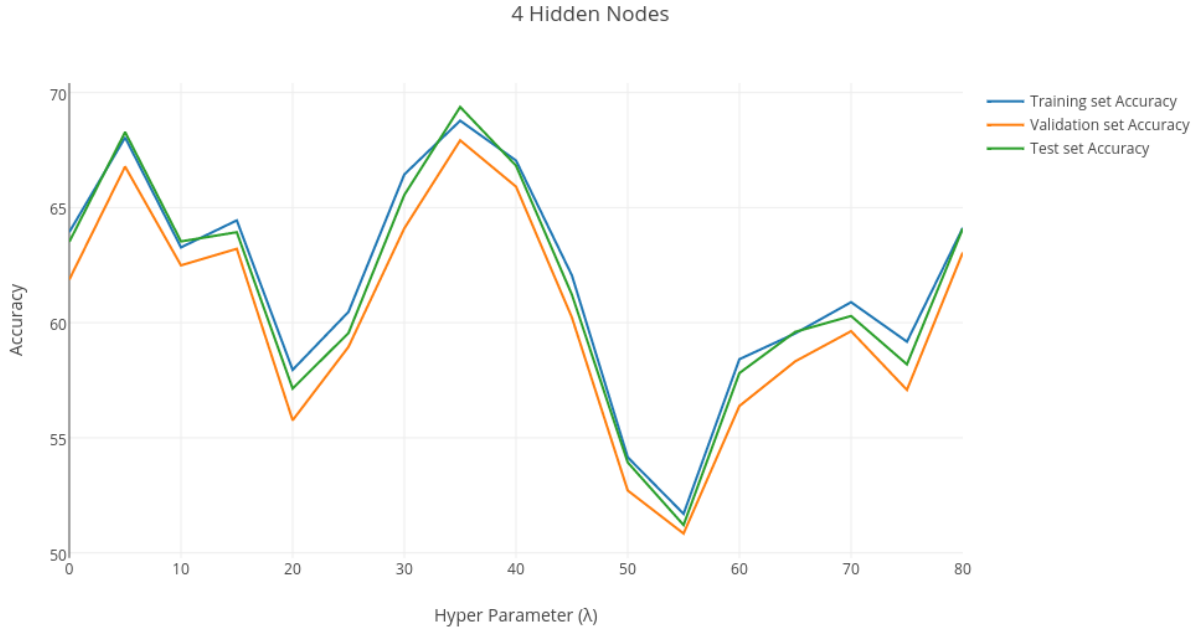
In this assignment we implement a Multilayer Perceptron Neural Network and evaluate its performance in classifying handwritten digits. We also use the same network to analyze a more challenging face dataset and compare the performance of the neural network against a deep neural network using the TensorFlow library.



Results for Handwritten Digits Classification:

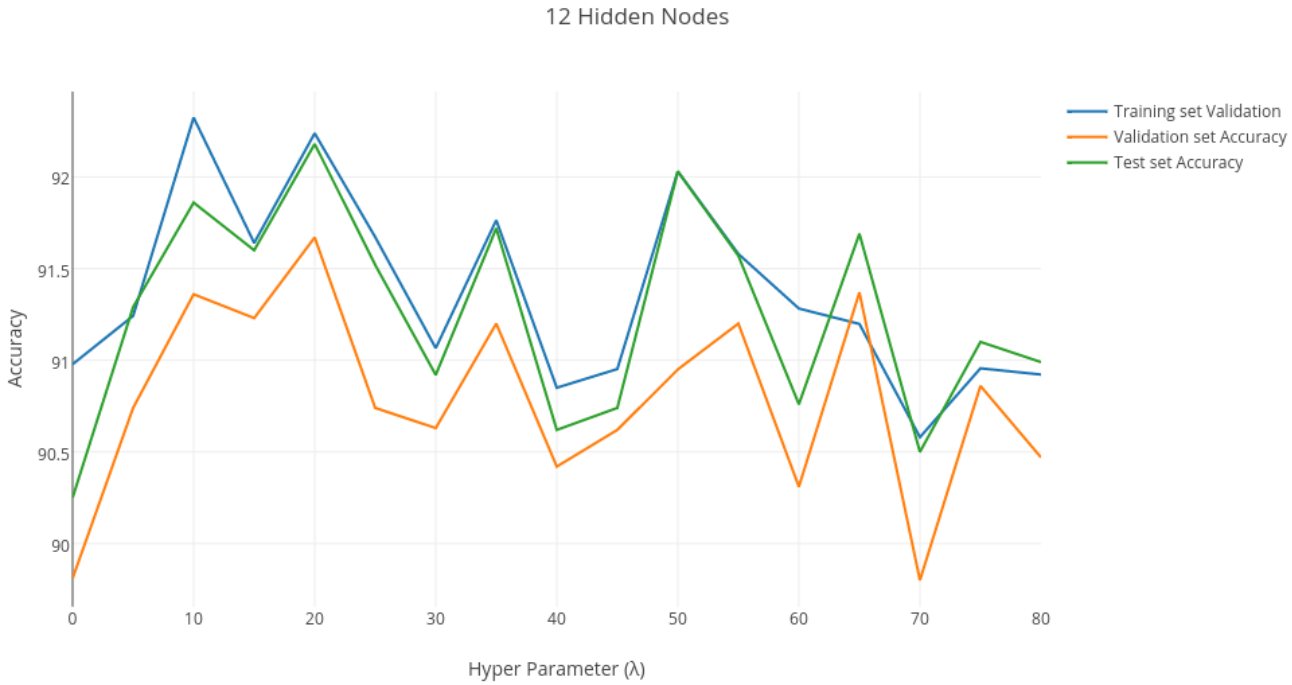


As observed from the above graph, as lambda values are increased, the accuracy peaks at a certain value of lambda and then the overall accuracy gradually decreases.

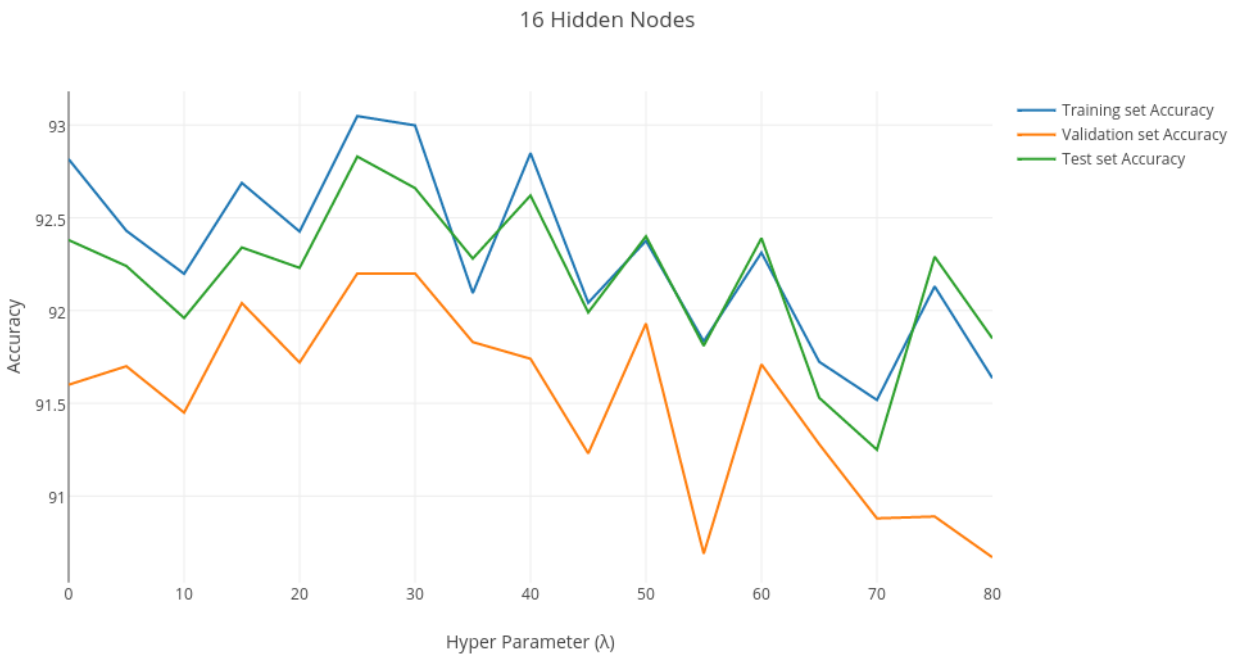


For 4 nodes in hidden layer, it can be observed that as the lambda value is increased, we encounter a point at which accuracy is maximum, which is equal to 35 in this case.

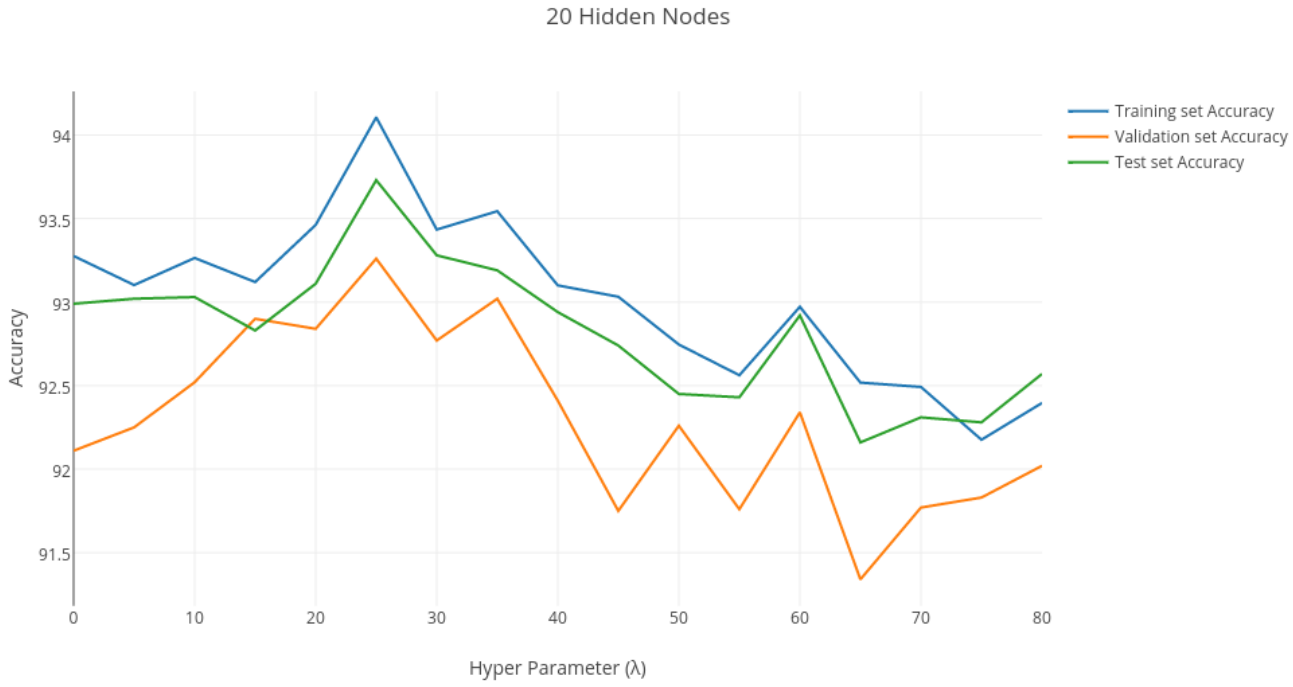
For 8 nodes in hidden layer, it can be observed that as the lambda value is increased, we encounter a point at which accuracy is maximum, which is equal to 60 in this case.



For 12 nodes in hidden layer, it can be observed that as the lambda value is increased, we encounter a point at which accuracy is maximum, which is equal to 10 in this case.



For 16 nodes in hidden layer, it can be observed that as the lambda value is increased, we encounter a point at which accuracy is maximum, which is equal to 25 in this case.



For 20 nodes in hidden layer, it can be observed that as the lambda value is increased, we encounter a point at which accuracy is maximum, which is equal to 25 in this case.

Choosing the optimal hyper-parameter

When the model fits the training data but does not have a good predicting performance and generalization power, we have an overfitting problem. **Regularization** is a technique used to avoid this overfitting problem. The idea behind regularization is that models that over fit the data are complex models that have for example too many nodes in hidden layer resulting in too many parameters.

In order to find the best model, the common method in machine learning is to define a loss or cost function that describes how well the model fits the data. The goal is to find the model that minimizes this loss function. The idea of **regularization** is to penalize loss function by adding a complexity term that would give a bigger loss for more complex models. So we manipulate the value of Lambda to penalize more or less, the complex models. This way, for a very large lambda, models with high complexity are ruled out. And for small lambda, models with high training errors are ruled out. The optimal solution lies somewhere in the middle.

After evaluating all the results of our neural network above, we infer that as the number of nodes in hidden layer increases, the overall accuracy of the results increase as well. We also noticed that we encounter a peak value for a certain hyper-parameter(lambda) value when we vary it in the given range (0-80). It is also noticeable that training time increases as the number of nodes in hidden layer increases which is quite obvious because the computation increases as well.

There are basically four methods for choosing an optimal hyper parameter:

Manual Search: Using knowledge we have about the problem, we can guess parameters and observe the result. Based on that result we can change the parameters. Repeating this process until we find parameters that work well. **This is the approach we have used in our project**.

Grid Search: Using knowledge we have about the problem; we can identify ranges for the hyper parameters. Then selecting several points from those ranges, usually uniformly distributed, we can train our network using every combination of parameters and select the combination that performs best. Alternatively, we can repeat our search on a narrower domain centered around the parameters that perform the best.

Random Search: Like grid search we can use knowledge of the problem to identify ranges for the hyper parameters. However instead of picking values from those ranges in a methodical manner we can instead select them at random. Repeat this process until we find parameters that work well or use what we learn to narrow the search.

Bayesian Optimization: This approach focuses on improving upon these other approaches by using the information gained from any given experiment to decide how to adjust the hyper parameters for the next experiment.

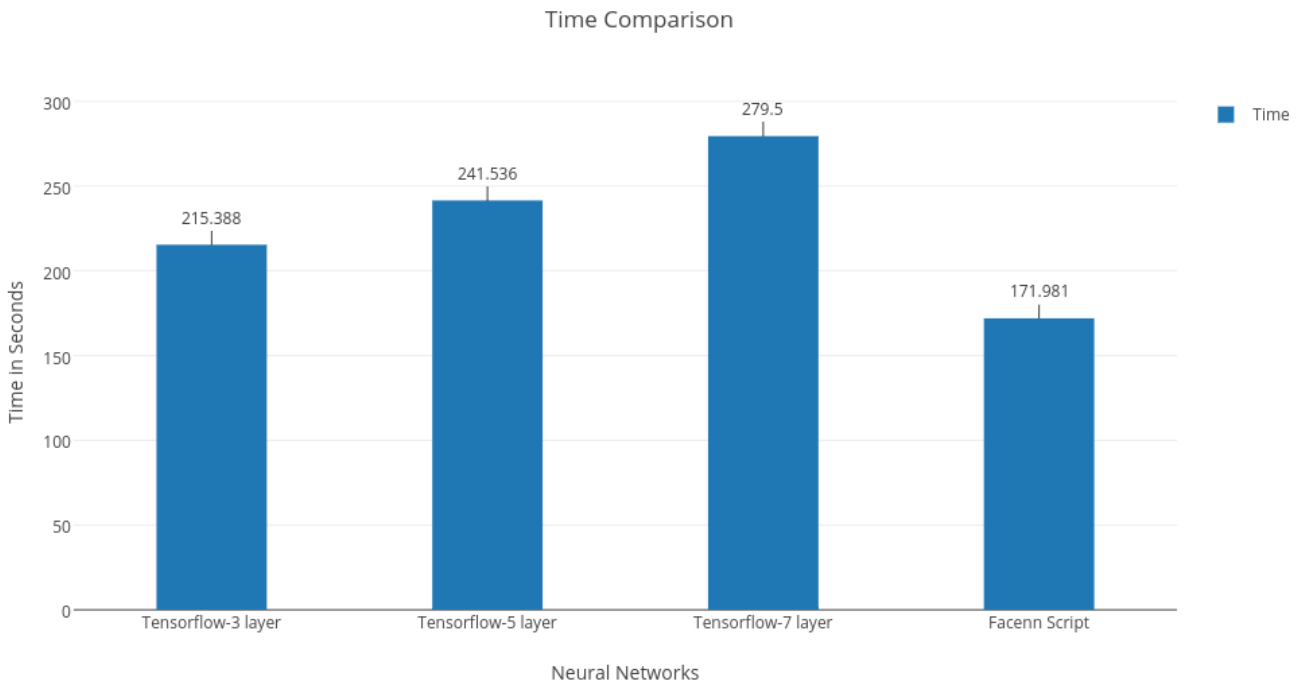
Accuracy for Handwritten Digits Classification

Training set Accuracy: 95.158%
Validation set Accuracy: 94.73%
Test set Accuracy: 94.99%
Lambda: 15
Number of nodes in hidden layer: 50

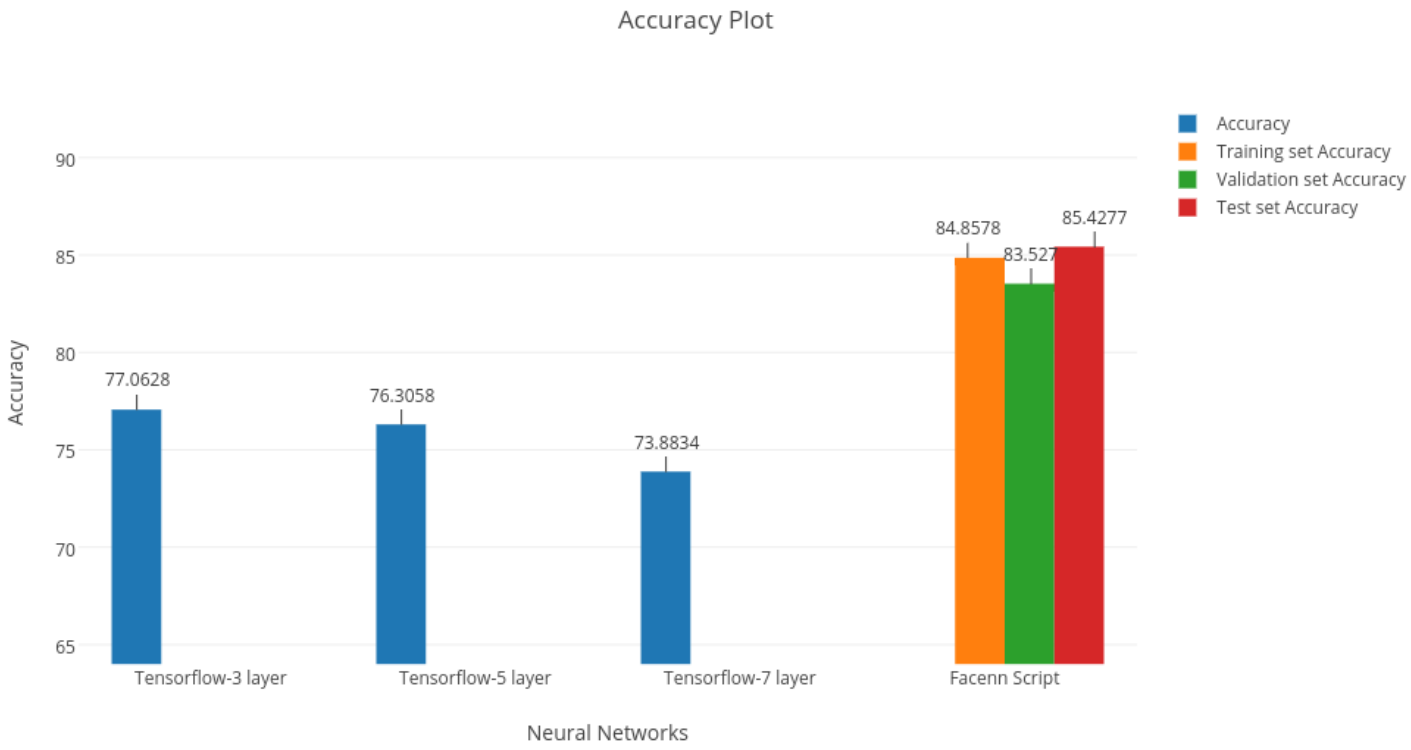
Accuracy on CelebA dataset

Training set Accuracy: 84.8578%
Validation set Accuracy: 83.527%
Test set Accuracy: 85.4277%

Comparison of our neural network with deep neural network



As we can observe from the bar graph below, the time taken increase as we increase the number of hidden layers in the Tensor Flow deepnnScript, which is quite obvious because with increase in the hidden layers the complexity of the neural network increase which leads to more time in computation. In case of our own neural network in facennScript, we are using only a single hidden layer which leads the comparatively the lowest time.



From the above plot, it is observed that even though the deep neural network has more layers, the accuracy is still lower as compared to our neural network with single hidden layer. Furthermore, it is noticed that as more hidden layers are added to the deep neural network, accuracy decreases. This is due to the fact that more complicated networks do not necessarily give better results as per the principle of Occam's razor which states that simpler theories are preferable to more complex ones because they are more testable.

References:

1. *https://en.wikipedia.org/wiki/Artificial_neural_network*
2. *http://stats.stackexchange.com/questions/95495/guideline-to-select-the-hyperparameters-in-deep-learning*
3. *https://justindomke.wordpress.com/2008/12/12/why-does-regularization-work/*
4. *https://datanice.github.io/machine-learning-101-what-is-regularization-interactive.html*
5. *https://en.wikipedia.org/wiki/Occam's_razor*
6. *https://piazza.com/class/iwz2vra81n2dg?cid=232*