Deep Learning Assignment – 3 (14EC35011)

The assignment is divided into 3 parts:

(a) Training the neural network using tensorflow without the use of tf.nn functions
(b) Testing the neural network with the given data
(c) Training each layer output using logistic regression.

Part 1: Training the Neural Network (python Assignment3_14EC35011.py --train)

1. Data was collected using the functions provided by the data_loader.py file
2. New implementations of relu and softmax was used as follows
    a. def relu(x): return tf.maximum(x,0)
    b. def softmax(x): return tf.divide(tf.exp(tf.subtract(x,tf.reduce_max(x))), tf.reduce_sum(tf.subtract(x,tf.reduce_max(x))))
3. The datasets were converted from np.int32 type to np.float64 type and then normalized by dividing 255. This was done to prevent the loss function or softmax function from shooting to a very high value.
4. The cost function was changed to –tf.reduce_sum(self.Y*tf.log(self.pred)) (The cross entropy function)
5. Gradient Descent Optimizer was used to train the network
6. Close to the peak accuracy value it was observed that the avg_cost value became nan. So required measures were taken.
7. The model with best validation accuracy was saved in the log files. The link to the weight files is : LINK
8. The best validation accuracy came out to be: 89.87%

Part 2: Testing the Neural Network (python Assignment3_14EC35011.py --test)

1. Testing accuracy came out to be: 88.53%.
2. Since the training and testing accuracy are close we can assume that overfitting has been avoided and the trained model is a good one.

Part 3: Training each layer output using logistic regression

1. The weights and biases were retrieved by reloading the session first and then accessing variables like dense_index/kernel:0 for weights and dense_index/bias:0 for biases. index can be replaced by 1,2 for layer 1-2,2-3. If there is no mention it is for layer input-1.
2. Required calculations were done and outputs for various layers were calculated.
3. The outputs were used to train a logistic regression classifier from sklearn module. The accuracies of layers 1,2,3 are as follows:
    a. Layer 1 = 81.244%
    b. Layer 2 = 82.056%
    c. Layer 3 = 82.154%
4. There is a steady increase in the accuracy due to the fact that some non-linearity is fitted by passing it through each hidden layer. There is only a marginal increase in

the accuracy due to the fact that for the given data set a single hidden layer would have sufficed as can be seen by comparing the accuracy with the previous assignment.

Link to the weight files: https://1drv.ms/u/s!AuWnQxJtgcQUiAB9trtrJbLA2L7k