CS5590 APS - Deep Learning Programming

ASSIGNMENT 1

Deadline: 4/6/2018

Task:

1. Implement the Logistic regression with new data set which is not used in class
2. Show the graph in TensorBoard
3. Change the hyperparameter and compare the result

Submission Guidelines:

1. Submit your source code and documentation to GitHub and represent the work through wiki page properly (submit your screenshots as well. The screenshot should have both the code and the output)
2. Comment your code appropriately
3. Submit **only** report at Turnitin in UMKC blackboard
4. Remember that similarity score should be less than 15%
5. Use this link to submit your assignment: https://goo.gl/forms/l9TitNZJ8yLCwGEW2
6. Report should include below details
   I. Introduction
   II. Objectives
   III. Approaches/Methods
   IV. Workflow
   V. Datasets
   VI. Parameters
   VII. Evaluation & Discussion
   VIII. Conclusion

Introduction

This project is to complete a logistic regression using TensorFlow and then plot it using TensorBoard.

Please note this is a class assignment and ht ecode used herein was largely obtained from the internet and modified. Any similarity to other code is not meant to infringe on the rights of others as this is simply a learning exercise to become familiar with the required process.

Note to instructors…I had this code running in PyCharm. When I attempted to install Anaconda in order to run TensorBoard, both PyCharm and Anaconda have ceased to function properly. Asa result, I cannot show the TensprBoard graph or the code output. I have worked for several evenings to fix this to no avail. My goal is to get it rectified by Saturday morning's class so I can complete the in-class assignment.

Objectives

The objectives are to learn TensorFlow and TensorBoard.

Approaches/Methods

Research was conducted online to find examples of the code for logistic regression for Python 3.6. Initially, PyCharm was used to runs the code for the logistic regression.

Anaconda was determined to be the best software to use to run Tensorboard. In the process of setting up Anaconda for a gpu, the Tensorflow modile for PyCharm was removed and cannot be reloaded. In addition, the setup for Anaconda is not working.

Datasets

The data set used was the input_data set from the Tensorflow example tutorials.

Parameters

The following parameters were applied to the regression.

```
learning_rate = 0.01
batch_size = 128
n_epochs = 25
logs_path = '/tmp/tensorflow_logs/example/'
```

Evaluation & Discussion

Originally the code ran in PyCharm but a screenshot of this run was not taken. Currently, PyCharm cannot find the tensorflow moduel. As a result, the following screenshot in Figure 1 was obtained from the run.

The following code was used of the run.

```
import tensorflow as tf
import matplotlib.pyplot as plt
from tensorflow.examples.tutorials.mnist import input_data
import tensorboard

MNIST = input_data.read_data_sets("/data/mnist",one_hot=True)

learning_rate = 0.01
batch_size = 128
n_epochs = 25
logs_path = '/tmp/tensorflow_logs/example/'

X = tf.placeholder(tf.float32, [batch_size, 784])
Y = tf.placeholder(tf.float32, [batch_size, 10])

w = tf.Variable(tf.random_normal(shape = [784,10], stddev=0.01),name="weights")
b = tf.Variable(tf.zeros([1,10]),name = "bias")

#Logisitc regression equation
logits = tf.matmul(X,w)+b

entropy =
```

```
tf.nn.softmax_cross_entropy_with_logits(_sentinel=None,labels=Y,logits=logits,dim=-
1,name=None)
loss = tf.reduce_mean(entropy)

optimizer =
tf.train.GradientDescentOptimizer(learning_rate=learning_rate).minimize(loss)

init = tf.global_variables_initializer()

# Runs Tensorflow to complete logistic regression and creates data to be used by
Tensorboard
with tf.Session() as sess:
    sess.run(init)
    # op to write logs to Tensorboard
    summary_writer = tf.summary.FileWriter(logs_path, graph=tf.get_default_graph())

    n_batches = int(MNIST.train.num_examples/batch_size)
    for i in range(n_epochs):
        for _ in range(n_batches):
            X_batch, Y_batch = MNIST.train.next_batch(batch_size)
            sess.run([optimizer,loss],feed_dict={X:X_batch, Y:Y_batch})
    n_batches = int(MNIST.test.num_examples/batch_size)
    total_correct_pred = 0
    for _ in range(n_batches):
        X_batch, Y_batch = MNIST.test.next_batch(batch_size)
        _, loss_batch, logits_batch = sess.run([optimizer,loss,logits],
feed_dict={X:X_batch,Y:Y_batch})
        preds = tf.nn.softmax(logits_batch)
        correct_preds = tf.equal(tf.argmax(preds,1),tf.argmax(Y_batch,1))
        accuracy = tf.reduce_sum(tf.cast(correct_preds, tf.float32))
        total_correct_pred += sess.run(accuracy)

        print("Accuracy {0}".format(total_correct_pred/MNIST.test.num_examples))

# Prints the Tensorbaard resultst to be viewed in browser
    print("Run the command line:\n" \
          "--> tensorboard --logdir=/tmp/tensorflow_logs " \
          "\nThen open http://0.0.0.0:6006/ into your web browser")
```

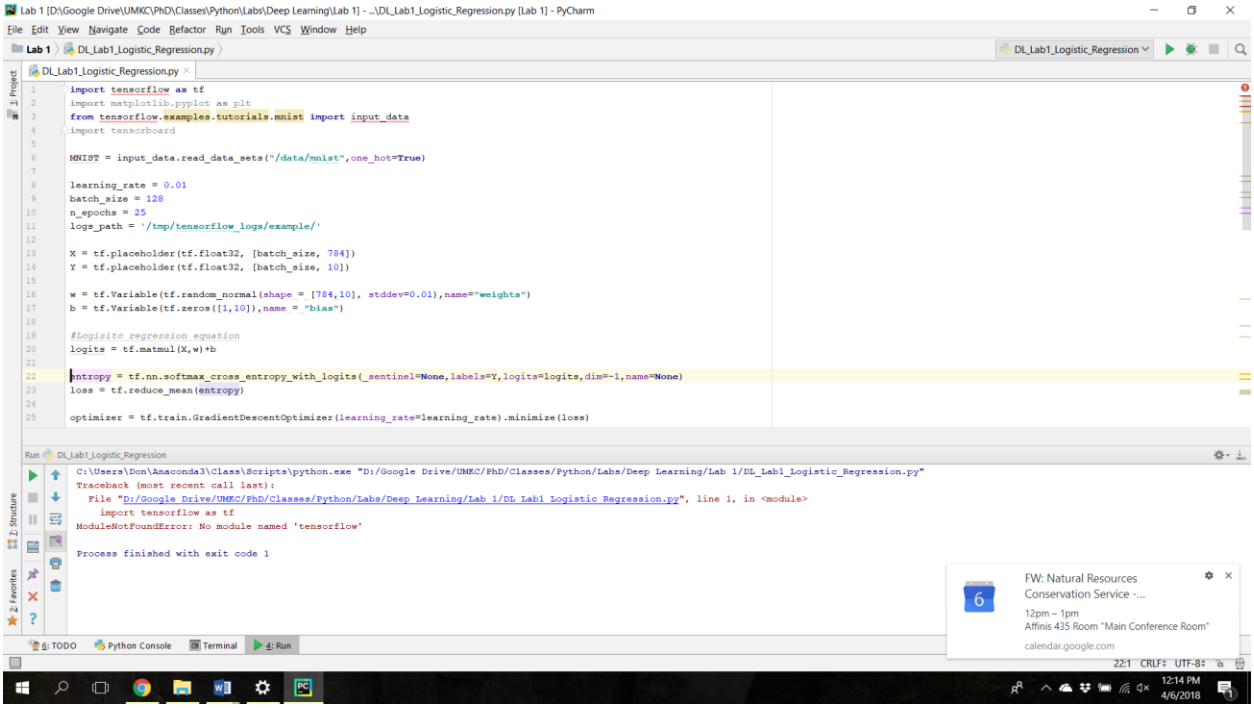

Figure 1 Screenshot from PyCharm

Conclusion

It appears from the research performed that Tensorflow is very efficient in handling very large datsets.

If everything was working I would be able to present the graph here illustrating the data and the logistic regression line.