Homework 1

Sami Ellougani

9/24/17

1 Problem 1

Run the code below for the current CNN architecture and record the time it took to train (estimate) and its final accuracy

```
for i in range(20000):
    batch = mnist.train.next_batch(50)
    if i%100 == 0:
        train_accuracy = accuracy.eval(feed_dict={
            x:batch[0], y_: batch[1], keep_prob: 1.0})
        print("step %d, training accuracy %g"%(i, train_accuracy))
    train_step.run(feed_dict={x: batch[0], y_: batch[1], keep_prob: 0.5})
```

Solution: It took 58 minutes to complete 20000 steps, with a training accuracy of 100%

2 Problem 2

Do the same but change the architecture slightly (e.g., the number of feature maps in each convolution layer), and record the time it took to train(estimate) and its final accuracy

```
#Convolutional Layer 1 feature maps cut in half
W_conv1 = tf.Variable(tf.truncated_normal([5,5,1,16], stddev=0.1))
b_conv1 = tf.Variable(tf.constant(0.1, shape=[16]))

#Convolution Layer 2 feature maps cut in half
w_conv2 = tf.Variable(tf.truncated_normal([5,5, 16, 32], stddev=0.1))
b_conv2 = tf.Variable(tf.constant(0.1, shape=[32]))

#Convolution Layer 2 feature maps cut in half
w_conv2 = tf.Variable(tf.truncated_normal([5,5, 16, 32], stddev=0.1))
b_conv2 = tf.Variable(tf.constant(0.1, shape=[32]))

#Flattening layer cut in half
layer2_matrix = tf.reshape(conv2, [-1, 7*7*32])

#Weights and Biases cut in h alf
W_fc1 = tf.Variable(tf.truncated_normal([7 * 7 * 32, 512], stddev=0.1))
b_fc1 = tf.Variable(tf.constant(0.1, shape=[512]))

#Softmax layer cut in half
W_fc2 = tf.Variable(tf.truncated_normal([512, 10], stddev=0.1))
```

Solution: It took 28 minutes to complete 20000 steps, with a training accuracy of 100%