quiz

April 13, 2020

```
In [1]: # Solution is available in the other "quiz_solution.ipynb"
        import tensorflow as tf

        def get_weights(n_features, n_labels):
            """
            Return TensorFlow weights
            :param n_features: Number of features
            :param n_labels: Number of labels
            :return: TensorFlow weights
            """
            # TODO: Return weights
            return tf.Variable(tf.truncated_normal((n_features, n_labels)))


        def get_biases(n_labels):
            """
            Return TensorFlow bias
            :param n_labels: Number of labels
            :return: TensorFlow bias
            """
            # TODO: Return biases
            return tf.Variable(tf.zeros(n_labels))

        def linear(input, w, b):
            """
            Return linear function in TensorFlow
            :param input: TensorFlow input
            :param w: TensorFlow weights
            :param b: TensorFlow biases
            :return: TensorFlow linear function
            """
            # TODO: Linear Function (xW + b)
            return tf.add(tf.matmul(input, w) , b)

In [2]: # Solution is available in the other "quiz_solution.ipynb" tab

        import tensorflow as tf
```

```
from tensorflow.examples.tutorials.mnist import input_data
from test import *

def mnist_features_labels(n_labels):
    """
    Gets the first <n> labels from the MNIST dataset
    :param n_labels: Number of labels to use
    :return: Tuple of feature list and label list
    """
    mnist_features = []
    mnist_labels = []

    mnist = input_data.read_data_sets('/datasets/ud730/mnist', one_hot=True)

    # In order to make quizzes run faster, we're only looking at 10000 images
    for mnist_feature, mnist_label in zip(*mnist.train.next_batch(10000)):

        # Add features and labels if it's for the first <n>th labels
        if mnist_label[:n_labels].any():
            mnist_features.append(mnist_feature)
            mnist_labels.append(mnist_label[:n_labels])

    return mnist_features, mnist_labels


# Number of features (28*28 image is 784 features)
n_features = 784
# Number of labels
n_labels = 3

# Features and Labels
features = tf.placeholder(tf.float32)
labels = tf.placeholder(tf.float32)

# Weights and Biases
w = get_weights(n_features, n_labels)
b = get_biases(n_labels)

# Linear Function xW + b
logits = linear(features, w, b)

# Training data
train_features, train_labels = mnist_features_labels(n_labels)

with tf.Session() as session:
    # TODO: Initialize session variables
    session.run(tf.global_variables_initializer())
    # Softmax
```

```
            prediction = tf.nn.softmax(logits)

            # Cross entropy
            # This quantifies how far off the predictions were.
            # You'll learn more about this in future lessons.
            cross_entropy = -tf.reduce_sum(labels * tf.log(prediction), reduction_indices=1)

            # Training loss
            # You'll learn more about this in future lessons.
            loss = tf.reduce_mean(cross_entropy)

            # Rate at which the weights are changed
            # You'll learn more about this in future lessons.
            learning_rate = 0.08

            # Gradient Descent
            # This is the method used to train the model
            # You'll learn more about this in future lessons.
            optimizer = tf.train.GradientDescentOptimizer(learning_rate).minimize(loss)

            # Run optimizer and get loss
            _, l = session.run(
                [optimizer, loss],
                feed_dict={features: train_features, labels: train_labels})

        # Print loss
        print('Loss: {}'.format(l))
```

```
Successfully downloaded train-images-idx3-ubyte.gz 9912422 bytes.
Extracting /datasets/ud730/mnist/train-images-idx3-ubyte.gz
Successfully downloaded train-labels-idx1-ubyte.gz 28881 bytes.
Extracting /datasets/ud730/mnist/train-labels-idx1-ubyte.gz
Successfully downloaded t10k-images-idx3-ubyte.gz 1648877 bytes.
Extracting /datasets/ud730/mnist/t10k-images-idx3-ubyte.gz
Successfully downloaded t10k-labels-idx1-ubyte.gz 4542 bytes.
Extracting /datasets/ud730/mnist/t10k-labels-idx1-ubyte.gz
Loss: 12.332940101623535
```

0.0.1 Running the Grader

To run the grader below, you'll want to run the above training from scratch (if you have otherwise already ran it multiple times). You can reset your kernel and then run all cells for the grader code to appropriately check that you weights and biases achieved the desired end result.

```
In [3]: ### DON'T MODIFY ANYTHING BELOW ###
        ### Be sure to run all cells above before running this cell ###
        import grader
        try:
            grader.run_grader(get_weights, get_biases, linear)
        except Exception as err:
            print(str(err))
```

```
You got it!  That's the correct answer.
```

```
In [ ]:
```