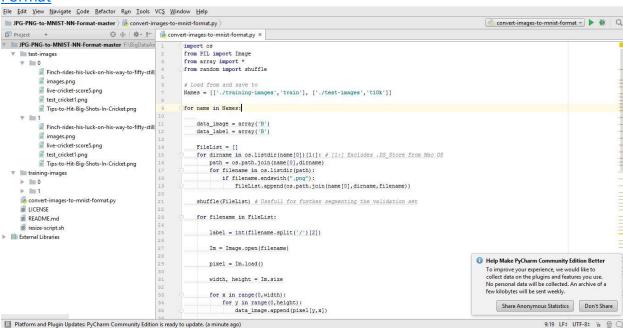
CS5542 Big Data Analytics and App LAB ASSIGNMENT #9

CNN model using tensor Tensor Flow

The CNN model for this lab is run on sport-cricket images. These images were collected from the internet and are resized into **28x28 pixel** size. Total of 15 images were collected for the data set, out of which 10 images were put into train images and 5 images were put into test images. These images were converted into .gz compressed format using a python program from https://github.com/gskielian/JPG-PNG-to-MNIST-NN-

Format



```
<u>File Edit View Navigate Code Refactor Run Jools VCS Window Help</u>
   🛑 convert-images-to-mnist-format 🔻 🕨 🗰
                                                                              ② 🛊 🔅 - I⁺ 🖟 convert-images-to-mnist-format.py ×
  Project
                                                                                                                                                   header.extend([0,0,8,1,0,0])
▼ Im JPG-PNG-to-MNIST-NN-Format-master F:\BigDataAn 44
                                                                                                                                                   header.append(int('0x'+hexval[2:][:2],16))
header.append(int('0x'+hexval[2:][2:],16))
             test-images
             w = 0
                             Finch-rides-his-luck-on-his-way-to-fifty-still
                                                                                                                                                   data_label = header + data_label
                             images.png
                                                                                                                                               # additional header for images array
                             live-cricket-score5.png
                              test_cricket1.png
                                                                                                                                                   if max([width,height]) <= 256:
    header.extend([0,0,0,width,0,0,0,height])</pre>
                            Tips-to-Hit-Big-Shots-In-Cricket.png
               v 🖿 1
                             Finch-rides-his-luck-on-his-way-to-fifty-still 55
                                                                                                                                                           raise ValueError('Image exceeds maximum size: 256x256 pixels');
                            images.png
                                                                                                                                                   header[3] = 3 # Changing MSB for image data (0x00000803)
                             live-cricket-score5.png
                             test_cricket1.png
                                                                                                                                                   data_image = header + data_image
                            Tips-to-Hit-Big-Shots-In-Cricket.png
        ▼ li training-images
                                                                                                                                                    output_file = open(name[1]+'-images-idx3-ubyte', 'wb')
              ▶ ■ 0
                                                                                                                                                   data_image.tofile(output_file)
output_file.close()
                 h 1m1
               💪 convert-images-to-mnist-format.py
                                                                                                                                                 output_file = open(name[1]+'-labels-idx1-ubyte', 'wb')
               ILICENSE
                                                                                                                                                   data_label.tofile(output_file)
output_file.close()
               resize-script.sh
 ▶ IIII External Libraries
                                                                                                                                           # gzip resulting files
                                                                                                                                          for name in Names.
                                                                                                                                                   os.system('gzip '+name[1]+'-images-idx3-ubyte')
os.system('gzip '+name[1]+'-labels-idx1-ubyte')
Platform and Plugin Updates: PyCharm Community Edition is ready to update. (2 minutes ago)

EXTRACTING MUISI Gata/Train-liaBels-idx1-ubyte. gz

Extracting MuIST data/Train-labels-idx1-ubyte. gz

EXTRACTING MUIST data/Tlbk-liaBels-idx3-ubyte. gz

EXTRACTING MUIST data/Tlbk-labels-idx1-ubyte. gz

EXTRACTING MUIST data/Tlbk-labels-idx1-ubyte. gz
                                                                                                                                                                                                                                                                                                                                                                                                                            9:19 LF¢ UTF-8¢ % 🚇 🕦
Extracting MINIST dataflels-labels-indu-upyte.gc
Extracting MINIST dataflels-labels-indu-upyte.gc
W tensorflow/core/platform/cpu feature guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX instructions, but these are available on your machine and could speed up CPU computations.
W tensorf
  test accuracy 1
Time for building convnet:
5383
```

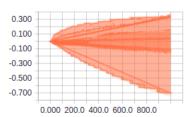
Process finished with exit code $\boldsymbol{\theta}$

```
NN - [-/Documents/MNIST_CNN] - .../CNNmodel.py - PyCharm Community Edition 2016.3.2
ile Edit View Navigate Code Refactor Ryn Jools VCS Window Help
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        En (3:10,50%) ◆II) 12:48 PM ♦
0
                               MNIST_CNN ) & CNNmodel.py )
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  CNNmodel - > *
                                                                                                                                                                                  ⊕ + + + E CNNmodel.py ×
                                   MNIST_CNN -/Doc
                                                                                                               nts/MNIST CNN
                                                                                                                                                                                                                                                        def conv2d(x, W):
    return tf.nn.conv2d(x, W, strides=[1, 1, 1, 1], padding='SAME')
                                             MNIST_data

t10k-images-idx3-ubyte.gz
t10k-labels-idx1-ubyte.gz
train-images-idx3-ubyte.gz
train-labels-idx1-ubyte.gz
                                                                                                                                                                                                                                                      def max_pool_Zx2(x):
    return tf.nn.nax_pool(x, ksize=[1, 2, 2, 11, strides=[1, 2, 2, 11, padding='SAME')
W_conv1 = weight_variable([5, 5, 1, 32])
b_conv1 = bias_variable([32])
x_image = tf.reshape(x, [-1, 28, 28, 1])
h_conv1 = tf.ne.relu(conv2d(x_image, W_conv1) + b_conv1)
h_pool1 = max_pool_2x2(h_conv1)
                                                                                                                                                                                                                                                       W_conv2 = weight_variable([5, 5, 32, 64])
b_conv2 = bias_variable([64])
                                                                                                                                                                                                                                                       h_conv2 = tf.nn.relu(conv2d(h_pool1, W_conv2) + b_conv2)
h_pool2 = max_pool_2x2(h_conv2)
                                                                                                                                                                                                                                                         W_fcl = weight_variable([7 * 7 * 64, 1024])
b_fcl = bias_variable([1024])
                                                                                                                                                                                                                                                         h pool2_flat = tf.reshape(h_pool2, [-1, 7 * 7 * 64])
h_fcl = tf.nn.relu(tf.matmul(h_pool2_flat, W_fcl) + b_fcl)
 0
                                                                                                                                                                                                                                                       keep_prob = tf.placeholder(tf.float32)
h_fcl_drop = tf.nn.dropout(h_fcl, keep_prob)
                                                                                                                                                                                                                                                       W_fc2 = weight_variable([1024, 10])
b fc2 = bias_variable([10])
                                                                                                                                                                                                                                                      y_conv = tf.matmul(h_fcl_drop, W fc2) + b_fc2
cross_entropy = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(logits=y_conv, labels=y_))
                                                                                                                                                                                                                                                        train_step = tf.train.AdamOptimizer(1e-4).minimize(cross_entropy)
correct_prediction = tf.equal(tf.argmax(y_conv, 1), tf.argmax(y_, 1))
accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
                                                                                                                                                                                                                                                       accuracy = ft.reduce monitr.est(correct_prediction, ft.ftcat2))
ff.sumary_scalif_cross_attropy, ross_entropy)
ff.sumary_histoprad(cross_hist', cross_entropy)
ff.sumary_histoprad(cross_hist', cross_entropy)
morped = ff.sumary_histoprad(cross_hist', cross_entropy)
for traincriter = ff.sumary_fileWriter(dataflogs', sess_graph)
sess_unitf.globul_variables_initializer())
for traincriter.add_sumary_scalif_cross_entropy
for = ft.sumary_filewriter(fataflogs', sess_graph)
sumary_ = sess_unif(erged_filex_filex_hist)
fraincriter.add_sumary[sumary_t]
ff i % 10 = ft.sumary_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_filex_fil
                                                                                                                                                                                                                                                        print("test accuracy % accuracy.eval(feed_dict={x: mist.test.images, y_: mist.test.labels, keep_prob: 1.0}))
end = int(round(time.time() * 1000))
print("Time for building commet: ")
print(end - start)
                                     CNNmodel
//usr/hin/nuthon3 5 /home/harath/Documents/MMTST_CNN/CNNmodel_nv.
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          27:1 CRLF: UTF-8: % @ Q
```

bias 1

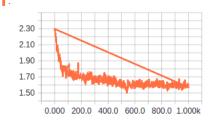
bias .



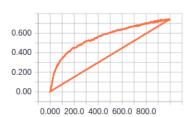
03

cross_hist 1

cross_hist



max_weight

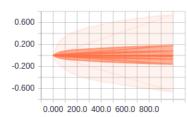


00

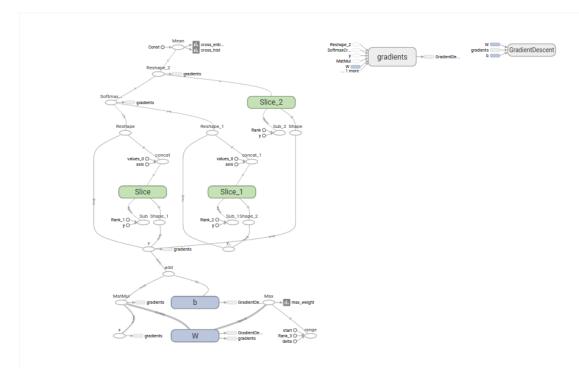
weights

1

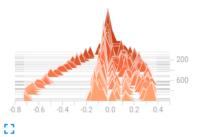
weights



00

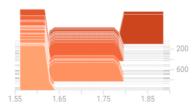


bias .

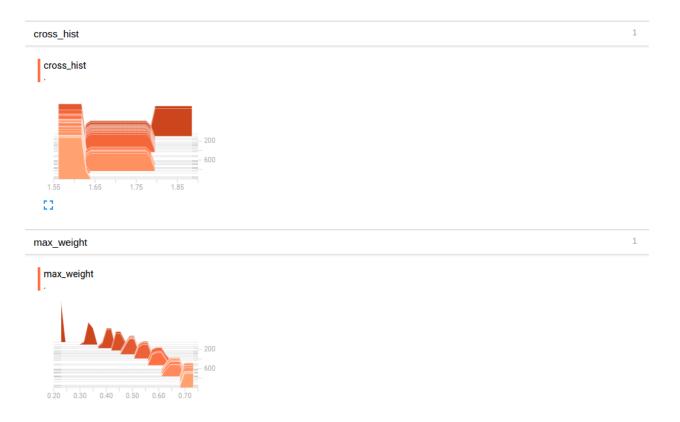


cross_hist

cross_hist



0



The accuracy came out as 100% because only 15 images were taken in total and as Tensor flow CNN model already generated high accuracy, the accuracy came out to be 100%.