

[Features](#) [Explore](#) [Pricing](#)

This repository

[Sign in](#) or [Sign up](#) [tensorflow](#) / [tensorflow](#) Watch



4,540

 Star

49,528

 Fork

23,097

 Code Issues 828 Pull requests 58 Projects 0 Pulse Graphs

Branch: master ▾

[tensorflow](#) / [tensorflow](#) / [examples](#) / [tutorials](#) / [mnist](#) / [mnist\\_softmax.py](#)

Find file

Copy path

**martinwicke** Change arg order for {softmax,sparse\_softmax,sigmoid}\_cross\_entropy\_w...

333dc32 on 5 Jan

6 contributors



80 lines (65 sloc) 2.68 KB

Raw

Blame

History



```
1  # Copyright 2015 The TensorFlow Authors. All Rights Reserved.
2  #
3  # Licensed under the Apache License, Version 2.0 (the "License");
4  # you may not use this file except in compliance with the License.
5  # You may obtain a copy of the License at
6  #
7  #     http://www.apache.org/licenses/LICENSE-2.0
8  #
9  # Unless required by applicable law or agreed to in writing, software
10 # distributed under the License is distributed on an "AS IS" BASIS,
11 # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12 # See the License for the specific language governing permissions and
13 # limitations under the License.
14 # =====
15
16 """A very simple MNIST classifier.
17
```

```
18 See extensive documentation at
19 http://tensorflow.org/tutorials/mnist/beginners/index.md
20 """
21 from __future__ import absolute_import
22 from __future__ import division
23 from __future__ import print_function
24
25 import argparse
26 import sys
27
28 from tensorflow.examples.tutorials.mnist import input_data
29
30 import tensorflow as tf
31
32 FLAGS = None
33
34
35 def main(_):
36     # Import data
37     mnist = input_data.read_data_sets(FLAGS.data_dir, one_hot=True)
38
39     # Create the model
40     x = tf.placeholder(tf.float32, [None, 784])
41     W = tf.Variable(tf.zeros([784, 10]))
42     b = tf.Variable(tf.zeros([10]))
43     y = tf.matmul(x, W) + b
44
45     # Define loss and optimizer
46     y_ = tf.placeholder(tf.float32, [None, 10])
47
48     # The raw formulation of cross-entropy,
49     #
50     #   tf.reduce_mean(-tf.reduce_sum(y_ * tf.log(tf.nn.softmax(y)),
51     #                                   reduction_indices=[1]))
52     #
53     # can be numerically unstable.
```

```
54 #
55 # So here we use tf.nn.softmax_cross_entropy_with_logits on the raw
56 # outputs of 'y', and then average across the batch.
57 cross_entropy = tf.reduce_mean(
58     tf.nn.softmax_cross_entropy_with_logits(labels=y_, logits=y))
59 train_step = tf.train.GradientDescentOptimizer(0.5).minimize(cross_entropy)
60
61 sess = tf.InteractiveSession()
62 tf.global_variables_initializer().run()
63 # Train
64 for _ in range(1000):
65     batch_xs, batch_ys = mnist.train.next_batch(100)
66     sess.run(train_step, feed_dict={x: batch_xs, y_: batch_ys})
67
68 # Test trained model
69 correct_prediction = tf.equal(tf.argmax(y, 1), tf.argmax(y_, 1))
70 accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
71 print(sess.run(accuracy, feed_dict={x: mnist.test.images,
72                                     y_: mnist.test.labels}))
73
74 if __name__ == '__main__':
75     parser = argparse.ArgumentParser()
76     parser.add_argument('--data_dir', type=str, default='/tmp/tensorflow/mnist/input_data',
77                         help='Directory for storing input data')
78     FLAGS, unparsed = parser.parse_known_args()
79     tf.app.run(main=main, argv=[sys.argv[0]] + unparsed)
```

