マイ Mnist - 0

TensorFlow + Keras 2.0 紹介
2017-4-1

目的

- MNISTを使ってシンプルなニューラルネットワークを作ること

内容は添っているpythonファイルとともに使われる考えで作りました。
(内容はほとんど全部が
https://blog.keras.io/keras-as-a-simplified-interface-to-tensorflow-tutorial.html)
により拝借しました)

インポート + セッション開始

```
import tensorflow as tf
from tensorflow.examples.tutorials.mnist import input_data

from keras import backend as K
sess = tf.Session()
K.set_session(sess)
```

Tensorflowの個性

- データと答えのプレースホルダーのシンボルを作ります

```
# mnist images are 28**2 = 784 pixels, and 10 classes
img = tf.placeholder(tf.float32, shape=(None, 784))
labels = tf.placeholder(tf.float32, shape=(None, 10))
```

始めのモデル

```
# make model
x = Dense(32, activation='relu')(img)
preds = Dense(10, activation='softmax')(x)
```

- 注意：sigmoid活性か関数ではなく、reluを使います

損失と精度を定義します

```
# 損失を定義します
loss = tf.reduce_mean(categorical_crossentropy(labels, preds))
```

- ほぼ微分可能関数を使ってニューラルネットワークをエラーにより調整します

```
# 精度を定義します
acc_value = tf.reduce_mean(categorical_accuracy(labels, preds))
```

- 人間が使えるモデルの評価

Tensorflowの個性

```
# 変数をイニシャライズします
init_op = tf.global_variables_initializer()
sess.run(init_op)
```

- 定義したTensorflowのシンボルを使う前にこうします

魔法

```
for iter in range(num_iter):
  batch = mnist_data.train.next_batch(batch_size)
  train_step.run(feed_dict={
      img: batch[0],
      labels: batch[1]
  })
```

- KerasよりTensorflowの方が手で訓練を設定しなければいけません

成功をターミナルに表示します

```
print('train acc = {:.4f}'.format(
  acc_value.eval(feed_dict={
      img: mnist_data.train.images,
      labels: mnist_data.train.labels
  })
))
```

- バッチのあとの訓練精度

```
print('valid acc = {:.4f}'.format(
  acc_value.eval(feed_dict={
      img: mnist_data.test.images,
      labels: mnist_data.test.labels
  })
))
```

- バッチのあとのテスト精度

精度が約0.85-0.90
になるはずです