

原

TensorFlow基础教程：搭建卷积神经网络CNN

2018年01月28日 10:58:32 阅读数：161

卷积神经网络 CNN tensorflow 深度学习 python 更多

个人分类： tensorflow python

版权声明：本文为博主原创文章，未经博主允许不得转载。 [https://blog.csdn.net/cetrol\\_chen/article/details/79182964](https://blog.csdn.net/cetrol_chen/article/details/79182964)

手把手教你使用TensorFlow搭建卷积神经网络

TensorFlow版本1.4.0  
python版本>3.5.0

卷积神经网络的原理大家可以参考[这篇文章](#)

本教程使用LeNet网络对MNIST数据集进行分类。

LeNet基本结构如下

输入—>卷积层C1—>池化层P1—>卷积层C2—>池化层P2—>全连接层F1—>全连接层F2(输出)

输入参数

输入图像大小**28\*28**

卷积层C1参数

卷积核大小**5\*5**，步长为**1**，输出通道**20**  
输出大小为 $(28-5+1)*(28-5+1)*20=24*24*20$

池化层P1参数

池化核大小**2\*2**，步长为**2**  
输出大小为 $(24/2)*(24/2)*20=12*12*20$

卷积层C2参数

卷积核大小**5\*5**，步长为**1**，输出通道**50**  
输出大小为 $(12-5+1)*(12-5+1)*50=8*8*50$

池化层P2参数

池化核大小**2\*2**，步长为**2**  
输出大小为 $(8/2)*(8/2)*50=4*4*50=800$

全连接层F1参数

输出神经元个数**500**

全连接层F2(输出)参数

输出神经元个数**10**

了解了LeNet网络结构之后，就可以动手编写代码了

1.载入MNIST数据集

```

1 from tensorflow.examples.tutorials.mnist import input_data
2 mnist = input_data.read_data_sets("./tmp/data/", one_hot=True)
3 import tensorflow as tf

```

## 2.定义参数

```

1 learning_rate = 0.001
2 train_epochs = 1
3 batch_size = 64
4 n_input = 784
5 n_classes = 10

```

## 3.定义网络输入参数

```

1 x = tf.placeholder(tf.float32, shape=[None, n_input])
2 y = tf.placeholder(tf.float32, shape=[None, n_classes])
3 keep_prob = tf.placeholder(tf.float32) #采用dropout, 防止过拟合

```

## 4.定义权重与偏置

```

1 #权重的形状为[kernel_size, kernel_size, in_channels, out_channels]
2 #------卷积核高-----卷积核宽-----输入通道数-----输出通道数--
3 weights = {'wc1': tf.Variable(tf.random_normal([5,5,1,20])),
4            'wc2': tf.Variable(tf.random_normal([5,5,20,50])),
5            'wf1': tf.Variable(tf.random_normal([4*4*50, 500])),
6            'wf2': tf.Variable(tf.random_normal([500, 10]))}
7
8 biases = {'bc1': tf.Variable(tf.random_normal([20])),
9           'bc2': tf.Variable(tf.random_normal([50])),
10          'bf1': tf.Variable(tf.random_normal([500])),
11          'bf2': tf.Variable(tf.random_normal([10]))}

```

## 5.定义前向推断过程

```

1 def inference(x):
2     #将图片大小变为[batch_size, height, width, channels]
3     #------训练个数-----高-----宽-----通道---
4     x = tf.reshape(x, shape=[-1, 28, 28, 1])
5
6     #步长stride中间两个维度表示高和宽, 其他两个维度默认为1即可
7     #卷积层C1
8     conv1 = tf.nn.conv2d(x, weights['wc1'], strides=[1, 1, 1, 1], padding='VALID')
9     conv1 = tf.nn.bias_add(conv1, biases['bc1'])
10    #池化层P1
11    conv1 = tf.nn.max_pool(conv1, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='VALID')
12
13    #卷积层C2
14    conv2 = tf.nn.conv2d(conv1, weights['wc2'], strides=[1, 1, 1, 1], padding='VALID')
15    conv2 = tf.nn.bias_add(conv2, biases['bc2'])
16    #池化层P2
17    conv2 = tf.nn.max_pool(conv2, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='VALID')
18
19    #将4*4*50变为800
20    fc1 = tf.reshape(conv2, [-1, weights['wf1'].get_shape().as_list()[0]])
21    #全连接层F1
22    fc1 = tf.nn.xw_plus_b(fc1, weights['wf1'], biases['bf1'])
23    fc1 = tf.nn.relu(fc1)
24    #dropout层,dropout原理参考https://yq.aliyun.com/articles/68901
25    fc1 = tf.nn.dropout(fc1, keep_prob)
26    #全连接层F2(输出)
27    out = tf.nn.xw_plus_b(fc1, weights['wf2'], biases['bf2'])
28    return out

```

## 6.构建网络

```

1 logits = inference(x)
2 prediction = tf.nn.softmax(logits)

```

7.定义损失函数与优化器

```
1  loss_op = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(logits=logits, labels=y))
2  optimizer = tf.train.AdamOptimizer(learning_rate=learning_rate)
3  train_op = optimizer.minimize(loss_op)
```

8.定义评价指标

```
1  pre_correct = tf.equal(tf.argmax(prediction, 1), tf.argmax(prediction, 1))
2  accuracy = tf.reduce_mean(tf.cast(pre_correct, tf.float32))
```

9.开始训练

```
1  init = tf.global_variables_initializer()
2  with tf.Session() as sess:
3      sess.run(init)
4      total_batch = int(mnist.train.num_examples / batch_size)
5      for epoch in range(train_epochs):
6          for batch in range(total_batch):
7              batch_x, batch_y = mnist.train.next_batch(batch_size)
8              sess.run(train_op, feed_dict={x:batch_x, y:batch_y, keep_prob:0.8})
9
10             if batch % 80 == 0:
11                 loss, acc = sess.run([loss_op, accuracy], feed_dict={x:batch_x, y:batch_y, keep_prob:1.0})
12                 print("epoch {}, loss {:.4f}, accuracy {:.3f}".format(epoch, loss, acc))
13
14             print("optimization finished!")
15
16             #在测试集上测试
17             test_acc = sess.run(accuracy, feed_dict={x:mnist.test.images, y:mnist.test.labels, keep_prob:1.0})
18             print('test accuracy', test_acc)
```

只训练1轮就达到了93.39%  
github源码下载  
[https://github.com/gamersover/tensorflow\\_basic\\_tutorial/blob/master/basic\\_model/cnn\\_mnist.py](https://github.com/gamersover/tensorflow_basic_tutorial/blob/master/basic_model/cnn_mnist.py)



想对作者说点什么

**tensorflow之简单卷积神经网络（CNN）搭建**

 1106


本篇文章采用cnn搭建来实现对mnist数据集的分类，增加了卷积运算。

**机器学习：Tensor Flow with CNN 做表情识别**

 1万


我们利用 TensorFlow 构造 CNN 做表情识别，我们用的是FER-2013 这个数据库，这个数据库一共有 35887 张人脸图像，这里只是做...

**用Tensorflow搭建CNN卷积神经网络，实现MNIST手写数字识别**

 1367


写在前面的话不同于Tensorflow官方教程简略的DEMO，我们自己动手实现以下目标 - 从本地文件系统中加载图片、标签 - 对图片和标...

**深度学习之卷积神经网络CNN及tensorflow代码实现示例**

 10.1万

一、CNN的引入在人工的全连接神经网络中，每相邻两层之间的每个神经元之间都是有边相连的。当输入层的特征维度变得很高时，...

**卷积神经网络CNN原理以及TensorFlow实现**

 3.3万

在知乎上看到一段介绍卷积神经网络的文章，感觉讲的特别直观明了，我整理了一下。首先介绍原理部分。 通过一个图像分类问...

**TensorFlow搭建CNN卷积神经网络**

 704

TensorFlow搭建CNN卷积神经网络 该教程采用TernsorFlow搭建CNN卷积神经网络，并利用MNIST数据集进行数字的手写识别...

**03：一文全解：使用Tensorflow搭建卷积神经网络CNN识别手写数字图片**

 5358

标签（空格分隔）： 王小草Tensorflow笔记笔记整理者：王小草 笔记整理时间：2017年2月25日 官方文档原文地址：https://www.tens...

**TensorFlow实战四：实现简单的卷积神经网络（CNN）**

 427

# -\*- coding: utf-8 -\*- &quot;&quot;&quot; # 载入MNIST数据集 import input\_data import tensorflow as tf ...