



01 cnn的简单介绍 结构,卷积,池化

102 代码分享 Tensorflow版本分享。

训练中可能出现的问题 过拟合和欠拟合





问题1:人是如何区分不同的图片

卷积----cnn中的特征寻找器

池化---cnn中的降维方式

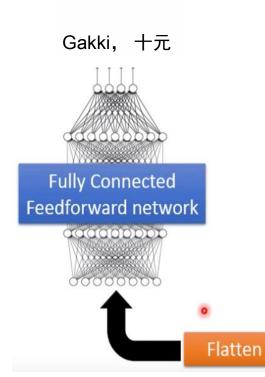


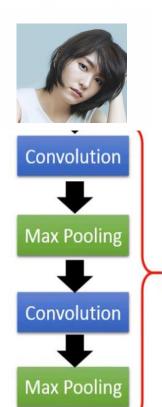






# The whole CNN





三个部分: input hidden output

Can repeat many times

池化层石接在卷积层之后的

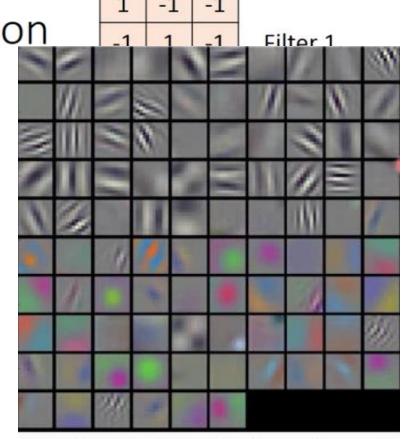
Created with EverCam.

CNN – Convolution

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image



http://cs231n.github.io/understanding-cnn/



# 真实图片







```
[[[255 255 255 255]
  [255 255 255 255]
  [255 255 255 255]
  [255 255 255 255]
  [255 255 255 255]
  [255 255 255 255]]
 [[255 255 255 255]
  [255 255 255 255]
  [255 255 255 255]
  [255 255 255 255]
  [255 255 255 255]
  [255 255 255 255]]
```

# Output

```
[0,0,1,0,0,0,0,0,0,0...]
```

## 图片存贮方式

- ▶ test ▶ trainning
- 1:4

### 输入和输出

```
INPUT = tf.placeholder(tf.float32, [None, IMAGE_HEIGHT * IMAGE_WIDTH])
OUTPUT = tf.placeholder(tf.float32, [None, MAX_CAPTCHA * CHAR_SET_LEN])
keep_prob = tf.placeholder(tf.float32)
```

### 急藏层的编写

```
def crack_captcha_cnn(w_alpha=0.01, b alpha=0.1):
   # 3 conv layer batchsize大小-1 代表有n个自动计算匹配
   x = tf.reshape(INPUT, shape=[-1, IMAGE_HEIGHT, IMAGE_WIDTH, 1])
   w c1 = tf.Variable(w alpha * tf.random normal([3, 3, 1, 32]))
   b c1 = tf.Variable(b alpha * tf.random normal([32]))
   conv1 = tf.nn.elu(#[batch,height,weight,channels]
       tf.nn.bias_add(tf.nn.conv2d(x, w_c1, strides=[1, 1, 1, 1], padding='SAME'), b_c1)) # out size = 80*160*32
   conv1 = tf.nn.max_pool(conv1, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='SAME') # out_size = <math>40*80*32
   # dropout的作用是防止过拟合按照一定的概率将一部分神经网络单元暂时从网络中丢弃
   conv1 = tf.nn.dropout(conv1, keep prob)
   w c2 = tf.Variable(w alpha * tf.random normal([3, 3, 32, 64]))#大小input, output
   b c2 = tf.Variable(b alpha * tf.random normal([64]))
   conv2 = tf.nn.elu(
       tf.nn.bias_add(tf.nn.conv2d(conv1, w_c2, strides=[1, 1, 1, 1], padding='SAME'), b_c2)) # out_size = 40*80*64
   conv2 = tf.nn.max_pool(conv2, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='SAME') # out size = 20*40*64
   conv2 = tf.nn.dropout(conv2, keep_prob)
   w c3 = tf.Variable(w alpha * tf.random normal([3, 3, 64, 64]))
   b c3 = tf.Variable(b alpha * tf.random normal([64]))
   conv3 = tf.nn.elu(
       tf.nn.bias_add(tf.nn.conv2d(conv2, w_c3, strides=[1, 1, 1, 1], padding='SAME'), b_c3)) # out_size = 20*40*64
   conv3 = tf.nn.max_pool(conv3, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='SAME')  # out size = 10*20*64
   conv3 = tf.nn.dropout(conv3, keep prob)
   # Fully connected laver
   w d = tf.Variable(w alpha * tf.random normal([10 * 20 * 64, 1024]))
   b d = tf.Variable(b alpha * tf.random normal([1024]))
   # 将这一层的结果变平
   dense = tf.reshape(conv3, [-1, w_d.get_shape().as_list()[0]]) #将10, 20, 64 变成10*20*64
   dense = tf.nn.elu(tf.add(tf.matmul(dense, w d), b d))
   dense = tf.nn.dropout(dense, keep_prob)
   w out = tf.Variable(w alpha * tf.random normal([1024, MAX CAPTCHA * CHAR SET LEN]))
   b out = tf.Variable(b alpha * tf.random normal([MAX CAPTCHA * CHAR SET LEN]))
   out = tf.add(tf.matmul(dense, w out), b out)
   return out
```

#### 存储和其他

```
def begin to train(restudy=False):
    # predictioner = layer init()
    predictioner = crack captcha cnn()
    # sigmoid_cross_entropy_with_logit是用于计算交叉熵的
    cross entropy = tf.reduce mean(tf.nn.sigmoid cross entropy with logits(logits=predictioner, labels=0UTPUT))
    # 优化器
    train step = tf.train.AdamOptimizer(1e-4).minimize(cross entropy)
    print(predictioner)
    prediction = tf.reshape(predictioner, [-1, MAX CAPTCHA, CHAR SET LEN])
    # print(prediction)
    max idx p = tf.argmax(prediction, 2)
    # print(max idx p)
    \max_{l} = \text{tf.argmax}(\text{tf.reshape}(\text{OUTPUT}, [-1, MAX_CAPTCHA, CHAR SET LEN]), 2)
    # print(max_idx_l)
    correct_pred = tf.equal(max_idx_p, max_idx_l)
    accuracy = tf.reduce_mean(tf.cast(correct_pred, tf.float32))
    saver = tf.train.Saver()
    with tf.Session() as sess:
        if restudy:
            saver.restore(sess, tf.train.latest_checkpoint('my_model'))
        sess.run(tf.global_variables_initializer())
        step = 0
            batch x, batch y = get one batch(256)
            _, new_cross_entropy = sess.run([train_step, cross_entropy],
                                             feed_dict={INPUT: batch_x, OUTPUT: batch_y, keep_prob: 0.5}) # 越小收敛的越慢
            if step % 100 == 0:
                # 读自己
                batch_x_test, batch_y_test = get_one_batch(512, select=1)
                acc = sess.run(accuracy, feed_dict={INPUT: batch_x_test, OUTPUT: batch_y_test, keep_prob: 1.})
                print('train step is{}, lost is{}, acc is{}'.format(step, new_cross_entropy, acc))
                if acc > 0.97:
                    path = 'my model/crack capcha num.ckpt'
                    saver.save(sess, path, global step=step)
                    print('wegotone')
                    break
            step += 1
```





#### 过拟合

过拟合是指样本在我们的traing库中效果很好但是在test库里很差,主要的原因是样本有问题



#### 欠拟合

欠拟合就是是指模型不收敛,原因在于模型写法有问题,激活函数有问题,样本有问题



图片的预处理

大小,格式



Thank you for your download to watch

https://www.jianshu.com/u/7109af2617fd