# Tensorflow中Attention机制和实验

今天在写Attention结构时，突然想知道tensorflow内部点乘法（非矩阵乘法）的具体实现，于是自己写了一段代码测试，并结合代码解释下。

1. **Attention机制**

首先简单讲解下Attention机制。Attention机制在近几年的深度学习模型中可谓是刷分利器，万物皆可Attention。那么在卷积神经网络（CNN）中，Attention是怎么工作的呢？在深度学习模型中，一般从深度和宽度两个方面增加模型的复杂度，深度自然是指模型层数，在Resnet框架出来之后，目前模型可以叠到1000多层（针对大型数据集，小型数据集容易出现过拟合）。而宽度指的是每层卷积核的个数，如Res-34最后一层卷积核为512，Res-50，101，152等最后一层为2048。在CNN中，卷积被用于提取图像特征，然后各个卷积核提取到的特征并未有区别化。比如人眼看图1时，会自动聚焦到人物轮廓上，即赋予人物较大权重，实际上卷积就是一种赋权方式，也有论文基于此提出Spatial Attention结构。而这次所讲的是通道Attention结构。



图1 四代火影

所谓通道Attention，即对不同特征图像赋予不同的权重。如图2所示，左侧每层不同颜色表示不同的特征图，右侧表示不同权重，每种颜色代表一个权重值。通过左右相乘，得到不同的特征权重。计算过程中使用到了广播，左边一层特征图大小为w\*h（二维张量），右边一层为单一标量（一个数值），相乘即左边每个元素与右边标量相乘。

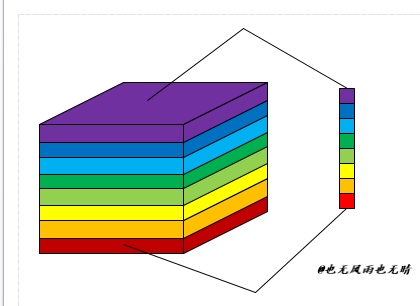


图2 通道注意力机制

1. **Tensorflow中attention的实现机制**

设某个卷积我们得到的张量形状为T（b, w, h, d）, b为batch\_size，w和h为特征图T的长和宽，d为特征图个数，即为深度。首先我们使用（b, w, h, c）形状的池化核对特征图T进行池化，得到（b, 1, 1, c）类似于全连接的层，c为全连接神经元个数，然后再通过一层全连接将全连接层映射回（b, d），并reshape成F（b, 1, 1, d），并使用softmax或者sigmoid等函数对F进行归一化。最后使用tensorflow点乘T\*F得到与原特征图大小一致的Attention图A（b , w, h, d）。

1. **Tensorflow实例**

代码：

#attention

def multiply():

tem\_list = [x for x in range(36)]

c1 = tf.constant(tem\_list, shape=[3,2,2,3])

c2 = tf.constant([0,1,2,3,4,5,6,7,8],shape=[3,1,1,3])

tensor1 = c1\*c2

sess=tf.Session()

with sess.as\_default():

print('c1:', c1.eval())

print('c1:', c2.eval())

print('结果是：', tensor1.eval())

首先吹一波python的优雅，建立一个序数列表只需要一行代码tem\_list = [x for x in range(36)]，此程序的运行结果为：

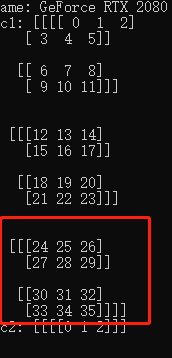
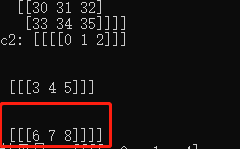
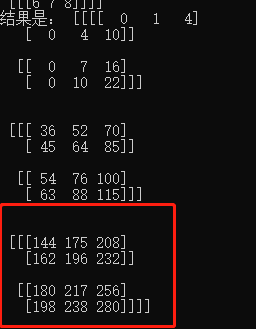
  

图3 运行结果

为了复现2中的计算过程，我创建了一个大小为（3, 2, 2, 3）的张量C1，同时创建了一个（3, 1, 1 ,3）的张量C2，两者进行点乘得到结果tensor1。解释下结果，我们首先抛开第一个维度3，只关注后面三维的数据，红框圈出的表示一个bacth中的C1，C2，C3数据，其中C1维度是（2,2,3），C2（1, 1,3）,tensor1（2,2,3）,对于这个数据点乘计算是，8\*35,8\*32,8\*29,8\*26。其含义是用标量8\*维度为（2,2）的二维特征图，其他维度依次类推，便可得到tensor1的结果。对于矩阵不熟悉的建议多输出矩阵，多看数据与形状之间的关系便可了解。

1. **结语**

这篇博客主要是讲Attention机制的理解及其在Tensorflow中的实现，并辅助代码理解tensorflow中张量点乘的实例计算方法。本博客仅仅代表个人理解，不具备任何教学和其他作用。如有错误，欢迎留言指正，也欢迎使用邮箱交流。如有转载，请注明出处。