一般我们认为在计算卷积时，是卷积核与图像中每个mxm大小的图像块做element-wise相乘，然后得到的结果相加得到一个值，然后再移动一个stride，做同样的运算，直到整副输入图像遍历完，上述过程得到的值就组成了输出特征。但是这样运算比较慢，我们用的深度学习框架可不是这么实现的。因此，我们来学习一下tensorflow和pytorch中是如何实现卷积操作的。

1、tf.nn.conv2d()

def
conv2d(input,filter,strides,padding,use_cudnn_on_gpu=True,data_format="NHWC",dilations=[1,1,1,1],name=None):

给定 4-D input 和 filter tensors计算2-D卷积.

其中，input tensor 的 shape是: [B, H, W, C]

filter / kernel tensor 的 shape是: [filter_height, filter_width, in_channels, out_channels]

这个op是这样执行的:

将filter 展开为一个 shape 为[filter_height * filter_width * in_channels, out_channels]大小的2-D 矩阵。

从 input tensor按照每个filter位置上提取图像patches来构成一个虚拟的shape大小为[batch, out_height, out_width,filter_height * filter_width * in_channels]的tensor 。

ps:把输入图像要经行卷积操作的这一区域展成列向量的操作通常称为im2col

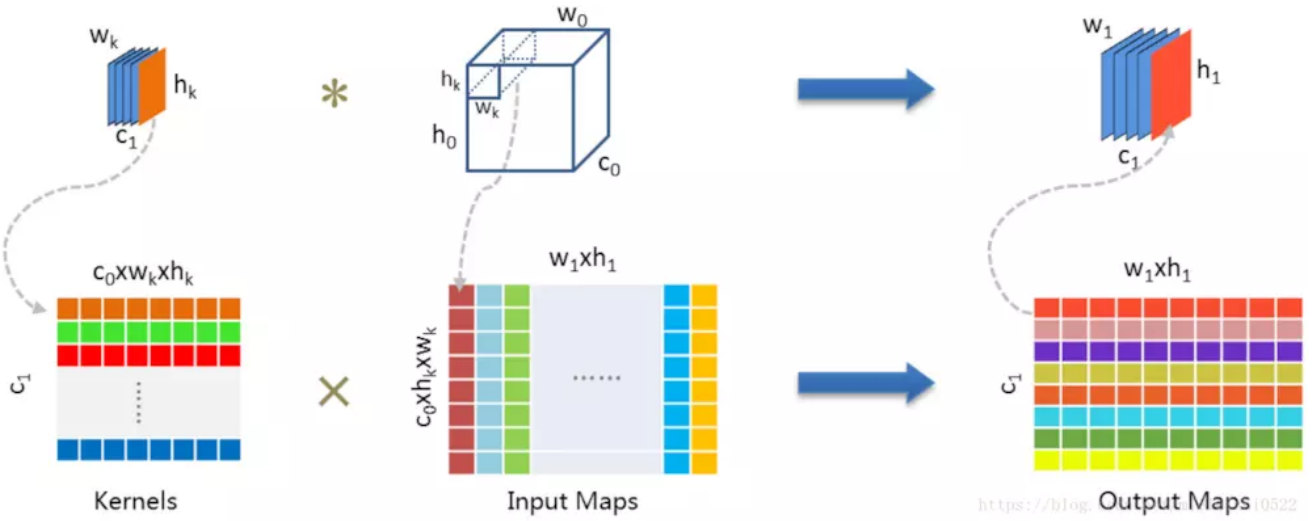
对每个patch, *右乘*以 filter matrix.得到[batch, out_height, out_width,out_channels]大小的输出。

其中，out_height和out_width是根据输入尺寸等参数计算好的，不会计算的自行补充学习吧。

详细来说，使用默认的NHWC形式，

个人理解相当于在input每个卷积核的位置上(包含了同一位值对应的不同channel)提取的patches展开之后，与展开的filter kernel相乘。

看图很清楚，示意图参考[3].



2、torch.nn.Conv2d()

pytorch源码里面说的就没有tf里说的清楚了，点不动

其中，\star是2D的cross-correlation_（互相关运算符）， N是batch_size。

*互相关函数*是许多机器学习的库中都会有实现的一个函数，和卷积运算几乎一样但是没有进行核的翻转。

参考资料：

[1]、https://github.com/tensorflow/tensorflow

[2]、https://buptldy.github.io/2016/10/01/2016-10-01-im2col/

[3]、https://blog.csdn.net/mieleizhi0522/article/details/80412804