Multiple input channels

当输入包含多个通道时，需要构造一个与输入数据具有相同输入通道数的卷积核，以便与输入数据进行互相关运算。

假设输入的通道数为c_i，那么卷积核的输入通道数也需要为c_i。如果卷积核的窗口形状是$k_h \times k_h$，那么当时，我们可以把卷积核看作形状为的二维张量。

Traditional Code

```
import torch
from d2l import torch as d2l
from torch import nn

def corr2d_multi_in(X,K):
    return sum(d2l.corr2d(x,k) for x,k in zip(X,K))

X = torch.tensor([[[0.0,1.0,2.0],[3.0,4.0,5.0],[6.0,7.0,8.0]],
                  [[1.0,2.0,3.0],[4.0,5.0,6.0],[7.0,8.0,9.0]]])
K = torch.tensor([[[0.0,1.0],[2.0,3.0]],[[1.0,2.0],[3.0,4.0]]])
print(corr2d_multi_in(X,K))

def corr2d_multi_in_out(X,K):  # X为3通道矩阵，K为4通道矩阵，最外面维为输出通道
    return torch.stack([corr2d_multi_in(X,k) for k in K],0)
# K中每个k是一个3D的Tensor。0表示stack堆叠函数里面在0这个维度堆叠。

print(K.shape)
print((K+1).shape)
print((K+2).shape)
print(K)
print(K+1)
K = torch.stack((K, K+1, K+2),0) # K与K+1之间的区别为K的每个元素加1
print(K.shape)
print(corr2d_multi_in_out(X,K))
```

Pytorch Code

```
def comp_conv2d(conv2d, X):
    X = X.reshape((1,1)+X.shape) # 加入通道数
    Y = conv2d(X)
    return Y.reshape(Y.shape[2:]) # 去掉前两个维度

X = torch.rand(size=(8,8))
conv2d = nn.Conv2d(1,1,kernel_size=3,padding=1,stride=2)
# Pytorch里面卷积函数的第一个参数为输出通道，第二个参数为输入通道
print(comp_conv2d(conv2d,X).shape)

conv2d = nn.Conv2d(1,1,kernel_size=(3,5),padding=(0,1),stride=(3,4))
print(comp_conv2d(conv2d,X).shape)
```