卷积神经网络

# 卷积介绍

# 图像卷积

# 填充和步幅

在前面的例子图6.2.1中，输入的高度和宽度都为，卷积核的高度和宽度都为，生成的输出表征的维数为。正如我们在 第2节中所概括的那样，假设输入形状为，卷积核形状为那么输出形状将是。因此，卷积的输出形状取决于输入形状和卷积核的形状。

还有什么因素会影响输出的大小呢？本节我们将介绍**填充**（padding）和**步幅**（stride）。假设以下情景：有时，在应用了连续的卷积之后，我们最终得到的输出远小于输入大小。这是由于卷积核的宽度和高度通常大于所导致的。比如，一个像素的图像，经过层的卷积后，将减少到像素。如此一来，原始图像的边界丢失了许多有用信息。而**填充**是解决此问题最有效的方法；有时，我们可能希望大幅降低图像的宽度和高度。例如，如果我们发现原始的输入分辨率十分冗余。**步幅**则可以在这类情况下提供帮助。填充

## 填充

如上所述，在应用多层卷积时，我们常常丢失边缘像素。由于我们通常使用小卷积核，因此对于任何单个卷积，我们可能只会丢失几个像素。但随着我们应用许多连续卷积层，累积丢失的像素数就多了。解决这个问题的简单方法即为**填充**（padding）：在输入图像的边界填充元素（通常填充元素是0）。例如，在图 3.1.1中，我们将输入填充到，那么它的输出就增加为。阴影部分是第一个输出元素以及用于输出计算的输入和核张量元素：。

Text

Description automatically generated with medium confidence

图 3.1.1填充为1核为2的二维互相关

通常，如果我们添加行填充（大约一半在顶部，一半在底部）和pw列填充（左侧大约一半，右侧一半），则输出形状将为 。

这意味着输出的高度和宽度将分别增加和。

在许多情况下，我们需要设置和，使输入和输出具有相同的高度和宽度。 这样可以在构建网络时更容易地预测每个图层的输出形状。假设是奇数，我们将在高度的两侧填充行。 如果是偶数，则一种可能性是在输入顶部填充行，在底部填充行。同理，我们填充宽度的两侧。

卷积神经网络中卷积核的高度和宽度通常为奇数，例如、、或。 选择奇数的好处是，保持空间维度的同时，我们可以在顶部和底部填充相同数量的行，在左侧和右侧填充相同数量的列。

此外，使用奇数的核大小和填充大小也提供了书写上的便利。对于任何二维张量，当满足： 1. 卷积核的大小是奇数； 2. 所有边的填充行数和列数相同； 3. 输出与输入具有相同高度和宽度 则可以得出：输出Y[i, j]是通过以输入X[i, j]为中心，与卷积核进行互相关计算得到的。

比如，在图 3.1.2的例子中，我们创建一个高度和宽度为的二维卷积层，并在所有侧边填充个像素。给定高度和宽度为的输入，则输出的高度和宽度也是。



图 3.1.2 Pytorch实现padding

## 步幅

在计算互相关时，卷积窗口从输入张量的左上角开始，向下、向右滑动。 在前面的例子中，我们默认每次滑动一个元素。 但是，有时候为了高效计算或是缩减采样次数，卷积窗口可以跳过中间位置，每次滑动多个元素。

我们将每次滑动元素的数量称为**步幅**（stride）。到目前为止，我们只使用过高度或宽度为的步幅，那么如何使用较大的步幅呢？ [图6.3.2](https://zh.d2l.ai/chapter_convolutional-neural-networks/padding-and-strides.html#img-conv-stride)是垂直步幅为，水平步幅为的二维互相关运算。 着色部分是输出元素以及用于输出计算的输入和内核张量元素：、。

可以看到，为了计算输出中第一列的第二个元素和第一行的第二个元素，卷积窗口分别向下滑动三行和向右滑动两列。但是，当卷积窗口继续向右滑动两列时，没有输出，因为输入元素无法填充窗口（除非我们添加另一列填充）。

通常，当垂直步幅为、水平步幅为时，输出形状为

如果我们设置了和，则输出形状将简化为。 更进一步，如果输入的高度和宽度可以被垂直和水平步幅整除，则输出形状将为。

# 多输入多输出通道

到目前为止，我们仅展示了单个输入和单个输出通道的简化例子。 这使得我们可以将输入、卷积核和输出看作二维张量。当我们添加通道时，我们的输入和隐藏的表示都变成了三维张量。例如，每个RGB输入图像具有的形状。我们将这个大小为的轴称为**通道**（channel）维度。本节将更深入地研究具有多输入和多输出通道的卷积核。

## 多输入通道

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，那么卷积核的输入通道数也需要为。如果卷积核的窗口形状是，那么当时，我们可以把卷积核看作形状为的二维张量。

然而，当时，我们卷积核的每个输入通道将包含形状为的张量。将这些张量连结在一起可以得到形状为的卷积核。由于输入和卷积核都有个通道，我们可以对每个通道输入的二维张量和卷积核的二维张量进行互相关运算，再对通道求和（将的结果相加）得到二维张量。这是多通道输入和多输入通道卷积核之间进行二维互相关运算的结果。

在图 4.1.1中，我们演示了一个具有两个输入通道的二维互相关运算的示例。第一个输出元素以及用于计算这个输出的输入和核张量元素：。

|  |  |
| --- | --- |
| 图 4.1.1·多通道输入-单通道输出 | 图 4.1.2多通道输入-单通道输出 |

## 多输出通道

**到目前为止，不论有多少输入通道，我们还只有一个输出通道**。然而，正如我们在 [6.1.4.1节](https://zh.d2l.ai/chapter_convolutional-neural-networks/why-conv.html#subsec-why-conv-channels)中所讨论的，每一层有多个输出通道是至关重要的。在最流行的神经网络架构中，随着神经网络层数的加深，我们常会增加输出通道的维数，通过减少空间分辨率以获得更大的通道深度。直观地说，我们可以将每个通道看作对不同特征的响应。而现实可能更为复杂一些，因为每个通道不是独立学习的，而是为了共同使用而优化的。因此，多输出通道并不仅是学习多个单通道的检测器。

用和分别表示输入和输出通道的数目，并让和为卷积核的高度和宽度。为了获得多个通道的输出，我们可以为每个输出通道创建一个形状为的卷积核张量，这样卷积核的形状是。在互相关运算中，每个输出通道先获取所有输入通道，再以对应该输出通道的卷积核计算出结果。

如图 4.1.2所示，我们实现一个计算多个通道的输出的互相关函数。

## 卷积层（**一个特例**）

卷积，即，看起来似乎没有多大意义。 毕竟，卷积的本质是有效提取相邻像素间的相关特征，而卷积显然没有此作用。 尽管如此，仍然十分流行，经常包含在复杂深层网络的设计中。下面，让我们详细地解读一下它的实际作用。

因为使用了最小窗口，卷积失去了卷积层的特有能力——在高度和宽度维度上，识别相邻元素间相互作用的能力。 其实卷积的唯一计算发生在通道上。

[图6.4.2](https://zh.d2l.ai/chapter_convolutional-neural-networks/channels.html#fig-conv-1x1)展示了使用卷积核与个输入通道和个输出通道的互相关计算。 这里输入和输出具有相同的高度和宽度，输出中的每个元素都是从输入图像中同一位置的元素的线性组合。 我们可以将卷积层看作在每个像素位置应用的全连接层，以个输入值转换为个输出值。 因为这仍然是一个卷积层，所以跨像素的权重是一致的。 同时，卷积层需要的权重维度为，再额外加上一个偏置。

# 汇聚层

# 卷积神经网络（LeNet）