1572/5000

CONV层将计算连接到输入中局部区域的神经元的输出，每个计算它们的权重与它们在输入体积中连接的小区域之间的点积。如果我们决定使用12个滤镜，这可能导致音量如[32x32x12]  
RELU层将应用元素激活函数，如零（0，x）阈值处理。这会使卷的大小不变（[32x32x12]）。  
POOL层将沿着空间维度（宽度，高度）执行下采样操作，从而产生诸如[16x16x12]的音量。  
FC（即完全连接）层将计算类别分数，从而得到大小为[1×1×10]的数量，其中每个数字对应于类别分数，例如10个类别的CIFAR-10。和普通的神经网络一样，顾名思义，这个层中的每个神经元都将连接到前一卷中的所有数字。  
通过这种方式，ConvNets将原始图像逐层从原始像素值转换为最终的类别分数。请注意，某些图层包含参数，其他图层则不包含。具体而言，CONV / FC层执行转换，这些转换不仅是输入体积中的激活，而且也是参数（神经元的权重和偏差）的函数。另一方面，RELU / POOL层将实现一个固定的功能。 CONV / FC图层中的参数将使用梯度下降进行训练，以便ConvNet计算的类别分数与每个图像的训练集中的标签一致。

562/5000

综上所述：  
  
     ConvNet体系结构最简单的情况就是将图像体积转换为输出体积的图层列表（例如，保持班级分数）  
     有几种不同类型的层（例如CONV / FC / RELU / POOL是目前最流行的）  
     每层接受一个输入3D音量并通过可微分功能将其转换为输出3D音量  
     每一层可能有也可能没有参数（例如，CONV / FC do，RELU / POOL不）  
     每一层可能有也可能没有额外的超参数（例如，CONV / FC / POOL，RELU没有）

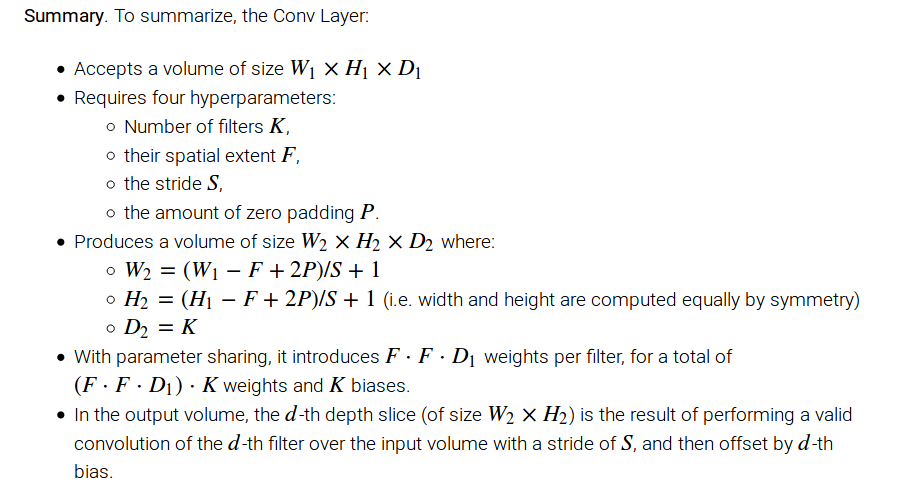
卷积层  
  
Conv层是卷积网络的核心组成部分，它完成大部分计算繁重的工作。  
  
概述和直觉，没有大脑的东西。让我们首先讨论CONV层在没有大脑/神经元类比的情况下计算什么。 CONV层的参数由一组可学习的过滤器组成。每个滤波器在空间上都很小（沿着宽度和高度），但是会延伸到输入音量的整个深度。例如，ConvNet的第一层上的典型滤波器可能具有5x5x3的尺寸（即，5像素的宽度和高度，以及3，因为图像具有深度3，颜色通道）。在正向传递期间，我们沿着输入体积的宽度和高度滑动（更精确地说，卷积）每个滤波器，并计算滤波器输入和任意位置输入之间的点积。当我们在输入体积的宽度和高度上滑动滤波器时，我们将生成一个二维激活图，该图可以在每个空间位置给出滤波器的响应。直观地说，网络将学习过滤器，当他们看到某种类型的视觉特征时激活，例如第一层上某种方向的边缘或某种颜色的斑点，或者最终在网络的更高层上最终形成整个蜂窝或轮状图案。现在，我们将在每个CONV层中有一整套滤波器（例如12个滤波器），并且它们中的每一个都将生成单独的二维激活图。我们将沿着深度维度堆叠这些激活图并产生输出量。

大脑观点。 如果你是大脑/神经元类比的粉丝，3D输出体积中的每个条目也可以被解释为神经元的输出，该输出只查看输入中的一个小区域，并与所有神经元共享参数， 在空间上是正确的（因为这些数字全都来自应用相同的过滤器）。 我们现在讨论神经元连接的细节，它们在空间中的排列以及它们的参数共享方案。

本地连接。当处理像图像这样的高维输入时，正如我们上面看到的，将神经元连接到前一个体积中的所有神经元是不切实际的。相反，我们将每个神经元连接到输入音量的局部区域。这种连接的空间范围是一个超参数，称为神经元的接受场（相当于过滤器的大小）。沿着深度轴的连通程度总是等于输入音量的深度。在我们如何处理空间尺寸（宽度和高度）和深度尺寸时，再次强调这种不对称是非常重要的：连接在空间上（沿着宽度和高度）是局部的，但始终沿着输入体积的整个深度。  
示例1.例如，假设输入音量大小为[32x32x3]（例如RGB CIFAR-10图像）。如果接受域（或过滤器大小）为5x5，那么Conv层中的每个神经元将对输入体积中的[5x5x3]区域具有权重，总共5 \* 5 \* 3 = 75个权重（以及+1偏置参数）。请注意，沿着深度轴的连通程度必须为3，因为这是输入量的深度。  
  
左：红色的示例输入体积（例如32x32x3 CIFAR-10图像）以及第一卷积层中的示例体积的神经元。卷积层中的每个神经元仅在空间上连接到输入体积中的局部区域，但是连接到全深度（即所有颜色通道）。请注意，沿深度有多个神经元（本例中为5个），所有神经元都在输入中查看相同的区域 - 请参阅下面文本中的深度列的讨论。正确：来自神经网络章节的神经元保持不变：它们仍然计算它们的权重与输入之间的点积，然后是非线性，但它们的连接现在被限制在局部空间上。

2653/5000

事实证明，通过作出一个合理的假设，我们可以大大减少参数的数量：如果一个特征对于在某个空间位置（x，y）进行计算是有用的，那么在另一个位置（x2 ，y2）上。换句话说，将单个2维深度切片表示为深度切片（例如，大小为[55x55x96]的体积具有96个深度切片，每个切片的大小为[55x55]），我们将限制每个深度切片中的神经元使用相同的权重和偏差。使用这个参数共享方案，我们例子中的第一个Conv层现在只有96个独特权重集合（每个深度切片一个权重集合），共96 \* 11 \* 11 \* 3 = 34,848个唯一权重或34,944个参数（ +96偏见）。或者，每个深度切片中的所有55 \* 55神经元现在将使用相同的参数。在反向传播实践中，体积中的每个神经元都会计算其权重的梯度，但是这些梯度将叠加在每个深度切片上，并且仅更新每个切片的一组权重。  
注意，如果在一个单一的深度切片所有神经元都使用相同的权重向量，则CONV层的直传可以在每个深度切片被计算为神经元的与输入量（因此命名为权重的卷积：卷积层）。这就是为什么通常将权重集合称为过滤器（或内核），与输入进行卷积的原因。  
Krizhevsky等人学习的示例过滤器此处显示的96个过滤器中的每个过滤器的大小为[11x11x3]，并且每个过滤器在一个深度切片中由55 \* 55个神经元共享。请注意，参数共享假设是比较合理的：如果在图像中的某个位置检测到水平边缘很重要，则由于图像的平移不变结构，它应该在其他位置直观地发挥作用。因此不需要重新学习来检测Conv层输出体积中每个55 \* 55个不同位置的水平边缘。  
请注意，有时参数共享假设可能没有意义。当ConvNet的输入图像具有特定的中心结构时，尤其如此，例如，我们应该期望在图像的一侧应该学习完全不同的特征。一个实际的例子是当输入是在图像中居中的面。你可能会期望在不同的空间位置可以（而且应该）学习不同的眼睛特定或头发特定的特征。在这种情况下，通常放宽参数共享方案，而只需将图层称为本地连接层



作为矩阵乘法实现。请注意，卷积运算本质上是在滤波器和输入的局部区域之间执行点积。 CONV层的常见实现模式是利用这一事实并将卷积层的正向通过制定为如下的一个大矩阵乘法：  
  
  
    输入图像中的局部区域在通常称为im2col的操作中被拉伸成列。例如，如果输入是[227x227x3]，并且要在步幅4与11x11x3滤波器卷积，则我们将在输入中采用[11x11x3]个像素块，并将每个块拉伸到大小为11 \* 11 \* 3 = 363.在步幅4的输入中迭代该过程沿着宽度和高度给出（227-11）/ 4 + 1 = 55个位置，导致尺寸为[363×3025]的im2col的输出矩阵X\_col，其中每一列都是一个伸出的感受野，总共有55 \* 55 = 3025个。请注意，由于接受域重叠，输入体积中的每个数字可能会复制到多个不同的列中。  
    CONV层的权重类似地伸展成行。例如，如果有大小为[11x11x3]的96个过滤器，则会给出大小为[96 x 363]的矩阵W\_row。  
    卷积的结果现在相当于执行一个大的矩阵乘法np.dot（W\_row，X\_col），它可以计算每个滤波器与每个接收场位置之间的点积。在我们的例子中，这个操作的输出是[96 x 3025]，给出每个位置上每个过滤器的点积的输出。  
    结果必须重新调整为适当的输出尺寸[55x55x96]。  
这种方法的缺点是可以使用大量内存，因为输入卷中的某些值在X\_col中被复制多次。但是，好处是有很多非常有效的Matrix Multiplication实现可以利用（例如，在常用的BLAS API中）。此外，我们可以重复使用相同的im2col想法来执行池操作，我们将在下面讨论这个操作。