卷积神经网络

卷积层

1. 引言

- **分类猫和狗的图片**

  使用一个还不错的相机采集图片（12M 像素）

  RGB 图片有 36M 元素

  使用 100 大小的单隐藏层 MLP，模型有 3.6B 元素远多于世界上所有猫和狗总数（900M 狗，600M 猫）

- **两个原则:**

  a) 平移不变性 (translation invariance): 不管检测对象出现在图像中的哪个位置，神经网络的前面几层应该对相同的图像区域具有相似的反应，即为 ˝ 平移不变性”。

  b) 局部性 (locality): 神经网络的前面几层应该只探索输入图像中的局部区域，而不过度在意图像中相隔较远区域的关系，这就是“局部性”原则。最终，可以聚合这些局部特征，以在整个图像级别进行预测。

2. 重新考察全连接层

- 将输入和输出变成矩阵（为了考虑图片的位置信息）

  输入矩阵宽度和高度分别为k和l；输出矩阵宽度和高度分别为i和j

- 将权重变形为 4-D 张量

$$h_{i,j} = \sum_{k,l} w_{i,j,k,l} x_{k,l}$$

- 对$w_{i,j,k,l}$进行重新索引: $v_{i,j,a,b} = w_{i,j,i+a,i+b}$

  v和w不是同样的张量，是一一对应的关系，为了方便引出后续卷积

$$h_{i,j} = \sum_{a,b} v_{i,j,a,b} x_{i+a,j+b}$$

2.1 原则#1 - 平移不变形

- x 的平移导致 h 的平移 $h_{i,j} = \sum_{a,b} v_{i,j,a,b} x_{i+a,j+b}$
- v 不应该依赖于（i，j）
- 解决方案: $v_{i,j,a,b} = v_{a,b}$

$$h_{i,j} = \sum_{a,b} v_{a,b} x_{i+a,j+b}$$

2.2 原则#2 - 局部性

$$h_{i,j} = \sum_{a,b} v_{a,b} x_{i+a,j+b}$$

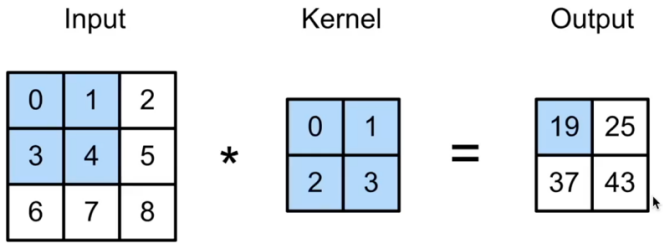
- 当评估 $h_{i,j}$ 时，我们不应该用远离 $x_{i,j}$ 的参数
- 解决方案：当 $|a|, |b| > \Delta$ 时，使得 $v_{a,b} = 0$

$$h_{i,j} = \sum_{a=-\Delta}^{\Delta} \sum_{b=-\Delta}^{\Delta} v_{a,b} x_{i+a,j+b}$$

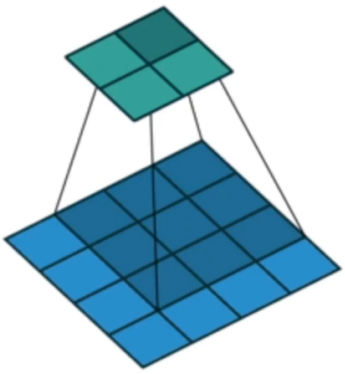
2.3 总结

- **对全连接层使用平移不变性和局部性得到卷积层**

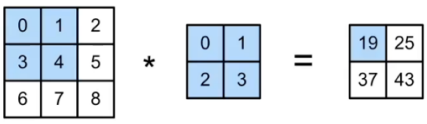
3. 二维交叉相关（'卷积'）



$$0 \times 0 + 1 \times 1 + 3 \times 2 + 4 \times 3 = 19,$$
$$1 \times 0 + 2 \times 1 + 4 \times 2 + 5 \times 3 = 25,$$
$$3 \times 0 + 4 \times 1 + 6 \times 2 + 7 \times 3 = 37,$$
$$4 \times 0 + 5 \times 1 + 7 \times 2 + 8 \times 3 = 43.$$



4. 二维卷积层



- 输入 $\mathbf{X}: n_h \times n_w$
- 核 $\mathbf{W}: k_h \times k_w$
- 偏差 $b \in \mathbb{R}$
- 输出 $\mathbf{Y}: (n_h - k_h + 1) \times (n_w - k_w + 1)$

$$\mathbf{Y} = \mathbf{X} \star \mathbf{W} + b$$

- **W** 和 b 是可学习的参数

5. 交叉相关 vs 卷积

- 二维交叉相关

$$y_{i,j} = \sum_{a=1}^{h}\sum_{b=1}^{w} w_{a,b}x_{i+a,j+b}$$

- 二维卷积

$$y_{i,j} = \sum_{a=1}^{h}\sum_{b=1}^{w} w_{-a,-b}x_{i+a,j+b}$$

- **由于对称性，**在实际使用中没有区别

6. 一维和三维交叉相关

6.1 一维

$$y_i = \sum_{a=1}^{h} w_a x_{i+a}$$

- **文本，语言，时序序列**

6.2 三维

$$y_{i,j,k} = \sum_{a=1}^{h}\sum_{b=1}^{w}\sum_{c=1}^{d} w_{a,b,c}x_{i+a,j+b,k+c}$$

- **视频，医学图像，气象地图**

7. 总结

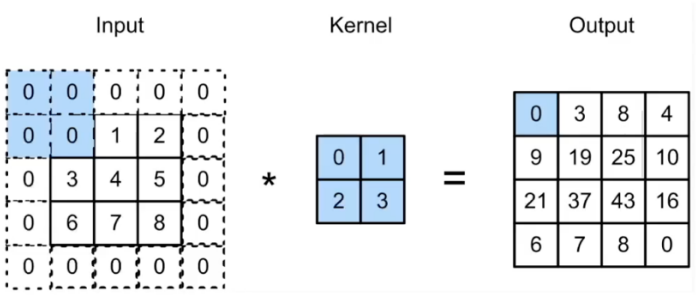
- 卷积层将输入和核矩阵进行交叉相关，加上偏移后得到输出
- 核矩阵和偏移是可学习的参数
- 核矩阵的大小是超参数
- **卷积的本质是有效提取相邻像素间的相关特征**

填充和步幅

1. 填充

- 给定（32 x 32）输入图像
- 应用 5 x 5 大小的 卷积核
  - 第1层得到输出大小 28 x 28
  - 第7层得到输出大小4 x 4
- 更大的卷积核可以更快地减小输出大小
  - 形状从 $n_h \times n_w$ 减少到
    $(n_h - k_h + 1) \times (n_w - k_w + 1)$

- 在应用多层卷积时，常常丢失边缘像素。由于通常使用小卷积核，因此对于任何单个卷积，可能只会丢失几个像素。但随着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）。



$$0 \times 0 + 0 \times 1 + 0 \times 2 + 0 \times 3 = 0$$

- 填充 p_h 行和 p_w 列，输出形状为

$$(n_h - k_h + p_h + 1) \times (n_w - k_w + p_w + 1)$$

- 通常取 $p_h = k_h - 1, \quad p_w = k_w - 1$
  - 当 k_h 为奇数：在上下两侧填充 $p_h/2$
  - 当 k_h 为偶数：在上侧填充 $\lceil p_h/2 \rceil$，在下侧填充 $\lfloor p_h/2 \rfloor$

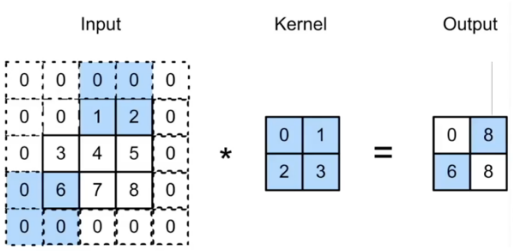
- 卷积神经网络中卷积核的高度和宽度通常为奇数，例如 1、3、5 或 7。 选择奇数的好处是，保持空间维度的同时，可以在顶部和底部填充相同数量的行，在左侧和右侧填充相同数量的列。

2. 步幅

- 填充减小的输出大小与层数线性相关。

给定输入大小 224×224，在使用 5×5 卷积核的情况下，需要 55 层将输出降低到 4×4。需要大量计算才能得到较小输出。

- 步幅是指行/列的滑动步长
  - 例：高度3 宽度2 的步幅



$$0\times0+0\times1+1\times2+2\times3=8$$
$$0\times0+6\times1+0\times2+0\times3=6$$

- 给定高度 s_h 和宽度 s_w 的步幅，输出形状是

$$\lfloor(n_h-k_h+p_h+s_h)/s_h\rfloor\times\lfloor(n_w-k_w+p_w+s_w)/s_w\rfloor$$

- 如果 $p_h=k_h-1,\quad p_w=k_w-1$

$$\lfloor(n_h+s_h-1)/s_h\rfloor\times\lfloor(n_w+s_w-1)/s_w\rfloor$$

- 如果输入高度和宽度可以被步幅整除

$$(n_h/s_h)\times(n_w/s_w)$$

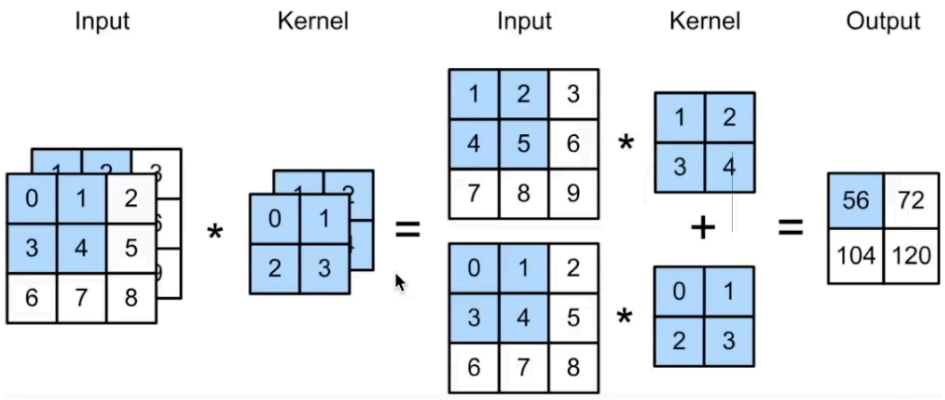
3. 总结

- 填充和步幅是卷积层的**超参数**
- 填充在输入周围添加额外的行/列，来控制输出形状的减少量
- 步幅是每次滑动核窗口时的行/列的步长，可以成倍的减少输出形状

多输入多输出通道

1. 多个输入通道

- 彩色图像可能有 RGB 三个通道
  转换为灰度可能会丢失信息
- 每个通道都有一个卷积核，结果是所有通道卷积结果的和



$$(1\times 1+2\times 2+4\times 3+5\times 4)$$
$$+(0\times 0+1\times 1+3\times 2+4\times 3)=56$$

- 输入 $\mathbf{X}: c_i \times n_h \times n_w$
- 核 $\mathbf{W}: c_i \times k_h \times k_w$
- 输出 $\mathbf{Y}: m_h \times m_w$

$$\mathbf{Y}=\sum_{i=0}^{c_i} \mathbf{X}_{i,:,:} \star \mathbf{W}_{i,:,:}$$

2. 多个输出通道

- 无论有多少输入通道，可以有多个三维卷积核，每个核生成一个输出通道。

- 输入 $\mathbf{X}: c_i \times n_h \times n_w$
- 核 $\mathbf{W}: c_o \times c_i \times k_h \times k_w$
- 输出 $\mathbf{Y}: c_o \times m_h \times m_w$

$$\mathbf{Y}_{i,:,:}=\mathbf{X}\star \mathbf{W}_{i,:,:,:} \quad \text{for } i=1,\ldots,c_o$$

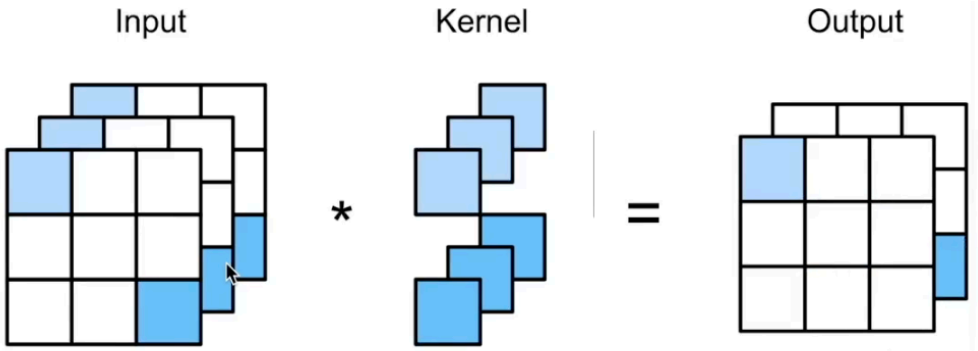
3. 多个输入和输出通道

- 每个输出通道可以识别特定模式

- 输入通道核识别并组合输入中的模式

4. 1×1卷积层

- $k_h = k_w = 1$是一个受欢迎的选择，它不是别空间模式，只是融合通道。



相当于输入形状为$n_h n_w \times c_i$，权重为$c_0 \times c_i$的全连接层

5. 二维卷积层

- 输入 $\mathbf{X}: c_i \times n_h \times n_w$
- 核 $\mathbf{W}: c_o \times c_i \times k_h \times k_w$
- 偏差 $\mathbf{B}: c_o \times c_i$
- 输出 $\mathbf{Y}: c_o \times m_h \times m_w$

$$\mathbf{Y} = \mathbf{X} \star \mathbf{W} + \mathbf{B}$$

- 计算复杂度 （浮点计算数 FLOP）$O(c_i c_o k_h k_w m_h m_w)$

$$c_i = c_o = 100$$
$$k_h = h_w = 5$$
$$m_h = m_w = 64$$

⇒ 1GFLOP

- 10 层，1M 样本，10 PFlops
(CPU: 0.15 TF = 18h, GPU: 12 TF = 14min)

6. 总结

- 输出通道数是卷积层的超参数
- 每个输入通道有独立的二维卷积核，所有通道结果相加得到一个输出通道结果
- 每个输出通道有独立的三维卷积核
- 多输入多输出通道可以用来扩展卷积层的模型。
- 当以每像素为基础应用时，1×1卷积层相当于全连接层。
- 1×1卷积层通常用于调整网络层的通道数量和控制模型复杂性。

池化层

1. 引言

- **卷积对位置敏感**

  例如，检测垂直边缘

```
X = [[1. 1. 0. 0. 0.
      [1. 1. 0. 0. 0.
      [1. 1. 0. 0. 0.
      [1. 1. 0. 0. 0.

Y = [[ 0.  1.  0.  0.
      [ 0.  1.  0.  0.
      [ 0.  1.  0.  0.
      [ 0.  1.  0.  0.
```

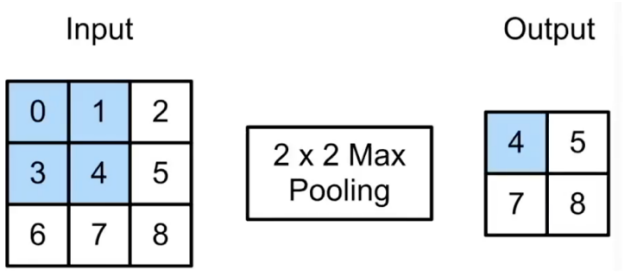
  1 像素位移，就会导致 0 输出

- 需要一定程度的平移不变性

  照明，物体位置，比例，外观等等因图像而异

2. 二维最大池化

- 返回滑动窗口中的最大值



$$\max(0,1,3,4) = 4$$

- 对于边缘检测问题: 可容 1 像素移位

垂直边缘检测

```
[[1. 1. 0. 0. 0.
 [1. 1. 0. 0. 0.
 [1. 1. 0. 0. 0.
 [1. 1. 0. 0. 0.
```

卷积输出

```
[[ 0.  1.  0.  0.
 [ 0.  1.  0.  0.
 [ 0.  1.  0.  0.
 [ 0.  1.  0.  0.
```

2 x 2 最大池化

```
[[ 1.  1.  1.  0.
 [ 1.  1.  1.  0.
 [ 1.  1.  1.  0.
 [ 1.  1.  1.  0.
```

3. 池化层的填充，步幅和多个通道

- 池化层和卷积层类似，都具有填充和步幅（超参数）
- 没有可学习的参数
- 在每个输入通道应用池化层以获得相应的输出通道
- 输出通道数 = 输入通道数

4. 平均池化层

- 最大池化层: 每个窗口中最强的模式信号
- 平均池化层: 将最大池化层中的“最大”操作替换为“平均”

5. 总结

- 池化层返回窗口中最大或平均值
- 缓解卷积层会位置的敏感性
- 同样有窗口大小、填充、和步幅作为超参数

 # LeNet

- LeNet 是早期成功的神经网络
- 先使用卷积层来学习图片空间信息
- 然后使用全连接层来转换到类别空间