Deep Learning I

6. 卷积神经网络

WU Xiaokun 吴晓堃

xkun.wu [at] gmail

2021/03/28

从全连接到卷积

图片有多大?

假设是猫狗分类任务，使用手机拍摄图片：12M像素

- RGB图片：36M元素

图片有多大?

假设是猫狗分类任务，使用手机拍摄图片：12M像素

- RGB图片：36M元素
- 即使用大小只有100的单隐藏层MLP：3.6B元素
  - 32位浮点运算：**14.4GB存储**
    - 3080 Ti：12GB显存

图片有多大?

假设是猫狗分类任务，使用手机拍摄图片：12M像素

- RGB图片：36M元素
- 即使用大小只有100的单隐藏层MLP：3.6B元素
  - 32位浮点运算：**14.4GB存储**
    - 3080 Ti：12GB显存
- 远多于世界上猫（600M）和狗（900M）的总和
  - 比一一记住这些猫、狗还要复杂

图片有多大?

假设是猫狗分类任务，使用手机拍摄图片：12M像素

- RGB图片：36M元素
- 即使用大小只有100的单隐藏层MLP：3.6B元素
  - 32位浮点运算：**14.4GB存储**
    - 3080 Ti：12GB显存
- 远多于世界上猫（600M）和狗（900M）的总和
  - 比一一记住这些猫、狗还要复杂

MLP是**全连接**网络

- **没有任何预先假设**：与特征交互相关的先验结构
  - 什么时候需要假设？简化计算，如物理定律
  - 低效，但不丢失信息：适用于表格数据

图片有多大?

假设是猫狗分类任务，使用手机拍摄图片：12M像素

- RGB图片：36M元素
- 即使用大小只有100的单隐藏层MLP：3.6B元素
  - 32位浮点运算：**14.4GB存储**
    - 3080 Ti：12GB显存
- 远多于世界上猫（600M）和狗（900M）的总和
  - 比一一记住这些猫、狗还要复杂

MLP是**全连接**网络

- **没有任何预先假设**：与特征交互相关的先验结构
  - 什么时候需要假设？简化计算，如物理定律
  - 低效，但不丢失信息：适用于表格数据

人是如何从图片寻物的?

人是如何从图片寻物的?

- **位置不重要**；特征才是决定因素
- **聚焦式注意力**：自动忽略背景

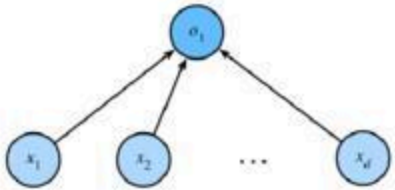
两个原则

- **平移不变性**：位置无关；相似的区域激活相似的数值
  - 例如：下图中俩个窗口内都有眼镜
- **局部性**：操作限制在小的局部窗口内

全连接的计算

类比（全连接）线性模型：输入、输出是**矩阵**

- 横（宽）、纵（高）两个维度



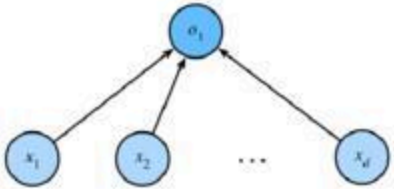
$$o_i = w^k x_k$$

$$\mathbf{H}_{i,j} = \mathbf{W}_{i,j}^{k,l} \mathbf{X}_{k,l}$$

全连接的计算

类比（全连接）线性模型：输入、输出是**矩阵**

- 横（宽）、纵（高）两个维度



$$o_i = w^k x_k$$

$$\mathbf{H}_{i,j} = \mathbf{W}_{i,j}^{k,l} \mathbf{X}_{k,l}$$

重新索引以适配局部窗口：$k = i + a, l = j + b$

- **以**(i, j)**为中心**的加权和：$\mathbf{H}_{i,j} = \mathbf{W}_{i,j}^{k,l} \mathbf{X}_{k,l} = \mathbf{W}_{i,j}^{a,b} \mathbf{X}_{i+a,j+b}$
- (a, b)：即是偏移量，也是**窗口索引**

原则I：平移不变性

检测对象在输入空间平移，**输出不变**

- 参数$\mathbf{W}$不应该依赖于位置(i, j)
  - $\mathbf{H}_{i,j} = \mathbf{W}_{i,j}^{a,b}\mathbf{X}_{i+a,j+b}$

原则I：平移不变性

检测对象在输入空间平移，**输出不变**

- 参数$\mathbf{W}$不应该依赖于位置(i,j)
  - $\mathbf{H}_{i,j} = \mathbf{W}_{i,j}^{a,b}\mathbf{X}_{i+a,j+b}$

只可能是：$\mathbf{W}_{i,j}^{a,b} = \mathbf{V}^{a,b}$

- 每个窗口索引(a,b)上都是**常数**
- 因此：$\mathbf{H}_{i,j} = \mathbf{W}_{i,j}^{a,b}\mathbf{X}_{i+a,j+b} = \mathbf{V}^{a,b}\mathbf{X}_{i+a,j+b}$

原则I：平移不变性

检测对象在输入空间平移，**输出不变**

- 参数$\mathbf{W}$不应该依赖于位置(i,j)
  - $\mathbf{H}_{i,j} = \mathbf{W}_{i,j}^{a,b}\mathbf{X}_{i+a,j+b}$

只可能是：$\mathbf{W}_{i,j}^{a,b} = \mathbf{V}^{a,b}$

- 每个窗口索引(a,b)上都是**常数**
- 因此：$\mathbf{H}_{i,j} = \mathbf{W}_{i,j}^{a,b}\mathbf{X}_{i+a,j+b} = \mathbf{V}^{a,b}\mathbf{X}_{i+a,j+b}$

这就是~~**卷积 convolution**~~**交叉相关**操作

原则I：平移不变性

检测对象在输入空间平移，**输出不变**

- 参数$\mathbf{W}$不应该依赖于位置(i,j)
  - $\mathbf{H}_{i,j} = \mathbf{W}_{i,j}^{a,b}\mathbf{X}_{i+a,j+b}$

只可能是：$\mathbf{W}_{i,j}^{a,b} = \mathbf{V}^{a,b}$

- 每个窗口索引(a,b)上都是**常数**
- 因此：$\mathbf{H}_{i,j} = \mathbf{W}_{i,j}^{a,b}\mathbf{X}_{i+a,j+b} = \mathbf{V}^{a,b}\mathbf{X}_{i+a,j+b}$

这就是~~**卷积 convolution**~~**交叉相关**操作

注意：从$\mathbf{W}_{i,j}^{a,b}$到$\mathbf{V}^{a,b}$参数量极大减少

原则II：局部性

检测区域仅限于输入空间中的**局部窗口**

- 远离位置(i, j)的参数$\mathbf{V}$不参与计算

解决方案：$|a|, |b| > \Delta \Rightarrow \mathbf{V}^{a,b} = 0$

$$\mathbf{H}_{i,j} = \sum_{a,b=-\Delta}^{\Delta} \mathbf{V}^{a,b} \mathbf{X}_{i+a,j+b}$$

原则II：局部性

检测区域仅限于输入空间中的**局部窗口**

- 远离位置(i, j)的参数$\mathbf{V}$不参与计算

解决方案：$|a|, |b| > \Delta \Rightarrow \mathbf{V}^{a,b} = 0$

$$\mathbf{H}_{i,j} = \sum_{a,b=-\Delta}^{\Delta} \mathbf{V}^{a,b} \mathbf{X}_{i+a,j+b}$$

$\mathbf{V}^{a,b}$被称为**卷积核 convolution kernel**或者**滤波器 filter**

- 即模型可学习的参数、权重

原则II：局部性

检测区域仅限于输入空间中的**局部窗口**

- 远离位置(i, j)的参数$\mathbf{V}$不参与计算

解决方案：$|a|, |b| > \Delta \Rightarrow \mathbf{V}^{a,b} = 0$

$$\mathbf{H}_{i,j} = \sum_{a,b=-\Delta}^{\Delta} \mathbf{V}^{a,b} \mathbf{X}_{i+a,j+b}$$

$\mathbf{V}^{a,b}$被称为**卷积核 convolution kernel**或者**滤波器 filter**

- 即模型可学习的参数、权重

注意：参数量再次大幅减少

小结：卷积计算

$$
\begin{aligned}
\mathbf{H}_{i,j} &= \mathbf{W}_{i,j}^{k,l}\mathbf{X}_{k,l} \\
&= \mathbf{W}_{i,j}^{a,b}\mathbf{X}_{i+a,j+b} \\
&\approx \mathbf{V}^{a,b}\mathbf{X}_{i+a,j+b} \\
\mathbf{H}_{i,j} &\approx \sum_{a,b=-\Delta}^{\Delta} \mathbf{V}^{a,b}\mathbf{X}_{i+a,j+b}
\end{aligned}
$$

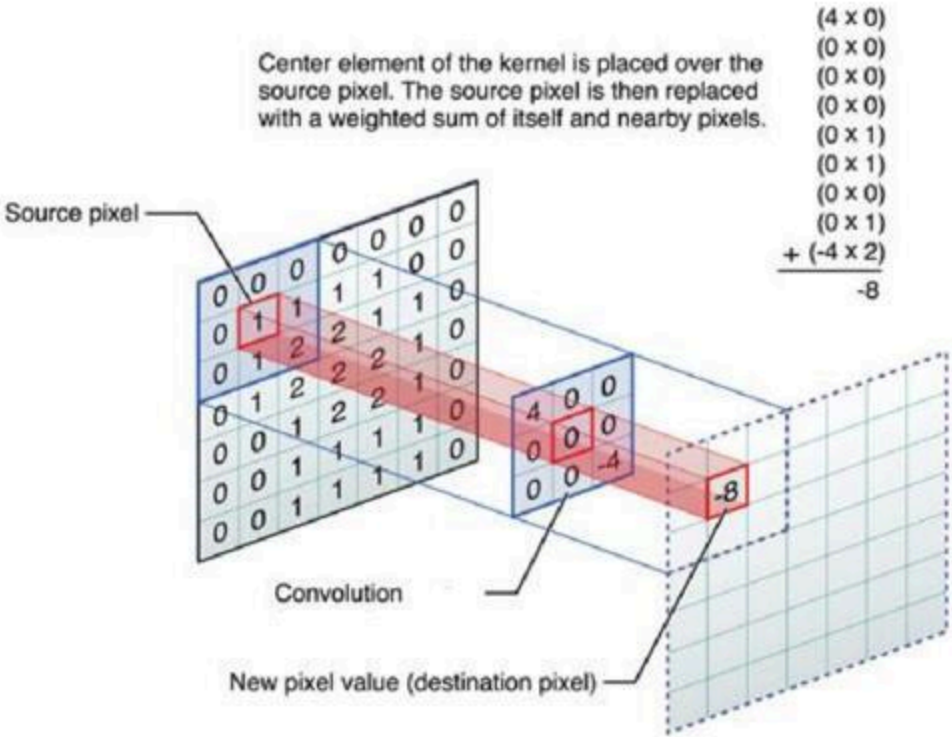
小结：卷积计算

$$
\begin{aligned}
\mathbf{H}_{i,j} &= \mathbf{W}_{i,j}^{k,l}\mathbf{X}_{k,l} \\
&= \mathbf{W}_{i,j}^{a,b}\mathbf{X}_{i+a,j+b} \\
&\approx \mathbf{V}^{a,b}\mathbf{X}_{i+a,j+b} \\
\mathbf{H}_{i,j} &\approx \sum_{a,b=-\Delta}^{\Delta} \mathbf{V}^{a,b}\mathbf{X}_{i+a,j+b}
\end{aligned}
$$

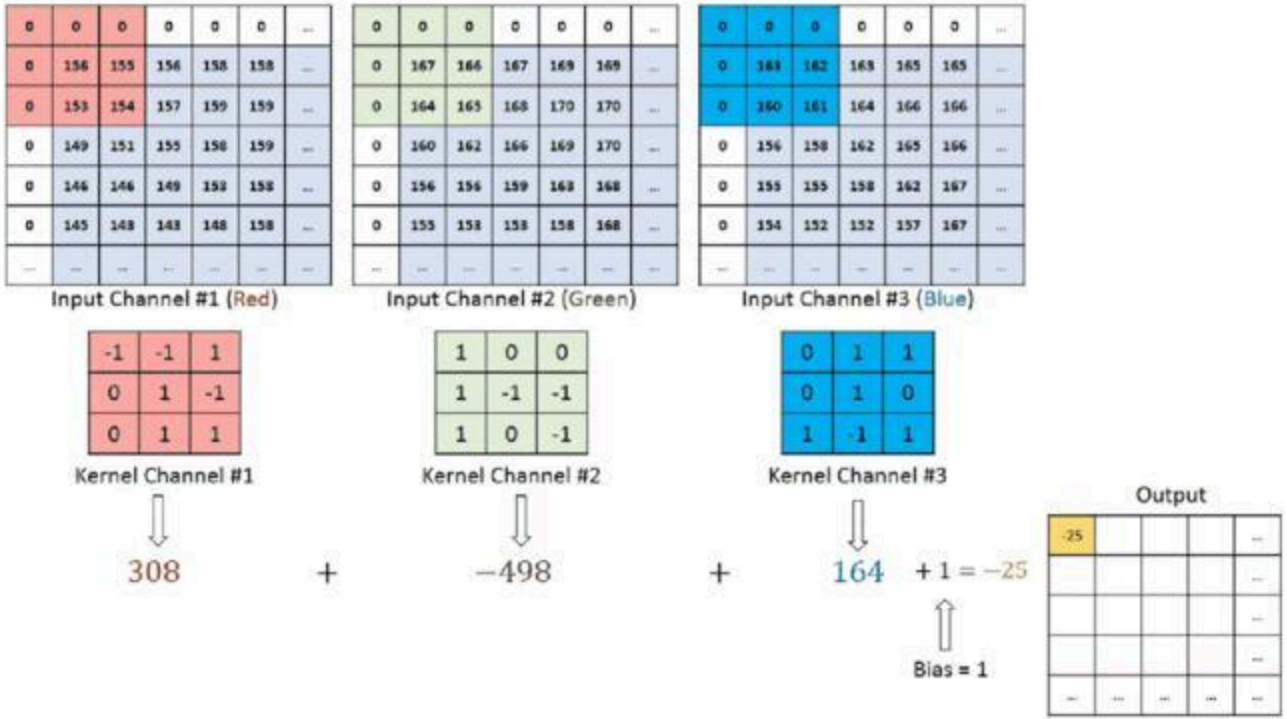
问题：局部观察无法获取覆盖范围较大的特征

图像卷积

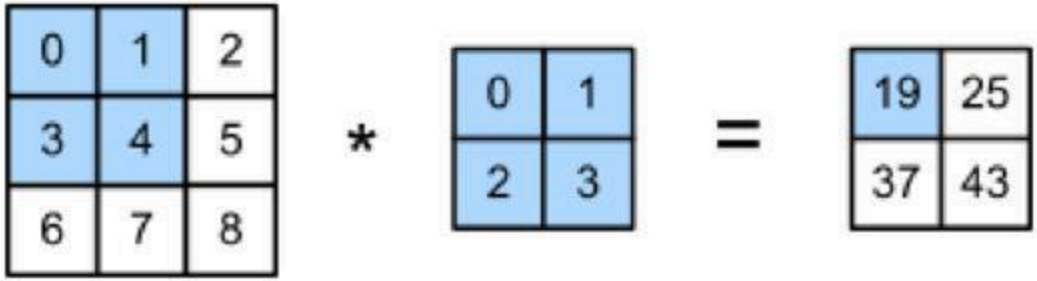
卷积运算图解


Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.
(4 x 0)
(0 x 0)
(0 x 0)
(0 x 0)
(0 x 1)
(0 x 1)
(0 x 0)
(0 x 1)
+ (-4 x 2)
-8
Source pixel
Convolution
New pixel value (destination pixel)

图像上的卷积运算


Input Channel #1 (Red)
Input Channel #2 (Green)
Input Channel #3 (Blue)
Kernel Channel #1
Kernel Channel #2
Kernel Channel #3
308
+
−498
+
164
+ 1 = −25
Bias = 1
Output

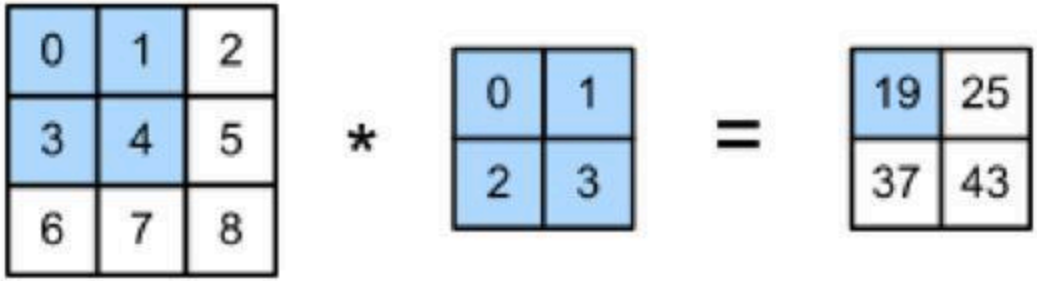
二维卷积层



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

- $\mathbf{X} \in \mathbb{R}^{n_h \times n_w}, \mathbf{W} \in \mathbb{R}^{k_h \times k_w}, b \in \mathbb{R}$

二维卷积层



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

- $\mathbf{X} \in \mathbb{R}^{n_h \times n_w}, \mathbf{W} \in \mathbb{R}^{k_h \times k_w}, b \in \mathbb{R}$

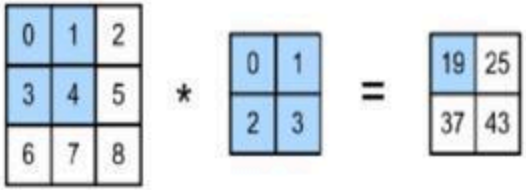
注意：输出尺寸略小于输入尺寸

- $\mathbf{Y} : (n_h - k_h + 1) \times (n_w - k_w + 1)$

多层卷积

应用卷积层：图像缩小，但核大小一般不变

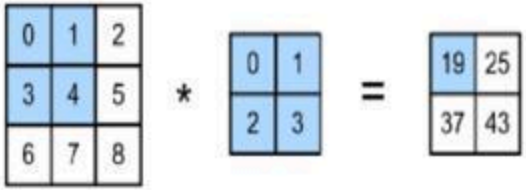
- 输出$\mathbf{Y}$变小：$(n_h - k_h + 1) \times (n_w - k_w + 1)$
- 可以看成观察窗口中的**特征提取**和**信息融合**



多层卷积

应用卷积层：图像缩小，但核大小一般不变

- 输出$\mathbf{Y}$变小：$(n_h - k_h + 1) \times (n_w - k_w + 1)$
- 可以看成观察窗口中的**特征提取**和**信息融合**



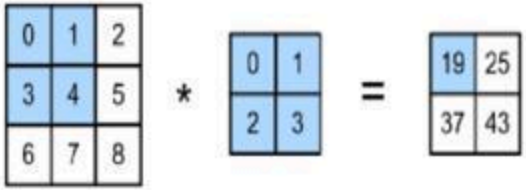
例如：28x28的图片，5x5的卷积核

- 第1层：24x24
- 第6层：4x4 < 核的大小
  - 5x5窗口：覆盖图像的全部信息

多层卷积

应用卷积层：图像缩小，但核大小一般不变

- 输出$\mathbf{Y}$变小：$(n_h - k_h + 1) \times (n_w - k_w + 1)$
- 可以看成观察窗口中的**特征提取**和**信息融合**



例如：28x28的图片，5x5的卷积核

- 第1层：24x24
- 第6层：4x4 < 核的大小
  - 5x5窗口：覆盖图像的全部信息

回顾：局部观察无法获取覆盖范围较大的特征?

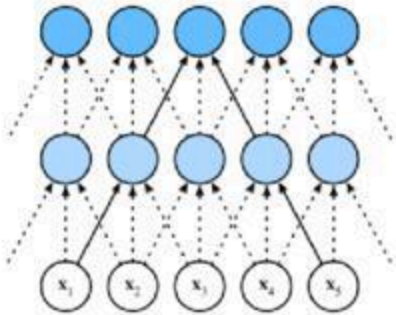
- 最高层：**融合整张图片的信息**
  - 小窗口也能观察到全局信息

特征映射、感受野

应用卷积层：图像缩小，但核大小一般不变

- 可以看成观察窗口中的**特征提取**和**信息融合**
- 最高层：**融合整张图片的信息**

想象：用显微镜观察目标；“站得高看得远”

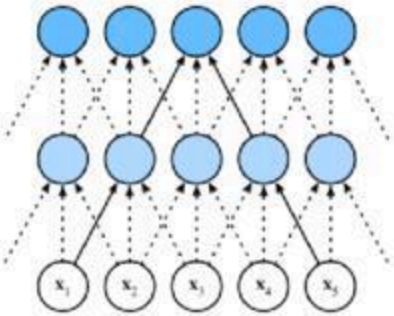


特征映射、感受野

应用卷积层：图像缩小，但核大小一般不变

- 可以看成观察窗口中的**特征提取**和**信息融合**
- 最高层：**融合整张图片的信息**

想象：用显微镜观察目标；“站得高看得远”

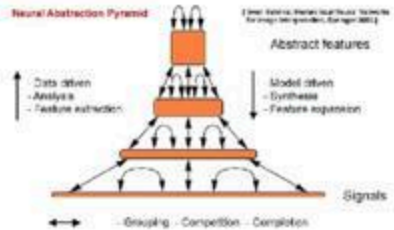


特征映射 feature map：特征提取的**过程（或结果）**

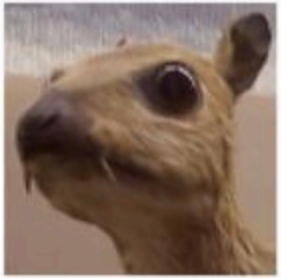
- 每个层将输入映射为下一个**特征空间中的表示**

感受野 receptive field：信息融合的**范围**

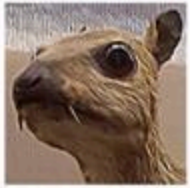
- 可能影响当前层中某元素**计算依赖**的所有先前层的元素



美颜：图像过滤



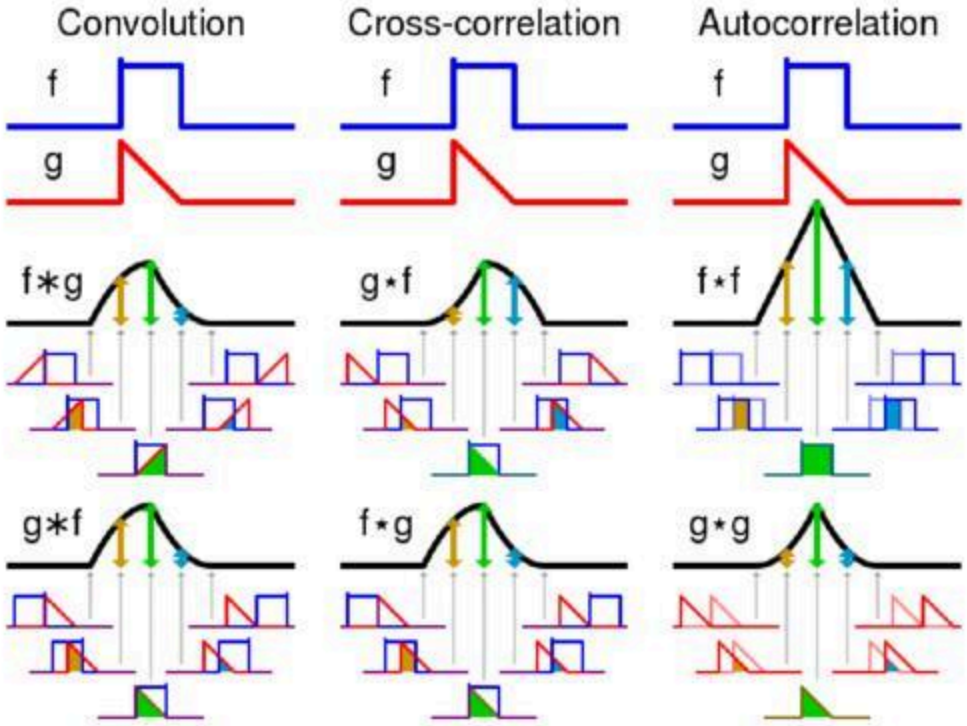
$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$



$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

拓展：卷积、交叉相关


Convolution
Cross-correlation
Autocorrelation
f
g
f∗g
g⋆f
f⋆f
g∗f
f⋆g
g⋆g

拓展：卷积、交叉相关公式

二维交叉相关

$$\mathbf{H}_{i,j} = \sum_{a,b=-\Delta}^{\Delta} \mathbf{V}^{a,b} \mathbf{X}_{i+a,j+b}$$

二维卷积：只是卷积核索引改为倒序

$$\mathbf{H}_{i,j} = \sum_{a,b=-\Delta}^{\Delta} \mathbf{V}^{-a,-b} \mathbf{X}_{i+a,j+b}$$

拓展：卷积、交叉相关公式

二维交叉相关

$$\mathbf{H}_{i,j} = \sum_{a,b=-\Delta}^{\Delta} \mathbf{V}^{a,b} \mathbf{X}_{i+a,j+b}$$

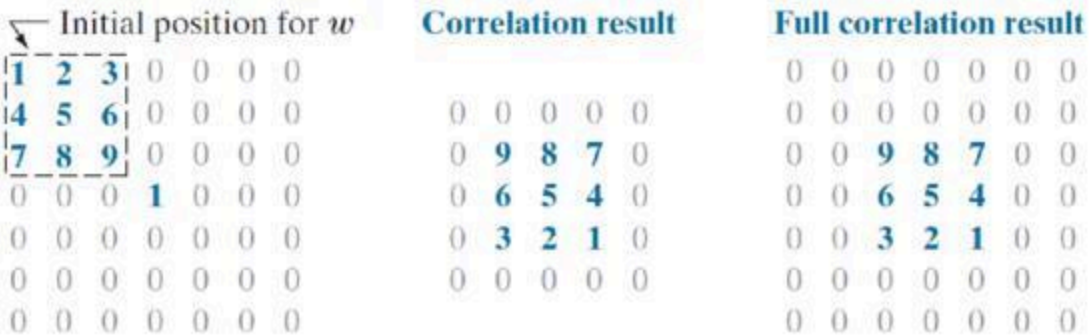
二维卷积：只是卷积核索引改为倒序

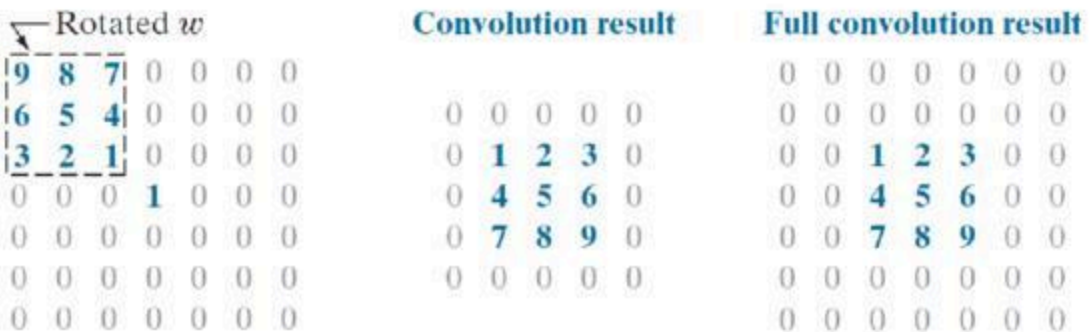
$$\mathbf{H}_{i,j} = \sum_{a,b=-\Delta}^{\Delta} \mathbf{V}^{-a,-b} \mathbf{X}_{i+a,j+b}$$

由于对称性：在实际应用中没有区别

拓展：卷积、交叉相关计算

区别：数学概念的约定俗成





其他应用

一维

$$\mathbf{y}_i = \sum_{a=-\Delta}^{\Delta} \mathbf{V}^a \mathbf{X}_{i+a}$$

- 序列数据
  - 文本、语言

三维

$$\mathbf{Y}_{i,j,k} = \sum_{a,b,c=-\Delta}^{\Delta} \mathbf{V}^{a,b,c} \mathbf{X}_{i+a,j+b,k+c}$$

- 时序影像
  - 视频、气象云图
- 断层扫描
  - 医学影像、地质

实验：图像卷积

小结：图像卷积

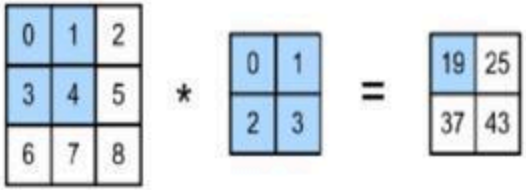
- 卷积层：输入与核的交叉相关，加上偏移后输出
- 可学习参数：核、偏移
- 超参数：核的大小

填充和步幅

填充

应用卷积层：图像缩小，但核大小一般不变

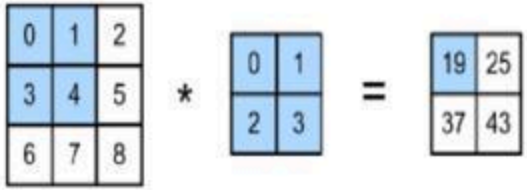
- 输出$\mathbf{Y}$变小：$(n_h - k_h + 1) \times (n_w - k_w + 1)$
- 可以看成观察窗口中的**特征提取**和**信息融合**



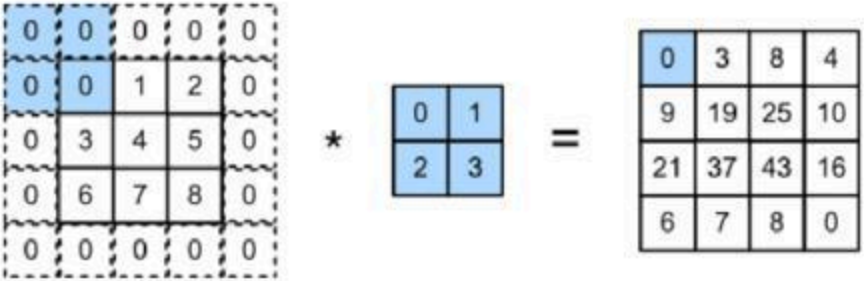
填充

应用卷积层：图像缩小，但核大小一般不变

- 输出$\mathbf{Y}$变小：$(n_h - k_h + 1) \times (n_w - k_w + 1)$
- 可以看成观察窗口中的**特征提取**和**信息融合**



填充：在图像周边添加额外的行/列

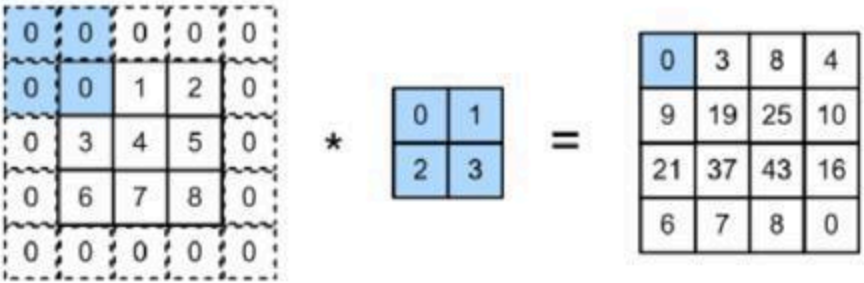


- 输出$\mathbf{Y}$：不一定缩小

填充：计算

假设填充p_h行、p_w列：两边同时填充，总数是p_h, p_w

- $\mathbf{Y}: (n_h - k_h + p_h + 1) \times (n_w - k_w + p_w + 1)$

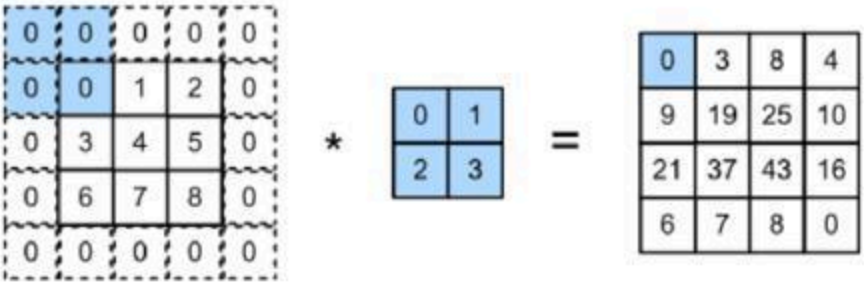


- 注意：卷积核大小通常是**奇数**，图只是示例

填充：计算

假设填充p_h行、p_w列：两边同时填充，总数是p_h, p_w

- $\mathbf{Y}$：$(n_h - k_h + p_h + 1) \times (n_w - k_w + p_w + 1)$



- 注意：卷积核大小通常是**奇数**，图只是示例

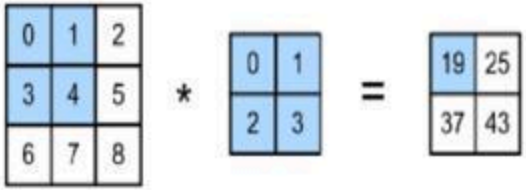
通常填充用于**保持图像大小不变**

- 计算得出：$p_h = k_h - 1, p_w = k_w - 1$

步幅

应用卷积层：图像缩小，但核大小一般不变

- 输出$\mathbf{Y}$变小：$(n_h - k_h + 1) \times (n_w - k_w + 1)$
- 可以看成观察窗口中的**特征提取**和**信息融合**



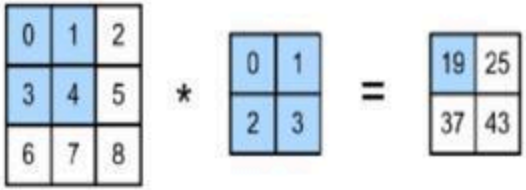
注意：图像缩小量是层数的**线性**函数

- 例如：224x224的图片，5x5的卷积核
  - 需要55层才能缩到4x4

步幅

应用卷积层：图像缩小，但核大小一般不变

- 输出$\mathbf{Y}$变小：$(n_h - k_h + 1) \times (n_w - k_w + 1)$
- 可以看成观察窗口中的**特征提取**和**信息融合**



注意：图像缩小量是层数的**线性**函数

- 例如：224x224的图片，5x5的卷积核
  - 需要55层才能缩到4x4

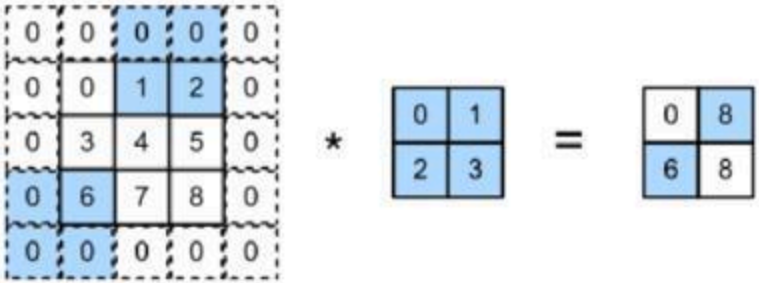
层数越多：模型越复杂，计算量越大

- 大多数实际问题不需要像素级分辨能力：无效计算
  - 例如：拍摄时手抖，偏移1个像素
- 需要**快速缩小窗口**的方法

步幅：计算

步幅：卷积核在行/列上每次滑动的位移量

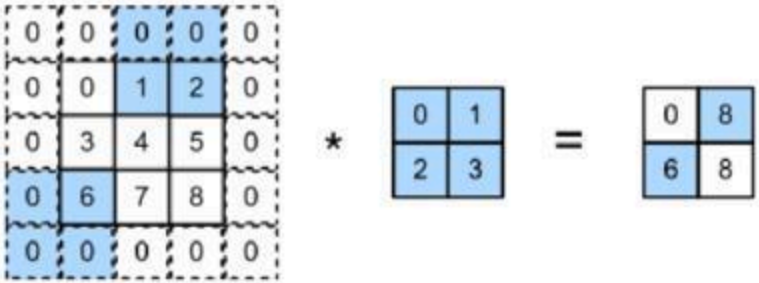
- 例如：纵向3、横向2的步幅



步幅：计算

步幅：卷积核在行/列上每次滑动的位移量

- 例如：纵向3、横向2的步幅



假设纵向、横向上的步幅分别是s_h、s_w

- $\mathbf{Y}$：$\lfloor (n_h - k_h + p_h + 1)/s_h \rfloor \times \lfloor (n_w - k_w + p_w + 1)/s_w \rfloor$

实验：填充和步幅

小结：填充和步幅

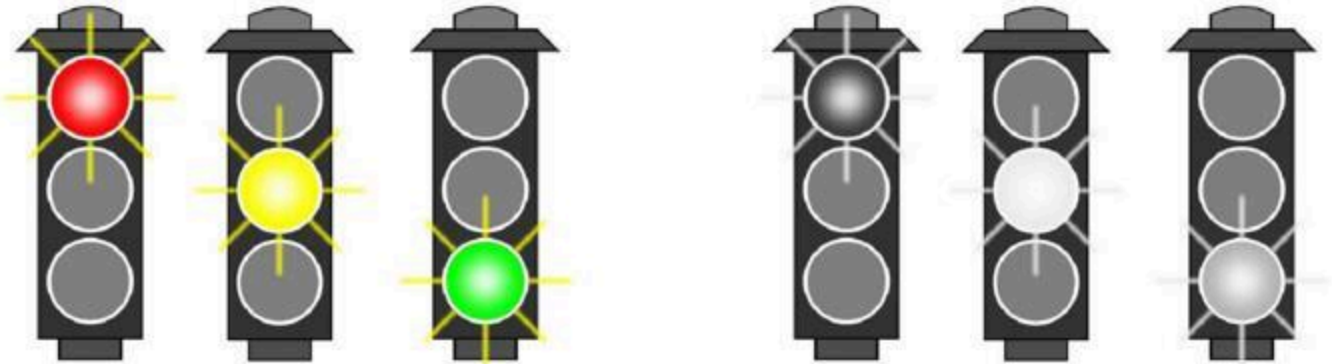
- 填充和步幅：卷积层的超参数
- 填充：在图像周边添加额外的行、列
  - 控制输出形状的缩小量
- 步幅：卷积核在行、列上每次滑动的位移量
  - 加倍减少输出形状

多输入多输出通道

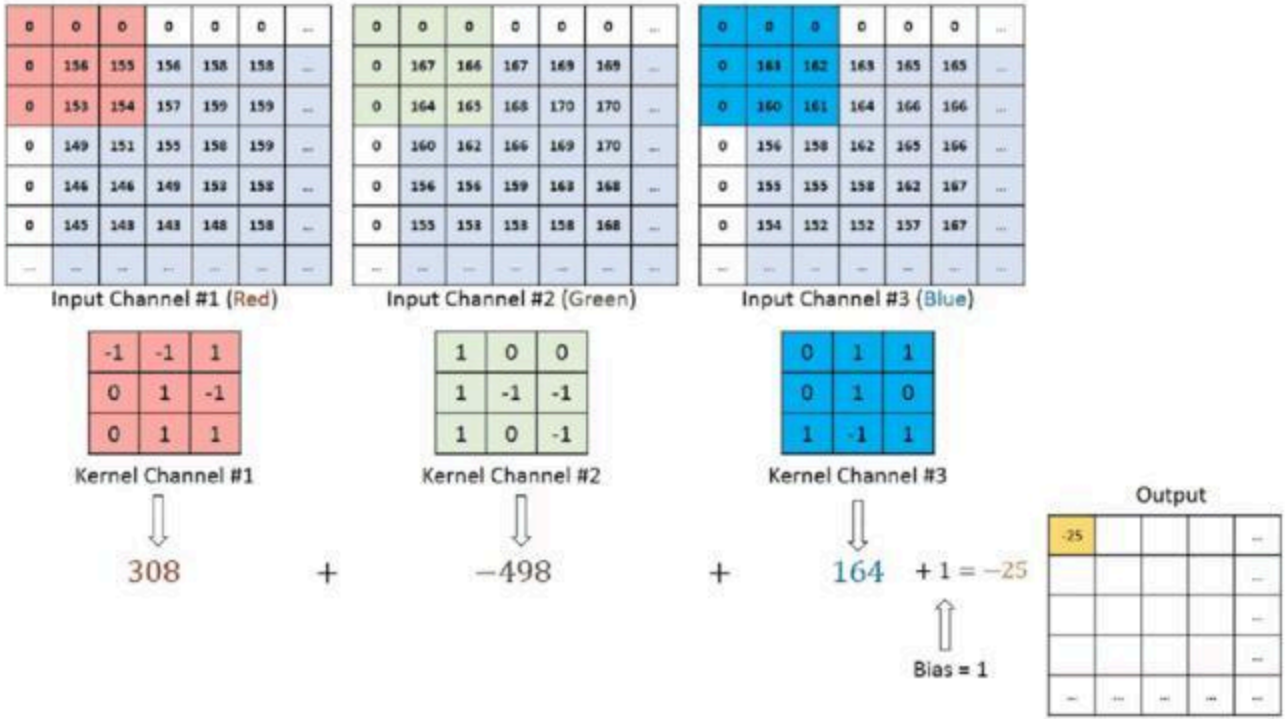
彩色图片上的卷积

彩色图片有多个通道，即**特征**：颜色是最常见的特征描述

- Jpeg：RGB；PNG：RGBA
- 转换为灰度值？三个通道的加权组合：丢失信息



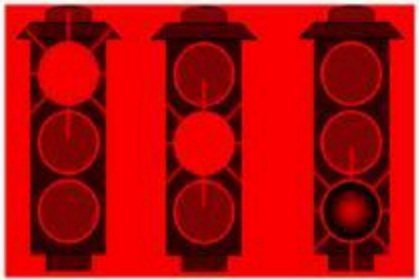
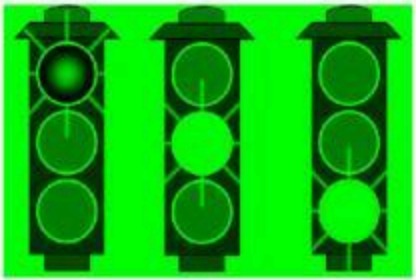
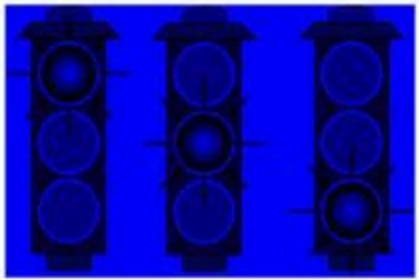
回顾：图像上的卷积运算


Input Channel #1 (Red)
Input Channel #2 (Green)
Input Channel #3 (Blue)
Kernel Channel #1
Kernel Channel #2
Kernel Channel #3
308
+
−498
+
164
+ 1 = −25
Bias = 1
Output

彩色图片的多个通道

彩色图片有多个通道，即**特征**：颜色是最常见的特征描述

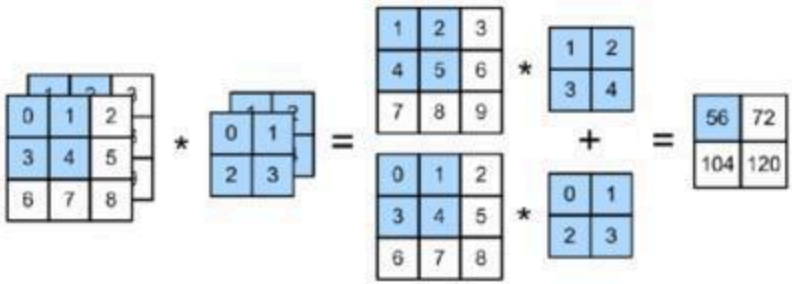
- 对每个通道**分别处理**：卷积核参数不同，但大小、位置相同

多输入通道：计算

每个通道应用一个卷积核

- 最终输出：所有处理之后的**通道求和**
  - RGB图片本身也是三个颜色混合而成

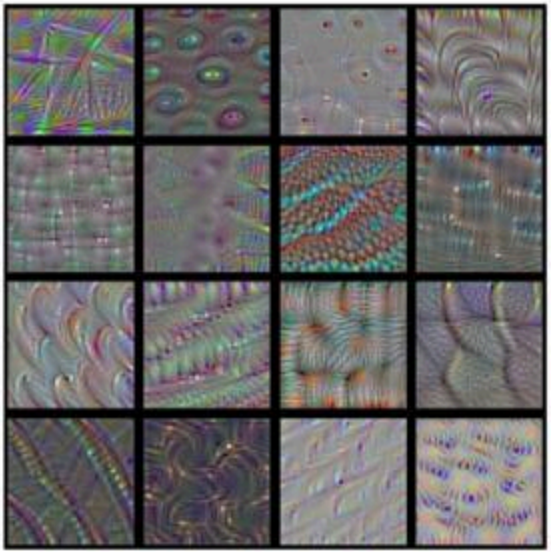


$$\mathbf{Y} = \sum_{k=0}^{c_i} \mathbf{X}_{k,:,:} \star \mathbf{W}_{k,:,:}$$

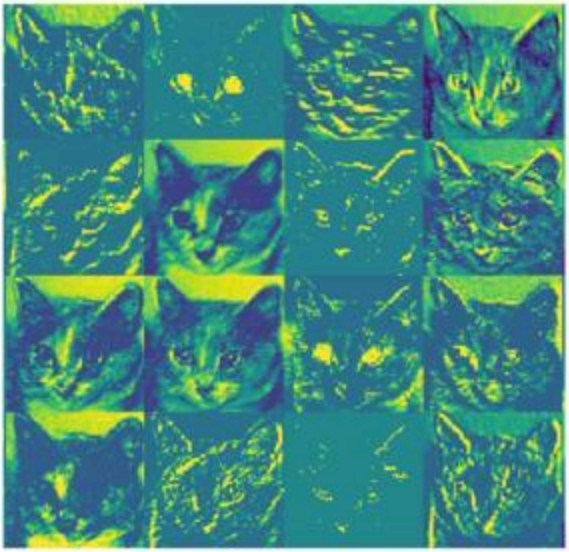
多输出通道

通道：**特征映射**；卷积核作用于每个通道，得到激活值

卷积核：可识别的**模式**



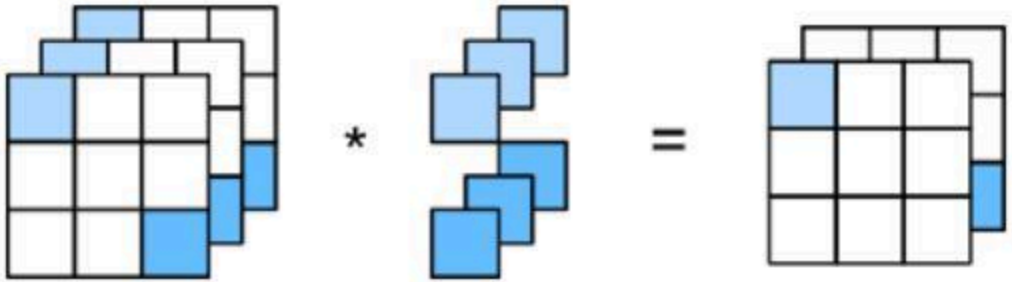
激活值：模式的**匹配度**



- 也有无效激活：背景

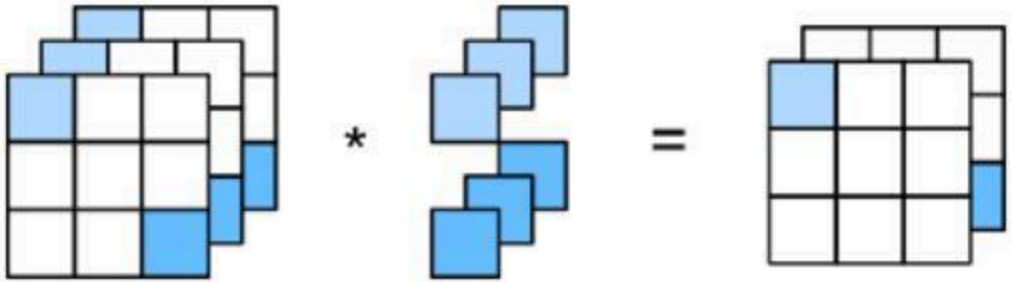
1x1 卷积层

1x1 卷积：不识别模式，只融合通道



1x1 卷积层

1x1 卷积：不识别模式，只融合通道



- 相当于**全连接**：每个像素的**通道维度**上可以看成感知机
  - 输入：$n_h n_w \times c_i$；权重（卷积核）：$c_o \times c_i$

二维卷积：计算复杂度

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

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

二维卷积：计算复杂度

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

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

计算复杂度：$O(c_i c_o k_h k_w m_h m_w)$

- 对**输出$\mathbf{Y}$的每个位置**计算一次卷积
  - 注意：与输入$\mathbf{X}$的大小无关；输入主要影响**存储**、**传输**

二维卷积：计算复杂度

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

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

计算复杂度：$O(c_i c_o k_h k_w m_h m_w)$

- 对**输出$\mathbf{Y}$的每个位置**计算一次卷积
  - 注意：与输入$\mathbf{X}$的大小无关；输入主要影响**存储**、**传输**
- 例如$c_i = c_o = 100, k_h = k_w = 5, m_h = m_w = 64 \Rightarrow$ 1GFLOPS

二维卷积：计算复杂度

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

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

计算复杂度：$O(c_i c_o k_h k_w m_h m_w)$

- 对**输出$\mathbf{Y}$的每个位置**计算一次卷积
  - 注意：与输入$\mathbf{X}$的大小无关；输入主要影响**存储**、**传输**
- 例如$c_i = c_o = 100, k_h = k_w = 5, m_h = m_w = 64 \Rightarrow$ 1GFLOPS
- 10层，ImageNet总共1M样本 $\Rightarrow$ 10 PFLOPS
  - CPU: 0.15TF $\Rightarrow$ 18h; GPU: 12TF $\Rightarrow$ 14min

二维卷积：计算复杂度

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

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

计算复杂度：$O(c_i c_o k_h k_w m_h m_w)$

- 对**输出$\mathbf{Y}$的每个位置**计算一次卷积
  - 注意：与输入$\mathbf{X}$的大小无关；输入主要影响**存储**、**传输**
- 例如$c_i = c_o = 100, k_h = k_w = 5, m_h = m_w = 64 \Rightarrow$ 1GFLOPS
- 10层，ImageNet总共1M样本 $\Rightarrow$ 10 PFLOPS
  - CPU: 0.15TF $\Rightarrow$ 18h; GPU: 12TF $\Rightarrow$ 14min
- 这只是一个数据轮次而已

推论：不能不买N卡

实验：多输入多输出通道

小结：多输入多输出通道

- （每个层的）输出通道数是超参数
- 每个输入通道应用独立的卷积核

池化层

池化：背景

提高数值计算的**稳定性**

- 卷积计算对位置敏感，但像素级分辨能力完全没必要

黑白交界

$$\begin{matrix} 1. & 1. & 0. & 0. \\ 1. & 1. & 0. & 0. \\ 1. & 1. & 0. & 0. \\ 1. & 1. & 0. & 0. \end{matrix}$$

卷积输出

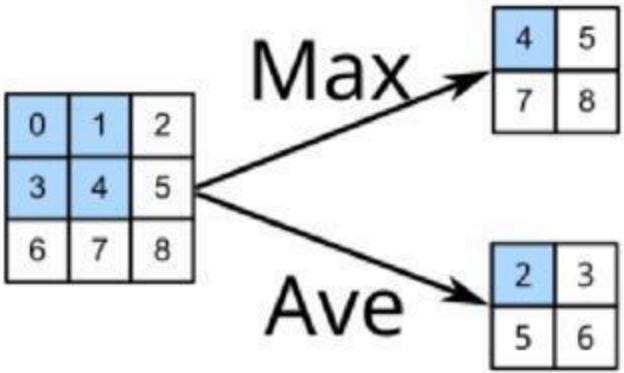
$$\begin{matrix} 0. & 1. & 0. & 0. \\ 0. & 1. & 0. & 0. \\ 0. & 1. & 0. & 0. \\ 0. & 1. & 0. & 0. \end{matrix}$$

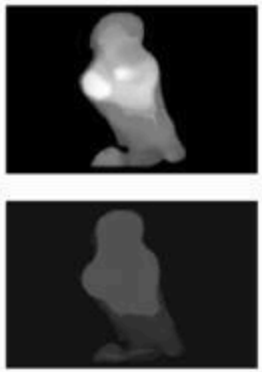
- 需要对**形变**有适当的容忍度
  - 以及照明，比例，手抖等

池化

池化 pooling：为固定形状窗口计算输出

- 没有可学习参数
- 遍历输入的所有位置

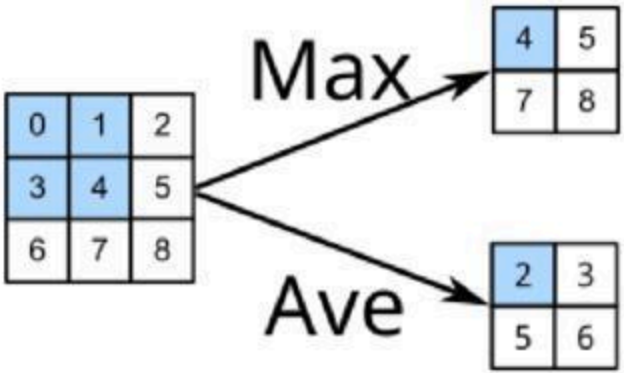


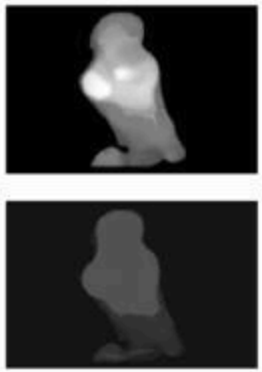


池化

池化 pooling：为固定形状窗口计算输出

- 没有可学习参数
- 遍历输入的所有位置





一般来说：最大池化对**检测、识别**类任务更有效

池化：作用

提高数值计算的**稳定性**

- 降低卷积层对位置的敏感性
- 降低对特征空间映射的敏感性

黑白交界

$$\begin{matrix} 1. & 1. & 0. & 0. \\ 1. & 1. & 0. & 0. \\ 1. & 1. & 0. & 0. \\ 1. & 1. & 0. & 0. \end{matrix}$$

卷积输出

$$\begin{matrix} 0. & 1. & 0. & 0. \\ 0. & 1. & 0. & 0. \\ 0. & 1. & 0. & 0. \\ 0. & 1. & 0. & 0. \end{matrix}$$

- 丢失一列信息

2x2最大池化

$$\begin{matrix} 1. & 1. & 0. & 0. \\ 1. & 1. & 0. & 0. \\ 1. & 1. & 0. & 0. \\ 1. & 1. & 0. & 0. \end{matrix}$$

- 重新还原

填充、步幅、多通道

池化层与卷积层类似：都具有填充和步幅

- 没有可学习的参数

填充、步幅、多通道

池化层与卷积层类似：都具有填充和步幅

- 没有可学习的参数

对每个输入通道应用池化层

- 输出通道数 = 输入通道数

实验：池化层

小结：池化层

- 池化层：返回固定窗口中的最大值或平均值
- 提高数值计算的稳定性
- 超参数：窗口大小、填充、步幅

卷积神经网络（LeNet）

回顾：从全连接到卷积

回顾：MLP用于Fashion-MNIST数据集

- 首先将28x28的图像**展平**为784维向量

回顾：从全连接到卷积

回顾：MLP用于Fashion-MNIST数据集

- 首先将28x28的图像**展平**为784维向量

卷积：直接在**二维图像空间**操作

- 模型更简洁、参数（计算量）更少

回顾：从全连接到卷积

回顾：MLP用于Fashion-MNIST数据集

- 首先将28x28的图像**展平**为784维向量

卷积：直接在**二维图像空间**操作

- 模型更简洁、参数（计算量）更少
- 保留空间结构：**识别二维模式**
  - 例如Logo、袖长

手写数字识别

LeNet：最早的、能够高效解决实际工业问题的卷积神经网络

- 邮政、银行：识别手写数字

手写数字识别

LeNet：最早的、能够高效解决实际工业问题的卷积神经网络

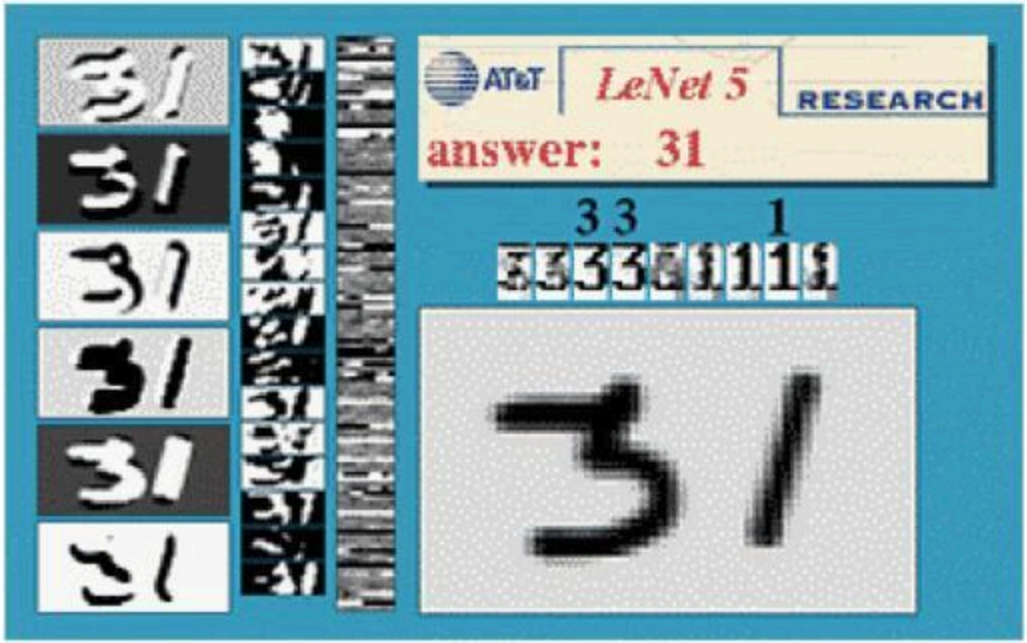
- 邮政、银行：识别手写数字

- 当前：**所有计算机视觉任务**的基础解决方案
- Yann LeCun：2018年Turing机获得者之一

MNIST数据集

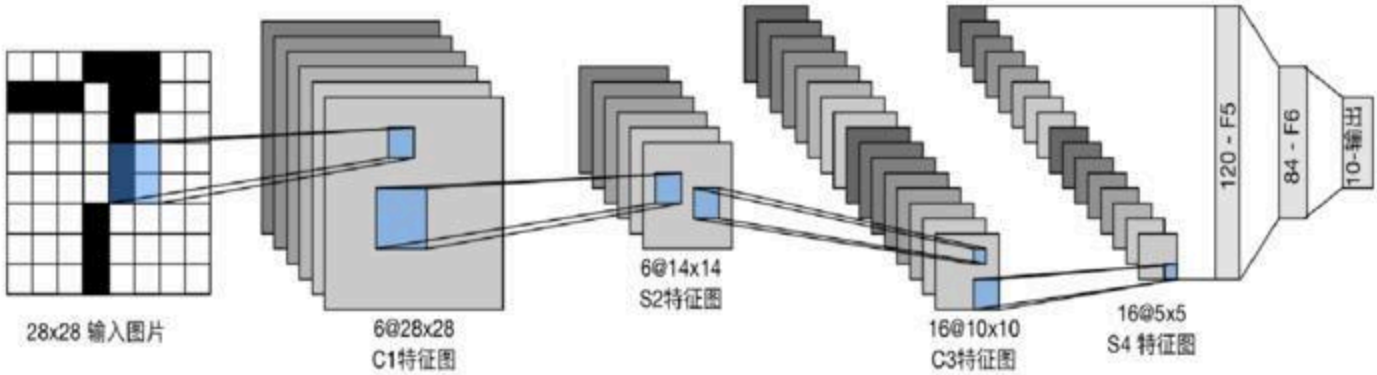
- 5万个训练数据
- 1万个测试数据
- 图像大小：28x28
- 10个类

LeNet：演示



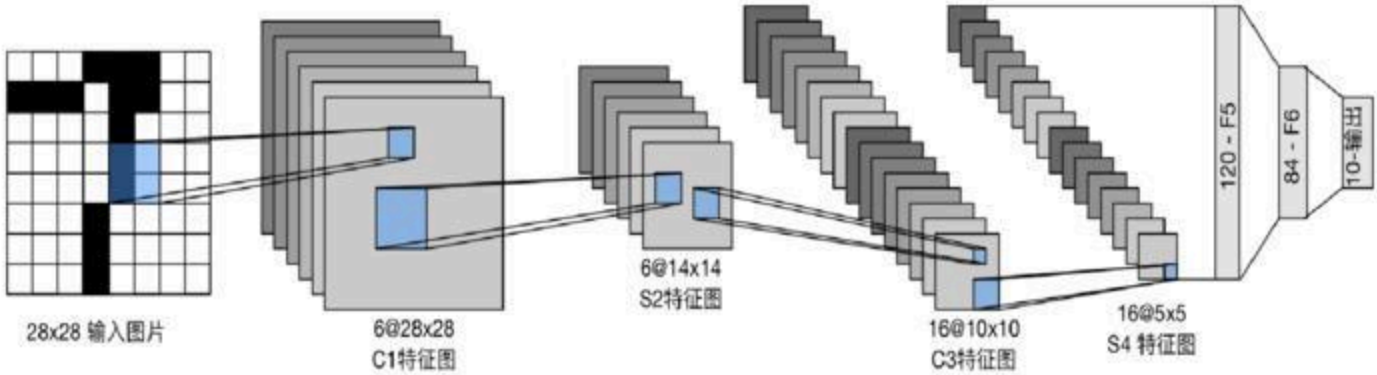
LeNet：架构

LeNet（LeNet-5）由两个部分组成：



LeNet：架构

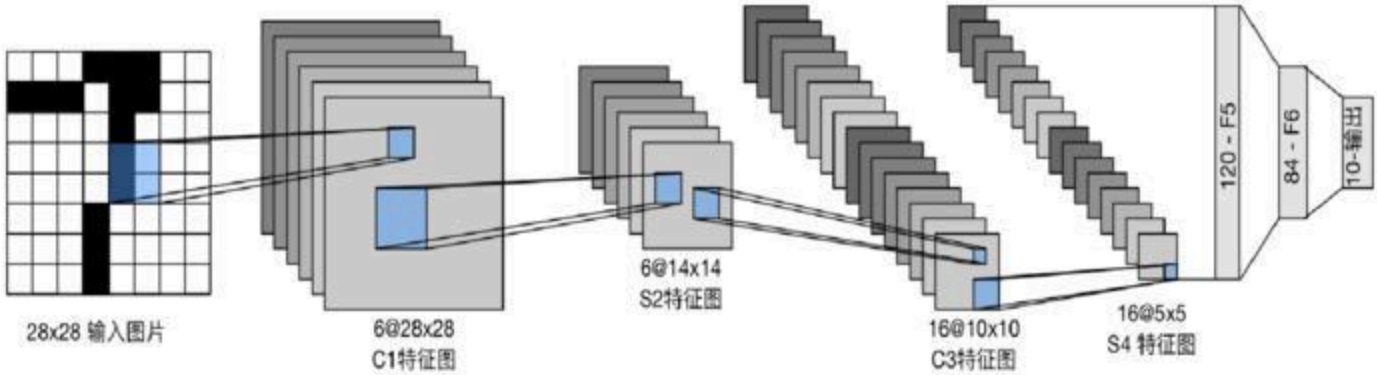
LeNet（LeNet-5）由两个部分组成：



- 卷积**编码器**：由两个卷积模块组成
  - 每个卷积层紧接2倍下采样：“浓缩信息”

LeNet：架构

LeNet（LeNet-5）由两个部分组成：

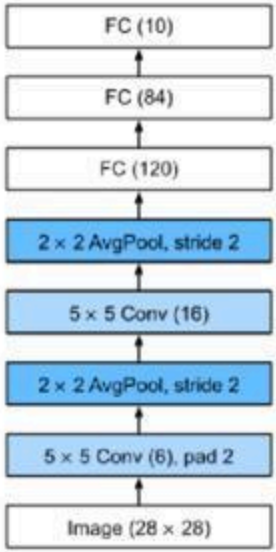


- 卷积**编码器**：由两个卷积模块组成
  - 每个卷积层紧接2倍下采样：“浓缩信息”
- **全连接**模块：由三个全连接层组成
  - 将编码逐步转换成输出格式

LeNet：架构 I

LeNet（LeNet-5）由两个部分组成：先看卷积层

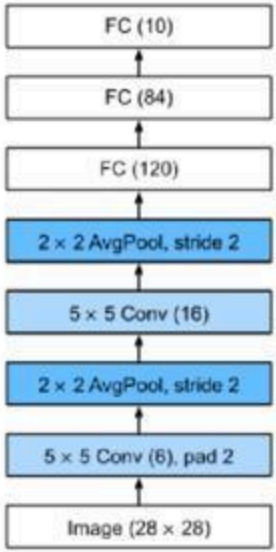
- 卷积**编码器**：由两个卷积层组成
  - 学习（映射到二维）**特征空间表示**
    - 逐步缩小特征映射、增加通道数量
  - 5x5卷积核、sigmoid激活、平均池化
    - ReLU和最大池化的研究当时还没出现



LeNet：架构 II

LeNet（LeNet-5）由两个部分组成：再看全连接层

- 卷积**编码器**：由两个卷积层组成
- **全连接**模块：由三个全连接层组成
  - 逐步降维到**类别空间表示**
    - 首先展平成向量
    - 逐步降维：防止出现“信息瓶颈”



实验：LeNet

小结：LeNet

- LeNet是早期神经网络的成功案例
- 架构分为两个部分
  - 卷积层：学习空间信息
  - 全连接层：转换到类别

Review

本章内容

图像识别的两个原则。图像卷积。填充、步幅。多输入、输出通道。池化。LeNet。

重点：平移不变性、局部性；图像卷积计算；卷积层输出、卷积核的含义；填充、步幅的作用；多通道的含义、1x1 卷积的作用；池化的作用；LeNet的架构设计。

难点：平移不变性、局部性的计算形式。

学习目标

- 理解图像识别的两个原则：平移不变性、局部性
- 了解图像识别两个原则的计算形式
- 理解图像卷积的计算方法
- 理解卷积层输出特征映射、卷积核代表感受野
- 理解填充、步幅的作用、计算公式
- 理解多通道的含义、1x1 卷积的作用
- 理解池化的作用
- 理解LeNet的架构设计

问题

简述图像识别两个原则的解释、计算。

简述图像卷积的计算方法。

简述卷积层输出、卷积核的含义。

简述填充、步幅的作用、计算公式。

简述1x1 卷积的作用。

简述池化的作用。

简述LeNet的架构设计。