

第五讲： 卷积神经网络

Convolutional Neural Network (CNN)

从像素到空间结构

廖振宇

电子信息与通信学院
华中科技大学
2026 年秋季学期



本讲在课程中的位置



本讲：利用图像空间结构；下一讲：学习图像分布

本讲内容

01

卷积怎样利用图像结构

02

怎样验证卷积结构的作用

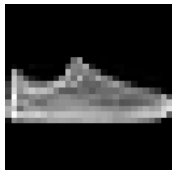
03

CNN 怎样形成层次表示

1

卷积怎样利用图像结构

测试 #12



真实 运动鞋
预测 凉鞋

测试 #23



真实 凉鞋
预测 凉鞋

测试 #25



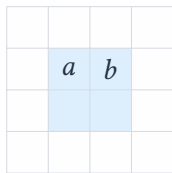
真实 外套
预测 套头衫

- 3 线性模型：像素线性加权
- 4 MLP：非线性表示
- 5 CNN：把图像结构写入网络

思考

鞋底、袖口等局部特征出现在新位置时，模型成功识别？

数据展平 + MLP 没有利用像素空间关系



$$\mathbf{x} = \text{vec}(X)$$

$\dots, a, b, \dots$

图像二维邻接关系

$$\mathbf{h} = \text{ReLU}(\mathbf{W}_1^\top \mathbf{x} + \mathbf{b}_1)$$

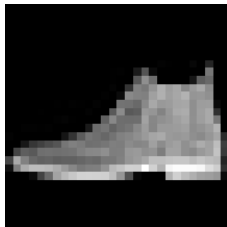
- 向量能按原顺序还原成图像
- 相邻位置没有被 MLP 结构“特殊对待”
- 本质是更“普适”的设计方案

结论

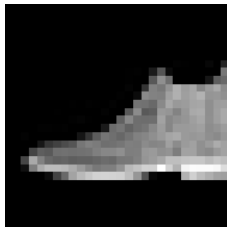
局限：模型的连接与参数组织方式，而不是展平操作本身

同一张图像，改变位置或像素排列

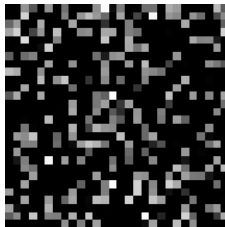
原图



右移 2 像素



固定像素置换



实际样本 | 平移补零并裁切边界 | 所有图像使用同一变换

思考

- 两种变换保留了局部结构和特征？
- 如果训练和测试都使用变换后图片，MLP 与 CNN 可能会有怎样表现？

- **局部相关**：相邻的像素共同描述边缘、纹理和局部形状
- **模式重复**：同一种边缘或纹理可能出现在图像的不同位置
- **层次组合**：局部特征在更大（空间）范围形成组合，对任务起决定性作用

结构先验

针对**特定任务**，在看到训练数据之前，就写入模型结构的假设

结论

- 卷积的**局部连接与权重共享**对应前两条先验规则
- 进一步通过（类似 MLP 的）多层堆叠扩大信息整合范围

局部连接：让一个单元只读取一个邻域

$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 & 0 & 0 \\ 0 & b_1 & b_2 & b_3 & 0 \\ 0 & 0 & c_1 & c_2 & c_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}$$

- 每个神经元只访问连续三个像素，与其他像素无关
- 不同每个神经元仍有不同参数

思考

同一种特征移到图片的其他位置，是否还需要/应当重新学习（一组）参数？

权重共享：让同一个检测规则滑过所有位置

$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} w_1 & w_2 & w_3 & 0 & 0 \\ 0 & w_1 & w_2 & w_3 & 0 \\ 0 & 0 & w_1 & w_2 & w_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}$$

定义 | 卷积核与权重共享

一组可学习权重在**不同空间位置**重复使用，产生一张特征图

- 局部连接：减少非零连接
- 参数共享：约束非零元素**相等**

结论

局部连接和权重共享是两种不同的结构约束

1 实例：二维卷积计算

$$X = \begin{bmatrix} 1 & 0 & 2 \\ 2 & 1 & 0 \\ 0 & 1 & 3 \end{bmatrix} \quad K = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad Z = \begin{bmatrix} 0 & 0 \\ 1 & -2 \end{bmatrix}$$

得到

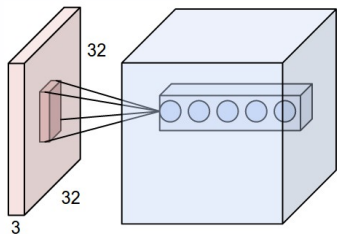
$$Z_{0,0} = 1 \times 1 + 0 \times 0 + 2 \times 0 + 1 \times (-1) = 0$$

- 每次取一个 2×2 邻域，与同一个核逐元素相乘后求和
- 步幅为 1 (stride = 1)、不补边 (no padding)、偏置为 0，得到 2×2 输出

注

和信号与线性系统对比，深度学习将不翻转卷积核的互相关运算也称作卷积

多通道输入：一个核作用于全部输入通道



先看一个输出单元连向输入的区域

来源：CS231n: Convolutional Neural Networks, Local Connectivity

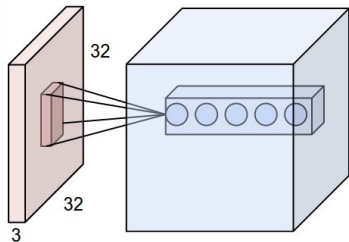
- RGB 彩色图片输入尺寸
 $32 \times 32 \times 3$
- 卷积核尺寸 $5 \times 5 \times 3$
- 三个通道的点积结果相加
- 加一个偏置，得到一个输出值

结论

输入三通道 $\Rightarrow$ 每个核覆盖三通道，产生一张特征图

多个卷积核产生多“层”输出特征图

$$K \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}} \times k_h \times k_w}$$



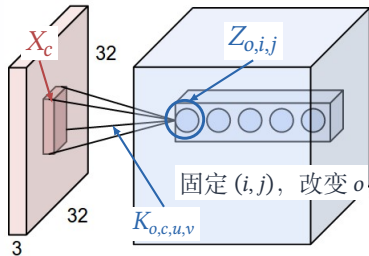
再看同一位置沿深度排列的 5 个单元

- 一个核产生一“层”输出特征图（feature map）
- 每个核作用于全部输入通道
- 图中 5 个单元读取同一区域，但使用不同卷积核

思考

使用 16 个核时，网络第一层输出几“层”特征图，或输出几通道？

来源：CS231n: Convolutional Neural Networks, Local Connectivity



来源：CS231n: Convolutional Neural Networks, Local

Connectivity

定义 | 二维卷积

$$Z_{o,i,j} = b_o + \sum_{c=0}^{C_{in}-1} \sum_{u=0}^{k_h-1} \sum_{v=0}^{k_w-1} K_{o,c,u,v} X_{c, is_h+u, js_w+v}$$

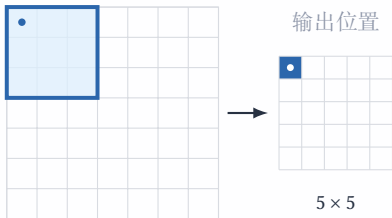
- X 为补边后输入, b_o 为通道 o 的偏置
- c, o : 输入、输出通道索引
- (i, j) : 输出位置; (u, v) : 核内偏移
- s_h, s_w : 高、宽方向的步幅
- 权重共享: $K_{o,c,u,v}$ 不随输出位置 (i, j) 改变

1 Stride 决定滑窗每次移动多远

7×7 输入, 3×3 卷积核, 不补边 ($p = 0$)

步幅 $s = 1$

从左上角开始



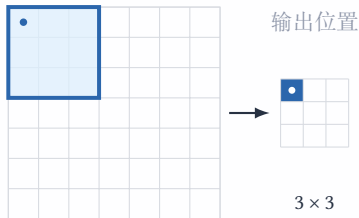
输入起点 $(0,0) \rightarrow$ 输出 $Z_{0,0}$

蓝框: 当前窗口 | 虚线: 上一个窗口 | 深蓝格: 当前输出 | 索引从 0 开始

逐步滑动

步幅 $s = 2$

从左上角开始



输入起点 $(0,0) \rightarrow$ 输出 $Z_{0,0}$

结论

一处滑窗对应一个输出位置; 增大步幅会减少合法起点

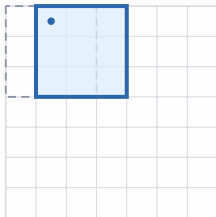
示意依据: CS231n: Spatial Arrangement 与 Convolution Demo | 沿用旧课件的 7×7 滑窗例子

1 Stride 决定滑窗每次移动多远

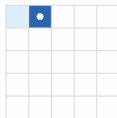
7×7 输入，3×3 卷积核，不补边 ($p=0$)

步幅 $s=1$

右移 1 格



输出位置



5×5

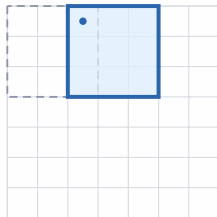
输入起点 (0,1) → 输出 $Z_{0,1}$

蓝框：当前窗口 | 虚线：上一个窗口 | 深蓝格：当前输出 | 索引从 0 开始

逐步滑动

步幅 $s=2$

右移 2 格



输出位置



3×3

输入起点 (0,2) → 输出 $Z_{0,1}$

结论

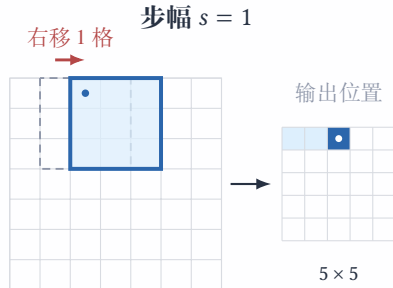
一处滑窗对应一个输出位置；增大步幅会减少合法起点

示意依据：CS231n: Spatial Arrangement 与 Convolution Demo | 沿用旧课件的 7×7 滑窗例子

1 Stride 决定滑窗每次移动多远

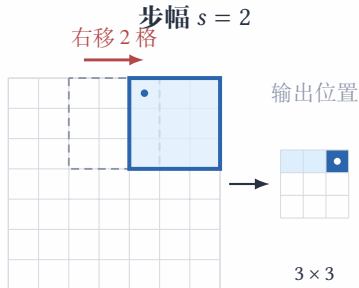
7×7 输入，3×3 卷积核，不补边 ($p=0$)

逐步滑动



输入起点 (0,2) → 输出 $Z_{0,2}$

蓝框：当前窗口 | 虚线：上一个窗口 | 深蓝格：当前输出 | 索引从 0 开始



输入起点 (0,4) → 输出 $Z_{0,2}$

结论

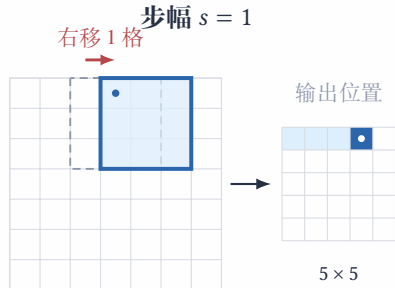
一处滑窗对应一个输出位置；增大步幅会减少合法起点

示意依据：CS231n: Spatial Arrangement 与 Convolution Demo | 沿用旧课件的 7×7 滑窗例子

1 Stride 决定滑窗每次移动多远

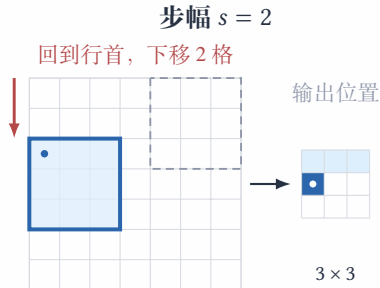
7×7 输入, 3×3 卷积核, 不补边 ($p = 0$)

逐步滑动



输入起点 $(0,3) \rightarrow$ 输出 $Z_{0,3}$

蓝框: 当前窗口 | 虚线: 上一个窗口 | 深蓝格: 当前输出 | 索引从 0 开始



输入起点 $(2,0) \rightarrow$ 输出 $Z_{1,0}$

结论

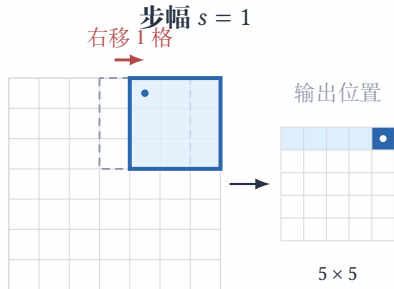
一处滑窗对应一个输出位置; 增大步幅会减少合法起点

示意依据: CS231n: Spatial Arrangement 与 Convolution Demo | 沿用旧课件的 7×7 滑窗例子

1 Stride 决定滑窗每次移动多远

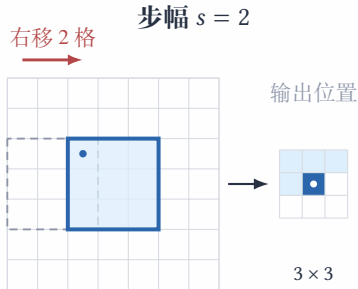
7×7 输入, 3×3 卷积核, 不补边 ($p = 0$)

逐步滑动



输入起点 $(0, 4) \rightarrow$ 输出 $Z_{0,4}$

蓝框: 当前窗口 | 虚线: 上一个窗口 | 深蓝格: 当前输出 | 索引从 0 开始



输入起点 $(2, 2) \rightarrow$ 输出 $Z_{1,1}$

结论

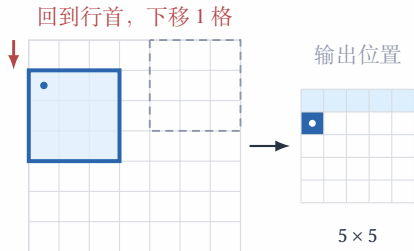
一处滑窗对应一个输出位置; 增大步幅会减少合法起点

示意依据: CS231n: Spatial Arrangement 与 Convolution Demo | 沿用旧课件的 7×7 滑窗例子

1 Stride 决定滑窗每次移动多远

7 × 7 输入，3 × 3 卷积核，不补边 ($p = 0$)

步幅 $s = 1$

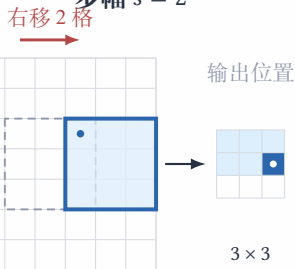


输入起点 (1, 0) → 输出 $Z_{1,0}$

蓝框：当前窗口 | 虚线：上一个窗口 | 深蓝格：当前输出 | 索引从 0 开始

逐步滑动

步幅 $s = 2$



输入起点 (2, 4) → 输出 $Z_{1,2}$

结论

一处滑窗对应一个输出位置；增大步幅会减少合法起点

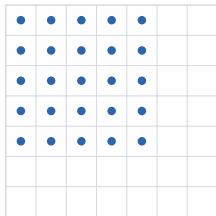
示意依据：CS231n: Spatial Arrangement 与 Convolution Demo | 沿用旧课件的 7 × 7 滑窗例子

1 Stride 决定滑窗每次移动多远

7 × 7 输入，3 × 3 卷积核，不补边 ($p = 0$)

步幅 $s = 1$

每行 5 个合法起点



输出位置

5 × 5

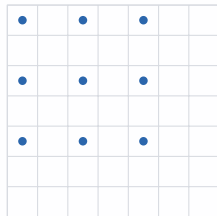
5 行 × 5 列 = 25 个输出位置

蓝点：合法窗口的左上角 | 沿高、宽两个方向继续滑动，得到完整输出

完整扫描汇总

步幅 $s = 2$

每行 3 个合法起点



输出位置

3 × 3

3 行 × 3 列 = 9 个输出位置

结论

一处滑窗对应一个输出位置；增大步幅会减少合法起点

示意依据：CS231n: Spatial Arrangement 与 Convolution Demo | 沿用旧课件的 7 × 7 滑窗例子

1 Padding: 改变边界和输出尺寸

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

7×7 输入

- 7×7 输入，四周各补 1 个像素
- 3×3 尺寸卷积核，步幅为 1
- 补边后输入尺寸为 9×9
- 输出尺寸为 7×7

来源: CS231n: Convolutional Neural Networks, Zero-padding

1 输出尺寸由最后一个（合法）滑窗起点决定

$$H_{\text{out}} = \left\lfloor \frac{H + 2p_h - k_h}{s_h} \right\rfloor + 1$$

记号

H	补边前的输入高度
p_h	顶部、底部各补边的像素数
k_h	卷积核的高度
s_h	沿高度方向的步幅
H_{out}	输出特征图的高度

从滑窗起点推导

1. 补边后长度为 $H + 2p_h$
2. 起点最多到 $H + 2p_h - k_h$
3. 起点为 $0, s_h, 2s_h, \dots$, 所以取整后还要加 1

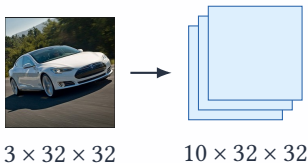
练习： $H = 8, k_h = 3, s_h = 2, p_h = 0$, 输出高度 H_{out} 是多少？

答案：3 | 不足一个完整窗口的部分不再产生输出

尺寸公式应用于 RGB 与灰度图像

$$H_{\text{out}} = \left\lfloor \frac{H + 2p_h - k_h}{s_h} \right\rfloor + 1$$

RGB 图像：设输入为 32×32

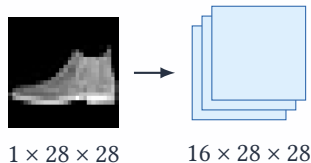


$$k = 5, s = 1, p = 2, C_{\text{out}} = 10$$

$$\lfloor (32 + 2 \times 2 - 5) / 1 \rfloor + 1 = 32$$

尺寸按 $C \times H \times W$ 标注

灰度图像：课程 CNN 第一层



$$k = 3, s = 1, p = 1, C_{\text{out}} = 16$$

$$\lfloor (28 + 2 \times 1 - 3) / 1 \rfloor + 1 = 28$$

来源：RGB 示意截取自 CS231n；灰度图为 Fashion-MNIST 测试 #0

1 从 MLP 到 CNN：参数量的显著减少

$$\text{参数数目 } P = C_{\text{out}}(C_{\text{in}}k_hk_w + 1)$$

$1 \times 28 \times 28 \Rightarrow 16 \times 28 \times 28$	参数量	参数约束
全连接	9 847 040	每条连接独立
局部连接，不共享	125 440	每个位置独立
3×3 卷积，共享	160	所有位置共用

结论

局部连接 + 参数共享显著减少参数数目，但每个位置仍需（局部）计算

1 参数少不（直接）等于计算量小

$$\text{FLOPs} \approx 2H_{\text{out}}W_{\text{out}} \times C_{\text{out}}C_{\text{in}} \times k_hk_w$$

本讲 CNN 第一层的线性变换

参数量 $16(1 \times 3 \times 3 + 1) = 160$

浮点运算次数 $\approx 2 \times 28 \times 28 \times 16 \times 9$
 $= 225\,792 \text{ FLOPs}$

FLOPs: Floating Point Operations, 浮点运算次数

本讲按一次乘法与一次累加计 2 FLOPs, 仅计卷积/线性层的主要运算

三种代价度量对比

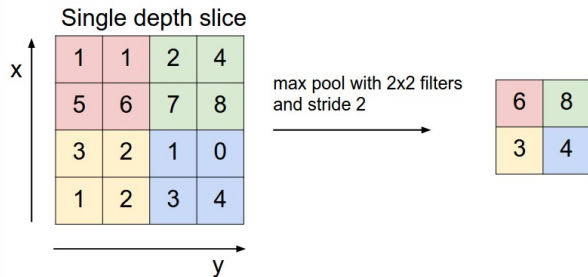
1. 参数量: (存储) 多少可学习数值
2. FLOPs: 执行多少浮点运算
3. 时间: + 硬件、访存与实现影响

1 卷积：（受约束的）线性变换

$$\mathbf{z} = A_K \mathbf{x} + \mathbf{b} \quad \longrightarrow \quad \mathbf{h} = \text{ReLU}(\mathbf{z})$$

- 卷积 A_K 是具有稀疏与共享结构的线性变换
- 和之前 MLP 类似，后接 ReLU，形成复合非线性函数

1 Max Pooling: 在邻域中保留最大响应



- 每个通道分别操作，不混合通道，**不含**可学习参数
- 此例中：高、宽尺寸各减半，每个窗口保留**最大**响应
- 同一窗口内的位置变化**不**改变最大值

注意

当响应**跨**窗口边界时，输出仍可能改变；不保证（完全）平移不变性

来源：CS231n: Convolutional Neural Networks, Pooling Layer

1 对比

全局平均池化：Global Average Pooling (GAP)

	Max Pooling	步幅卷积	GAP
主要操作	局部取最大	卷积并间隔采样	全图逐通道平均
可学习参数	无	有	无
空间输出	较小网格	较小网格	每通道一个值
主要取舍	丢失局部细节	同时改变提取与采样	丢失显式空间布局

$$\text{GAP}(A)_c = \frac{1}{HW} \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} A_{c,i,j}$$

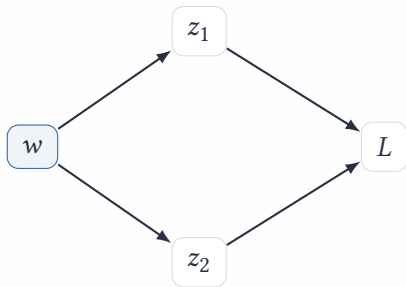
结论

GAP 常用于分类头 $\neq$ 中间层下采样

$$z_1 = wx_1$$

$$z_2 = wx_2$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial z_1} x_1 + \frac{\partial L}{\partial z_2} x_2$$



- 参数贡献：一个参数在计算图中有多个分支
- 反向传播把各分支贡献相加（见第4讲）

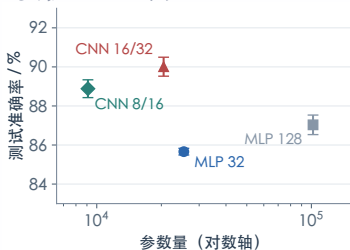


- **特征提取**：卷积利用局部连接与权重共享，ReLU 引入非线性，再下采样
- **分类**：Flatten 将特征排成向量，Linear 输出各类别的得分（和 MLP 一致）
- **训练**：损失 L 通过整个网络反向传播，更新卷积核与线性层参数

2

怎样验证卷积结构的作用

实验 CNN：两组 Conv-ReLU-Pool $\rightarrow$ Flatten $\rightarrow$ Linear | 通道 $a \rightarrow 2a$



结论

- CNN 16/32 用 20490 参数达到 90.01% 准确率，MLP 32 用 25450 达到 85.66%
- 但 CNN 的单图 FLOPs 更高：参数节省不代表计算或时间节省

Fashion-MNIST | 8 Epoch | 均值 $\pm$ 样本标准差

图像移动后，卷积响应会怎样变化？

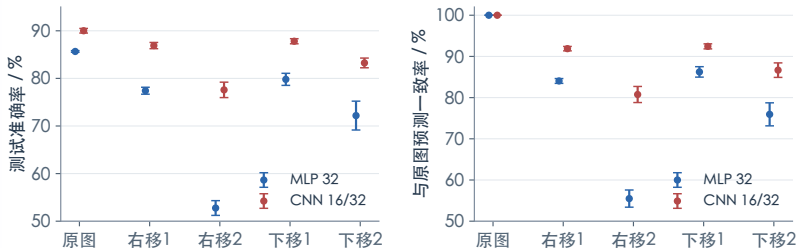
$$f(T_{\Delta}X) = T_{\Delta}f(X) \quad \text{等变性}$$



注意

- 步幅 1 时内部响应随输入移动；补边与裁切限制边界上的等式
- 注意：等变性 $\neq$ 分类结果不变（后者要求 $g(T_{\Delta}X) = g(X)$ ）

卷积响应跟着移动，预测类别就不变吗？



结论

- 右移 2 像素：CNN 90.01% $\rightarrow$ 77.59%，MLP 85.66% $\rightarrow$ 52.77%
- 局部等变性不（直接）保证模型的预测和性能不变

2 为什么平移后可能预测不一致？

池化窗口的分组可能改变

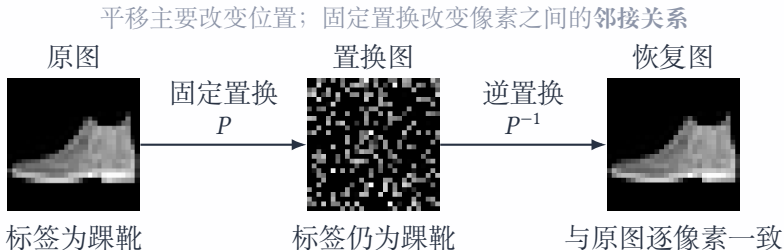
- 考虑窗口大小 2，步幅 2
- 原响应 $[0, 3 \mid 1, 0] \rightarrow [3, 1]$
- 向右移动一格
 $[0, 0 \mid 3, 1] \rightarrow [0, 3]$

竖线表示池化窗口边界

其他可能原因：

- **下采样**：移动一格可能改变窗口分组或采样位置
- **边界**：补零与裁切可能改变边缘信息和卷积响应
- **分类头**：Flatten 后，响应的位置对应不同的线性权重

打乱像素位置后，CNN 的优势还在吗？



- **不变量**：784 个像素值与标签不变，已知置换可完全恢复原图
- **变量**：原来邻近的像素被分散，局部边缘与轮廓关系被打乱

思考

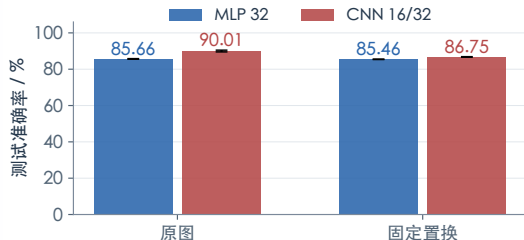
如果 CNN 的优势依赖（原始）局部结构，置换后会有什么结果？

原图	固定置换 P
MLP 在原图上训练与评价	在置换图上重新训练与评价
CNN 在原图上训练与评价	在置换图上重新训练与评价

- **固定映射**：训练、验证、测试与两个模型共用**同一置换**
- **控制设置**：标签、划分、训练 Epoch 数和评价方法保持一致

注意

- 只置换测试图片：分布变化；
- 每张图随机置换：完全另一个任务



- **MLP**: 85.66% → 85.46%, 变化较小
- **CNN**: 90.01% → 86.75%, 下降 3.26 个百分点
- **相对优势**: CNN 与 MLP 的差距缩小, 但未消失

结论

- CNN 和 MLP 的准确率优势: 原图 +4.35%, 置换后 +1.29%
- 结论: 支持关系有助于该 CNN 性能优势来自局部空间特征

MLP: 全连接读取全部像素

- **连接方式:** 首层每个单元仍能读取全部输入
- **函数表示:** 重加权重的输入连接, 即可表示原函数
- **训练结果:** 近似不变 (独立训练有波动)

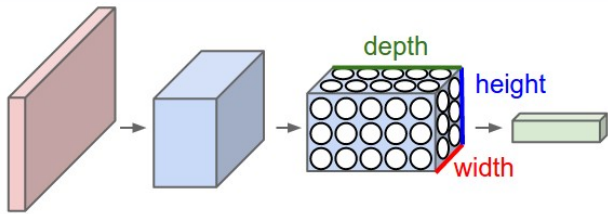
CNN: 局部连接依赖新的邻接关系

- **连接方式:** 小卷积核读取重排后的相邻格点
- **结构匹配:** 置换后格点通常来自原图分散位置, 原有局部模式被打乱
- **训练结果:** 仍学习“特征”用于分类, 但准确率下降更多

3

CNN 怎样形成层次表示

3 LeNet 的基本思路：特征提取、下采样与分类



来源：CS231n: Convolutional Neural Networks, Architecture Overview / Case Studies

- 局部特征提取
- 空间下采样
- 通过深度形成更高层的特征组合
- 分类输出

3 例：CNN 尺寸参数的逐层核对

层	输出 $C \times H \times W$	参数量
输入	$1 \times 28 \times 28$	0
Conv 3×3 , $p = 1 + \text{ReLU}$	$16 \times 28 \times 28$	160
Max Pooling 2×2 , $s = 2$	$16 \times 14 \times 14$	0
Conv 3×3 , $p = 1 + \text{ReLU}$	$32 \times 14 \times 14$	4 640
Max Pooling 2×2 , $s = 2$	$32 \times 7 \times 7$	0
Flatten + Linear	10 个类别得分	15 690
合计		20 490

结论

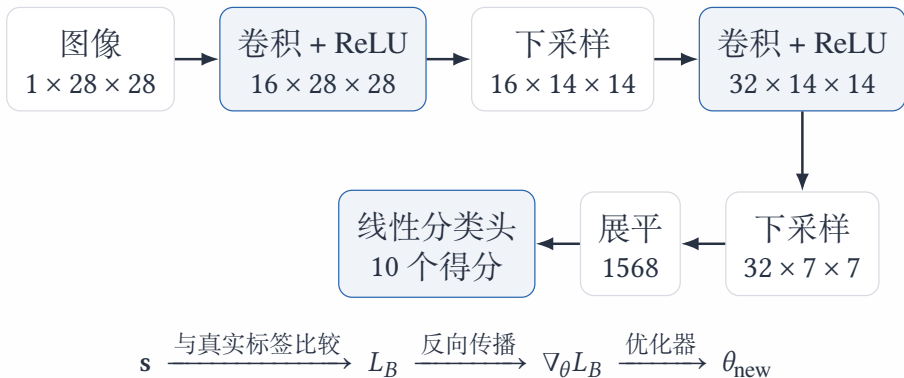
本例约 3/4 的参数来自于全连接分类头

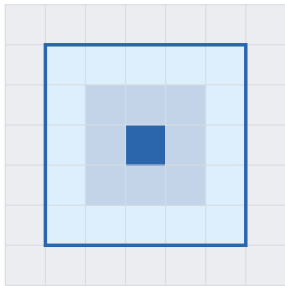
```
model = nn.Sequential(  
    nn.Conv2d(1, 16, 3, padding=1),  
    nn.ReLU(),  
    nn.MaxPool2d(2),  
    nn.Conv2d(16, 32, 3, padding=1),  
    nn.ReLU(),  
    nn.MaxPool2d(2),  
    nn.Flatten(),  
    nn.Linear(32 * 7 * 7, 10)  
)  
logits = model(x) # x: [B, 1, 28, 28]
```

来源: PyTorch Conv2d 与 CrossEntropyLoss

- 默认 Conv2d 步幅为 1
- MaxPool2d(2) 默认步幅为 2
- Flatten 保留批量维度
- 输出尺寸为 $B \times 10$

3 模型训练闭环流程保持不变





一个输出单元
追溯到输入

定义 | 理论感受野

在网络连接结构上，能够影响指定单元的输入区域

- 一个 3×3 卷积: 3×3
- 连续两个: 5×5
- 连续三个: 7×7

注：此例各层步幅均为 1，无空洞卷积，仅讨论不受边界裁切影响的内部单元

$$r_l = r_{l-1} + (k_l - 1)j_{l-1} \quad j_l = s_l j_{l-1} \quad r_0 = j_0 = 1$$

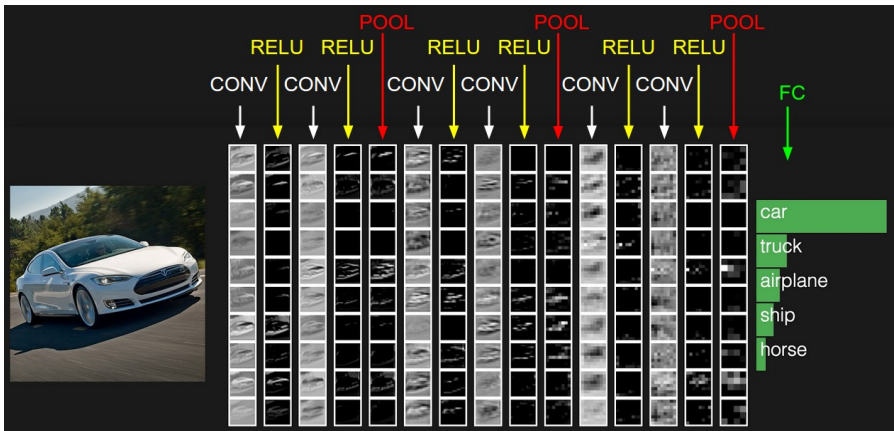
操作	k_l	s_l	间隔 j_l	感受野 r_l
Conv1	3	1	1	3
Pool1	2	2	2	4
Conv2	3	1	2	8
Pool2	2	2	4	10

结论

下采样增加后续单元在原图上的采样间隔，因此后续卷积能更快扩大感受野

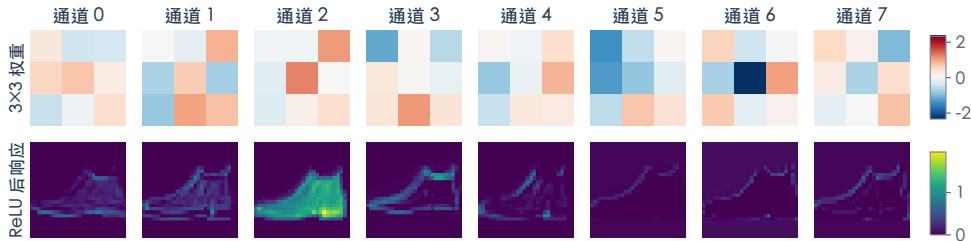
r_l 表示一维边长 | j_l 表示相邻输出中心在原图上的间隔

通过深度把局部响应逐步/层组合



CS231n 的小型 VGG 示例 | 每列展示一层的部分通道，右侧是类别得分

来源：CS231n: Convolutional Neural Networks, Layers Used to Build ConvNets



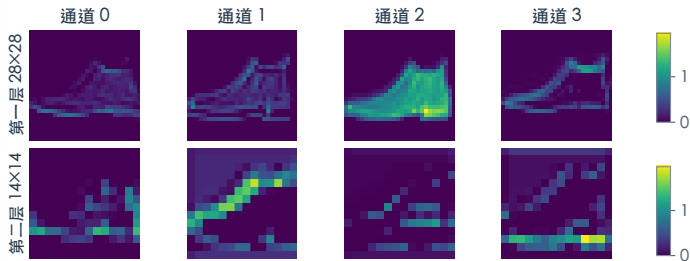
卷积核

局部规则如何对输入加权计算？

特征图

规则在当前图像哪些位置响应？

同一张图像在不同层有不同的表示



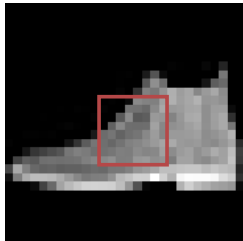
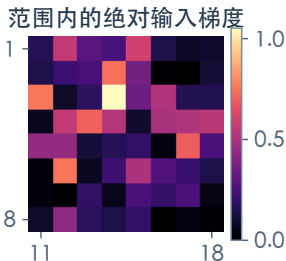
每层内部共用色标，两层色标不同

结论

第二层感受到更大邻域，但不同层和通道没有一一对应关系

理论感受野与有效影响范围并不相同

理论范围 8×8



- 理论感受野由网络连接决定
- 实际/有效感受野：取决于影响强度分布
- 权重、ReLU 和池化决定活跃路径

第二层预激活 $z_{0,7,7}$

$|\partial z_{0,7,7} / \partial X|$ | 框外梯度为 0

来源：Luo 等：Understanding the Effective Receptive Field, 2016

注：右图为单样本的局部梯度敏感度

从中间单元到类别得分



$$s_k = b_k + \sum_{c=0}^{31} \sum_{i=0}^6 \sum_{j=0}^6 W_{k,c,i,j} A_{c,i,j}$$

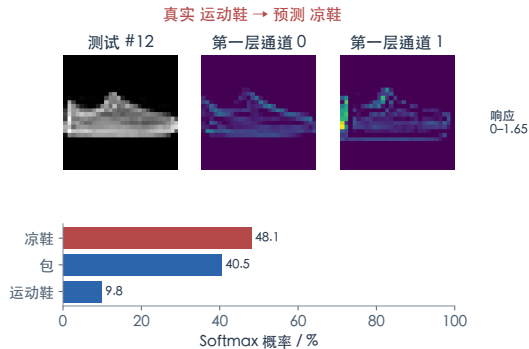
k 为类别, c 为通道, i, j 为空间位置; W 是按通道与位置展开记写的分类头权重

- **中间单元**: 前页 Conv2 单元在结构上只连接到局部 8×8 区域
- **类别得分**: 汇总所有通道和位置, 输入连接范围可覆盖整幅图像
- **实际影响**: 连接范围 $\neq$ 影响强度, 还取决于参数与当前图像的相关作用

思考

改变图像中的一个区域, 会怎样影响某一类别的预测概率?

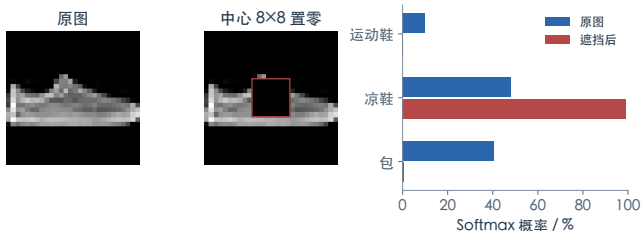
3 局部区域对分类预测的影响



- **观察：**运动鞋被判为凉鞋，真实类别概率为 9.8%
- **假设：**图片中心区域含有支持“运动鞋”判断的信息
- **检验：**固定模型参数，将中心 8×8 像素置零

思考

若假设成立，分类结果为“运动鞋”的概率应怎样变化？

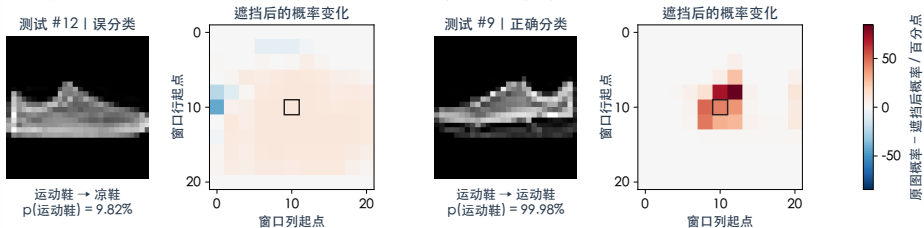


- 结果：“运动鞋”概率由 9.8% 变为 0.009%，最终类别仍为“凉鞋”
- 解释：与假设预期一致，模型对这次中心遮挡敏感
- 继续检验：更换遮挡位置和样本，检查结果是否稳定

3 从一次遮挡到遮挡敏感图

“运动鞋”分类的概率提升： $\Delta p_k(r, c) = p_k(X) - p_k(X^{(r, c)})$

$X^{(r, c)}$: 从行 r 、列 c 起将 8×8 区域置零；固定 k 为“运动鞋”



- 读图：红色表示遮挡后概率下降，蓝色表示上升；两图使用同一色标

每格对应一个窗口起点，黑框是中心遮挡 | 色标单位为百分点

3 大卷积核 VS 堆叠小卷积核？

一层 5×5

$$X \xrightarrow{5 \times 5} Z$$

- 感受野： 5×5
- 权重数： $25C^2$

两层 3×3

$$X \xrightarrow{3 \times 3} \text{ReLU} \xrightarrow{3 \times 3} Z$$

- 感受野： 5×5
- 权重数： $2 \times 9C^2 = 18C^2$

输入、中间层、输出均为 C 通道 | 步幅 1、无空洞、忽略偏置 VGG | 采用小核堆叠

- 组合：中间加入 ReLU，提供额外的非线性
- 取舍：需要保存中间激活；权重少不直接保证运行更快

来源：CS231n: Convolutional Neural Networks, Layer Patterns, 小卷积核堆叠

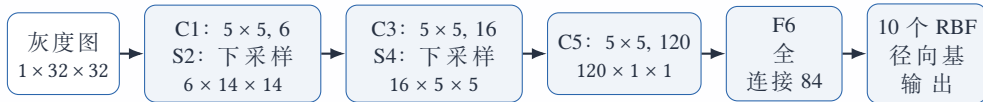
3 深度 CNN 挖掘复杂特征，也增加训练难度

网络	代表性设计	解决的问题
LeNet	局部提取、下采样、分类	怎样形成完整分类流程？
AlexNet / VGG	ReLU、规模化训练、 深层组合	加深网络带来什么变化？
ResNet	用 残差连接 改善深层优化	怎样让深层网络更易训练？

结论

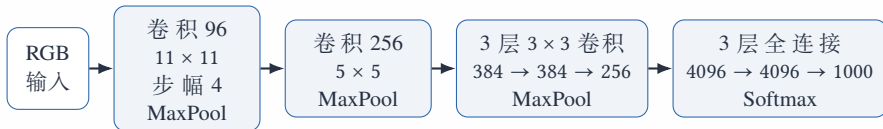
增加深度允许挖掘更复杂的特征，但表达能力增加**并不保证**有限（训练）预算下模型性能更好

LeNet-5 (1998): 卷积与下采样交替，完成 10 类手写数字识别



C: 卷积 | 逗号后为输出通道数 | LeNet 使用缩放 tanh 与可学习下采样, C3 仅连接部分通道

AlexNet (2012): 5 层卷积 + 3 层全连接，扩展到 1000 类自然图像



省略激活、归一化与双 GPU 分组 | ReLU、GPU 与 Dropout 支持更大模型

设计动机：类别和外观变化增加，需要容量更大的层次表示

结构依据：LeCun 等，1998，图 2 与第 II-B 节，Krizhevsky 等，2012，图 2 与第 3.5 节

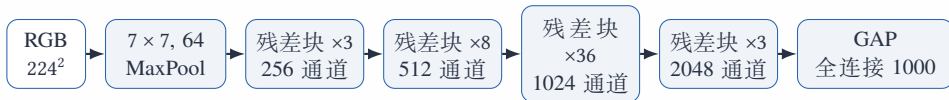
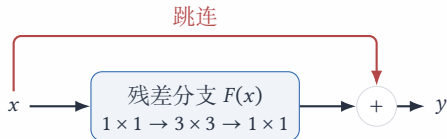
从 VGG 到 ResNet：由堆叠卷积到堆叠残差块

VGG-16 (2014): 统一使用 3×3 卷积, 13 层卷积 + 3 层全连接



64×2 : 64 通道、连续 2 层 | 小核堆叠在扩大感受野的同时增加非线性组合

ResNet-152 (2015): 将 50 个三层残差块串联



- 152 层的计数: $1 + 50 \times 3 + 1$

- 分类头: GAP + 全连接

省略激活与归一化, 分类后接 Softmax | 同尺寸跳连示意, 尺寸改变时需匹配分支

结构依据: Simonyan 与 Zisserman, 表 1, 配置 D (VGG-16), He 等, 表 1 (ResNet-152) 与图 5 (三层瓶颈块)

3 1×1 卷积怎样混合通道？

$$\underbrace{\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \end{bmatrix}}_{2 \text{ 个卷积核}} \underbrace{\begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix}}_{\text{同一位置的 3 个通道}} = \begin{bmatrix} 5 \\ -2 \end{bmatrix}$$

每个核覆盖 3 个输入通道，空间大小为 1×1

- 各输入通道加权求和
- 每个位置使用相同权重

对应 ResNet 的瓶颈残差分支

$$256 \xrightarrow{1 \times 1} 64 \xrightarrow{3 \times 3} 64 \xrightarrow{1 \times 1} 256$$

数字为通道数：先压缩，在较少通道上做空间卷积，再恢复通道数

来源：CS231n: 1×1 Convolution; He 等: Deep Residual Learning, 图 5

3 更深的结构带来了怎样的收益？

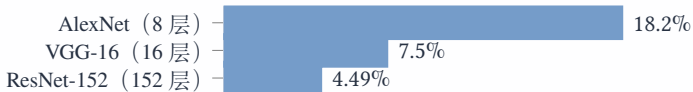
MNIST | 10 类数字

LeNet-5 测试错误率

0.95%

ImageNet | 1000 类自然图像

单模型 Top-5 验证错误率（越低越好）



Top-5 错误：真实标签不在得分最高的 5 类中

- 结论：模型结构 + 训练方法，逐步提升任务性能

思考

增加层数时，能否先保留浅网络的功能，再学习更好的表示？

来源：LeCun 等，III-B, 0.95% | Krizhevsky 等，表 2，单 CNN，18.2%，10 裁剪
Simonyan 与 Zisserman，表 4，配置 D，7.5%，多尺度稠密评测 | He 等，表 4，4.49%，多尺度全卷积评测

3 深度网络难训练：增加层数时，能否保留原（浅）网络的功能？

假设新增层能够表示恒等映射

- 浅网络先提取表示 $h(X)$ ，再由分类头 g 输出得分
- 在二者之间增加一个保持维度的（恒等）映射 G ，使 $G(h) = h$ ，则有

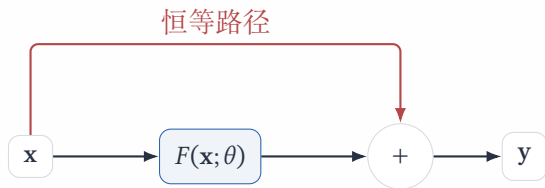
$$g(G(h(X))) = g(h(X))$$

- **表示能力**：存在一组参数，使更深网络保留原函数
- **优化过程**：梯度训练未必容易找到这组参数

结论

能否直接在结构中提供一条保留输入的路径？Residual neural network (ResNet)

来源：CS231n: Convolutional Neural Networks, Case Studies, ResNet



定义 | 残差连接

$$y = x + F(x; \theta)$$

学习目标映射相对输入的改变量

- $F(x) = 0$ 时保留输入

$$J_F = \frac{\partial F}{\partial \mathbf{x}} \quad \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = I + J_F$$

$$\nabla_{\mathbf{x}} L = (I + J_F)^\top \nabla_{\mathbf{y}} L = \nabla_{\mathbf{y}} L + J_F^\top \nabla_{\mathbf{y}} L$$

- 第一项：沿恒等路径直接传回，梯度优化更容易
- 第二项：经过残差分支的雅可比矩阵
- 与第四讲相同，分支梯度在输入节点相加

3 残差相加要求两条分支的尺寸相同

方案一：尺寸不变

$$32 \times 14 \times 14 : \mathbf{y} = \mathbf{x} + F(\mathbf{x})$$

两条分支都有完全相同的通道与空间尺寸

方案二：通道增加、空间下采样

$$32 \times 14 \times 14 \rightarrow 64 \times 7 \times 7 : \mathbf{y} = P(\mathbf{x}) + F(\mathbf{x})$$

可用 1×1 、步幅 2 的投影卷积

结论

1×1 卷积可以混合通道；步幅决定是否同时改变空间尺寸

注：投影路径不是恒等映射，其梯度应写作 $J_P^T \nabla_{\mathbf{y}} L$

接下来：已有表示能否用于新任务？

获得训练良好的特征提取器后，可以尝试复用处理新的分类问题



AlexNet 第一层学到的 96 个 $11 \times 11 \times 3$ 核

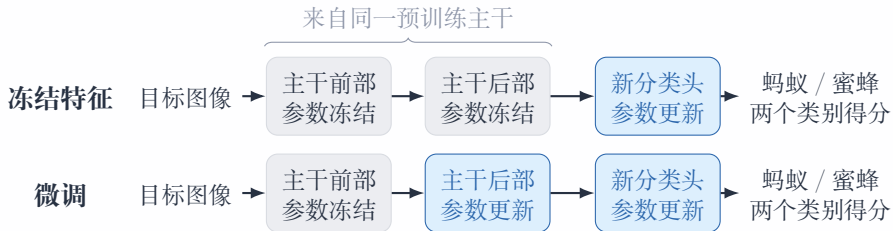
来源：CS231n: Convolutional Neural Networks, Parameter Sharing, 预训练卷积核示例

来源：PyTorch: Transfer Learning for Computer Vision Tutorial

例子：蚂蚁与蜜蜂分类

1. 在（大量）自然图像上预训练的 CNN
 2. 用目标任务的训练数据学习二分类输出层
- 通用局部特征可复用

例子：同一预训练 CNN，换上蚂蚁 / 蜜蜂二分类头



灰色表示参数不变，蓝色表示参与更新 | 下行示例仅解冻后部，也可微调整个主干

	冻结特征	微调
训练代价	通常较小	通常较大
适用起点	数据较少、任务较接近	有足够数据或需要适应新分布

来源：PyTorch: Transfer Learning for Computer Vision Tutorial，图示按蚂蚁 / 蜜蜂任务改编

1. 局部连接与权重共享分别约束了什么？
2. 输出尺寸、参数量和 FLOPs 为什么不是同一个概念？
3. 感受野如何随卷积与下采样增长？
4. 恒等捷径为何有助于模型训练的优化？

第三至第五讲

$X \rightarrow \text{表示} \rightarrow \hat{y}$

- 从图像提取用于类别预测的信息
- 分类头可以压缩、丢弃数据中与标签无关的细节

第六讲

噪声 $\rightarrow$ 逐步去噪 $\rightarrow X$

- 学习完整图像分布并生成新样本

思考

如果不要输出类别，而是一张新图像，网络的输出和训练目标该如何设计？

附录导览

- **A | 计算补充** 输出尺寸、参数量、计算代价与共享梯度
- **B | 空间性质与感受野** 固定置换、平移条件与尺寸推广
- **C | 结构与训练补充** 参数效率、残差梯度与迁移学习
- **D | 实验说明** 训练代价、输入变换与遮挡扫描
- **参考资料** 经典网络、特征可视化、数据与实现

附录 A

计算补充

输出尺寸、参数量、计算代价与共享梯度

A 输出尺寸与参数量练习答案

设置	输出空间尺寸	输出通道	参数量
$7 \times 7, k = 3, s = 1, p = 0$	5×5	1	10
$7 \times 7, k = 3, s = 2, p = 0$	3×3	1	10
$7 \times 7, k = 3, s = 1, p = 1$	7×7	1	10
$8 \times 8, k = 3, s = 2, p = 0$	3×3	1	10
RGB $32 \times 32, k = 5, s = 1, p = 2$	32×32	10	760

前四行输入为单通道，均计算偏置 | 第五行 $10(3 \times 5 \times 5 + 1) = 760$

结论

相同卷积核应用于不同大小的图像时，参数量不变，但输出元素数量改变

A 候选结构的静态代价核算

模型结构	参数量	单图 FLOPs
MLP: $784 \rightarrow 32 \rightarrow 10$	25 450	50 816
MLP: $784 \rightarrow 128 \rightarrow 10$	101 770	203 264
CNN: 通道 $8 \rightarrow 16$ + 展平分类	9 098	580 160
CNN: 通道 $16 \rightarrow 32$ + 展平分类	20 490	2 063 488
CNN: 通道 $16 \rightarrow 32$ + GAP 分类	5 130	2 032 768

注

静态结构计算，每次乘加按 2 FLOPs 估计
仅计卷积与线性层，不含池化、GAP、激活和偏置加法

GAP 变体仅作静态核算，本讲没有训练该变体或报告其准确率

A 卷积核梯度按使用位置汇总

L 为单样本损失, Z 为卷积预激活, K 为核权重, b 为偏置

$$\delta_{o,i,j} = \frac{\partial L}{\partial Z_{o,i,j}}$$

$$\frac{\partial L}{\partial K_{o,c,u,v}} = \sum_{i,j} \delta_{o,i,j} \tilde{X}_{c, i s_h + u, j s_w + v} \quad \frac{\partial L}{\partial b_o} = \sum_{i,j} \delta_{o,i,j}$$

o, c 为输出 / 输入通道, i, j 为输出位置, u, v 为核内位置

$\tilde{X}$ 为补边后的输入, s_h, s_w 为高 / 宽步幅 | 索引从 0 开始, 空洞率为 1

- 求和来自同一个参数被多个空间位置复用
- 对 mini-batch 平均损失 L_B , 再对样本贡献取平均
- Max Pooling 的梯度传给被选中的最大值位置
- ReLU 在正预激活位置传递梯度, 在负位置阻断

最大值并列与 ReLU 在零点的处理由实现约定决定 | 本讲不要求手写完整卷积反向传播

附录 B

空间性质与感受野

固定置换、平移条件与尺寸推广

B MLP 的函数族可以吸收固定置换

$\mathbf{x} \in \mathbb{R}^d$ 为列向量, $P \in \mathbb{R}^{d \times d}$ 为置换矩阵, $\mathbf{W}_1 \in \mathbb{R}^{d \times h}$

$$\mathbf{x}' = P\mathbf{x} \quad P^\top P = I \quad \mathbf{W}'_1 = P\mathbf{W}_1$$

$$\mathbf{W}'_1{}^\top \mathbf{x}' + \mathbf{b}_1 = \mathbf{W}_1{}^\top P^\top P\mathbf{x} + \mathbf{b}_1 = \mathbf{W}_1{}^\top \mathbf{x} + \mathbf{b}_1$$

- 同时重排第一层权重的输入维度, 即可表示原来的函数
- 固定置换不删除像素信息, 也不缩小普通 MLP 的函数族
- CNN 的局部邻域在置换后通常不再对应原来的图像邻域

注意

函数族等价不保证有限预算下独立训练的数值结果逐次相同

B 平移性质的适用条件

操作或条件	对平移性质的影响
步幅 1 的共享卷积	在无限域或边界兼容时具有平移等变性
有限图像上的零补边	边界引入额外条件，内部与边界应分开检查
步幅 $s > 1$	一像素移动未必对应整数个输出位置移动
Max Pooling	只可能提供有限的局部稳定性
GAP	对特征位置重排不变，但前层和边界仍可能改变特征值
展平后的全连接分类头	可使用绝对空间位置，因此不保证平移不变

标准卷积的这些性质不能直接推广为旋转、缩放或形变不变性

B 感受野与空洞卷积的尺寸推广

H 为输入高度, p 为每侧补边, s 为步幅, k 为核边长, d 为核内采样间距

$$k_{\text{eff}} = d(k - 1) + 1$$

$$H_{\text{out}} = \left\lfloor \frac{H + 2p - d(k - 1) - 1}{s} \right\rfloor + 1$$

$$r_l = r_{l-1} + d_l(k_l - 1)j_{l-1} \quad j_l = s_l j_{l-1}$$

r_l 为第 l 层理论感受野边长, j_l 为相邻单元在输入上的间距, 初值 $r_0 = j_0 = 1$

- 正文统一取 $d = 1$, 上式按层使用 k_l, d_l, s_l
- $k = 3, d = 2$ 的有效核边长为 5, 参数槽位仍是 3×3
- 有效跨度扩大不意味着区域内每个位置都被同样直接采样

同一公式分别应用于高和宽 | 理论感受野仍不等同于实测影响强度

附录 C

结构与训练补充

参数效率、残差梯度与迁移学习

C 深度可分离卷积的参数效率

$$\underbrace{k^2 C_{\text{in}} C_{\text{out}}}_{\text{普通卷积}} \longrightarrow \underbrace{k^2 C_{\text{in}}}_{\text{逐通道空间卷积}} + \underbrace{C_{\text{in}} C_{\text{out}}}_{1 \times 1 \text{ 通道混合}}$$

例子: $k = 3$, $C_{\text{in}} = 64$, $C_{\text{out}} = 128$

普通卷积: $9 \times 64 \times 128 = 73\,728$

深度可分离卷积: $9 \times 64 + 64 \times 128 = 8\,768$

结论

参数与计算节省来自更强的结构分解约束, 表达能力也随之改变

每个输入通道仅一个空间核, 均不计偏置 | 实际运行加速仍依赖实现与设备

来源: Howard 等: MobileNets, 2017, 深度可分离卷积

C 相加后再接 ReLU 时的残差梯度

$$\mathbf{u} = \mathbf{x} + F(\mathbf{x}) \quad \mathbf{y} = \text{ReLU}(\mathbf{u})$$

$$D = \text{diag}(\mathbf{1}_{u_i > 0}) \quad \nabla_{\mathbf{x}} L = (I + J_F)^\top D \nabla_{\mathbf{y}} L$$

$J_F = \partial F / \partial \mathbf{x}$ 为雅可比矩阵, D 为 ReLU 门控矩阵, I 为恒等矩阵

- 正文推导聚焦于加法节点, 因此暂时省略相加后的激活
- 若后面存在 ReLU, 梯度仍会经过对应门控
- 若捷径是投影 P , I 应替换为 J_P

结论

残差连接提供额外梯度路径, 不意味着网络中所有路径都畅通

C 冻结参数与固定特征还差一步

- **参数更新**: 冻结层设置 `requires_grad=False`, 优化器只接收需要更新的参数
- **运行状态**: Batch Normalization (批归一化) 的运行均值和方差不属于梯度更新的参数, 训练模式下仍可能改变
- **固定特征**: 冻结主干使用 `eval()`, 分类头使用训练模式, 沿用预训练模型的输入预处理
- **逐步微调**: 解冻选定层并加入优化器, 显式决定归一化统计量是否更新, 再用验证集选择方案

两个设置分别控制什么?

`requires_grad` 控制参数的梯度记录, `train()/eval()` 控制层的运行行为

这里假设 Batch Normalization 保存运行统计量 | 单独调用 `eval()` 不会冻结参数

来源: PyTorch: Autograd mechanics, 冻结参数与训练 / 推理模式

附录 D

实验说明

训练代价、输入变换与遮挡扫描

D 本讲实验的设置与实际代价

- **划分**：训练 / 验证 / 测试为 50,000 / 10,000 / 10,000，种子 2026，像素除以 255
- **训练**：8 Epoch，batch 512，Adam，学习率 0.003，无数据增强、无权重衰减
- **选择**：按验证交叉熵保存参数，种子 11、23、47，均值 $\pm$ 样本标准差
- **置换**：种子 52026，三种划分和所有模型共用同一索引

原图模型	测试准确率 / %	训练及验证时间 / s
MLP 32	85.66 ± 0.17	0.442 ± 0.061
MLP 128	87.03 ± 0.50	0.614 ± 0.004
CNN 8/16	88.88 ± 0.46	117.8 ± 35.3
CNN 16/32	90.01 ± 0.49	188.8 ± 12.7

本机 arm64 CPU | NumPy + Accelerate | 含训练与验证，不含读盘、测试与绘图

运行时间受实现与后台负载影响，不能直接推断 PyTorch / GPU 速度

D 输入变换实验的设置与解释

对照	改变什么	保持什么
小幅平移	向右 / 下各移动 1、2 像素 零填充，超出边界的像素裁切	固定原图训练的模型 报告准确率和预测一致率
固定置换	重排全部像素，破坏原二维邻接 原图 / 置换图四组分别训练	MLP32、CNN16/32 共用置换 三种划分和标签对应关系固定
遮挡扫描	8×8 窗口置零，步幅 2 每图 $11 \times 11 = 121$ 个位置	固定 CNN16/32、种子 23 测试样本 #12、#9，目标为运动鞋

$$\Delta p = 100[p_{\text{运动鞋}}(X) - p_{\text{运动鞋}}(X_{\text{遮挡}})] \quad \text{单位：百分点}$$

- **样本选择**：首个误判样本与同类别首个正确样本，按测试索引确定
- **读图边界**：窗口左上角从 0 到 20，间隔 2，无越界窗口，共用对称色标

每格表示一个遮挡窗口，不是单像素贡献 | 概率差说明干预敏感性，不能单独确定原误判原因

经典网络与特征可视化的原始来源

来源与参考文献

研究论文

- [1] **LeNet** | LeCun 等, 1998
Gradient-Based Learning Applied to Document Recognition
- [2] **AlexNet** | Krizhevsky、Sutskever、Hinton, 2012
ImageNet Classification with Deep Convolutional Neural Networks
- [3] **VGG** | Simonyan 与 Zisserman, 2014
Very Deep Convolutional Networks for Large-Scale Image Recognition
- [4] **ResNet** | He 等, 2016
Deep Residual Learning for Image Recognition
- [5] **特征可视化** | Zeiler 与 Fergus, 2014
Visualizing and Understanding Convolutional Networks

来源：教学配图：CS231n Convolutional Neural Networks

数据、机制与实现参考

来源与参考文献

数据与机制

- [6] 数据 | Fashion-MNIST 官方仓库
Xiao、Rasul、Vollgraf, 2017
- [7] 感受野
Understanding the Effective Receptive Field in Deep
Convolutional Neural Networks
Luo、Li、Urtasun、Zemel, 2016
- [8] 深度可分离卷积
MobileNets
Howard 等, 2017

实现参考

- [9] 卷积 API | PyTorch Conv2d
参数含义、互相关定义与输出尺寸
- [10] 分类损失 | PyTorch CrossEntropyLoss
类别得分与标签的接口约定
- [11] 迁移学习
PyTorch Transfer Learning for Computer Vision
Tutorial
自然图像分类、冻结特征与微调
- [12] 冻结与运行状态
PyTorch Autograd mechanics 与 BatchNorm2d
参数梯度与运行统计量分别控制