



卷积神经网络基础



深度学习理论基础

- 卷积神经网络的发展
- 卷积
- 感受野机制
- 卷积操作
- 池化操作
- 完整卷积神经网络
- 卷积神经网络优化与训练
- 案例

卷积神经网络的发展

1943-1986

起源于早期探索

1943年，麦卡洛克和皮茨提出神经元模型，为神经网络奠定基础。

1980年，福岛邦彦提出基于感受野的神经网络模型（如神经认知机），模拟视觉皮层机制

1986-2006

初步发展

1986年，多层感知机 (MLP) 的诞生引入了非线性激活函数和反向传播 (BP) 算法，使神经网络能够解决非线性问题。

2006年，辛顿和鲁斯兰提出深度学习概念及逐层训练算法，开启了深度学习时代。

2006-2012

快速发展阶段

2006年后，深度学习在图像识别、语音识别等领域取得突破，CNN和RNN相继被提出并广泛应用。

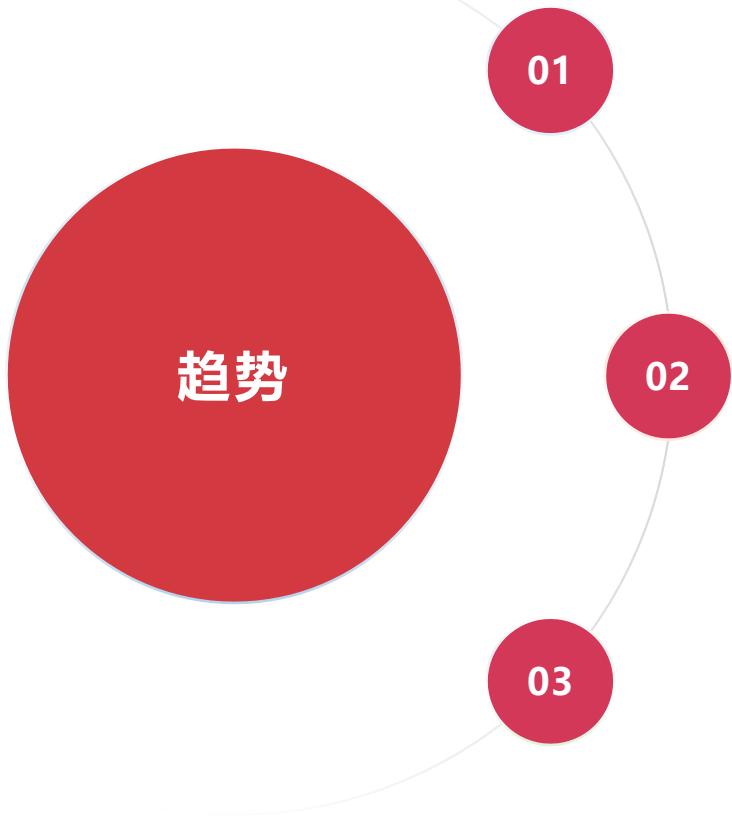
2012-至今

广泛应用阶段

2012年，AlexNet在ImageNet竞赛中取得优异成绩，推动了深度学习在图像识别领域的广泛应用。

近年来，CNN在图像分类、目标检测、语义分割等领域取得了显著进展，准确率不断提高。

发展趋势



网络结构优化

ResNet、DenseNet等网络结构的出现，解决了深层网络训练中的梯度消失和梯度爆炸问题。

- 轻量化与高效化：
 - 目标：在边缘设备（如 Orange Pi Alpro）实现低功耗推理
 - 技术：知识蒸馏、模型压缩（如剪枝、量化）、参数共享

轻量级网络结构

MobileNet、ShuffleNet等轻量级网络结构适应了移动和嵌入式设备的需求。

注意力机制深化：从通道注意力（SENet）扩展到空间注意力、自注意力（如 Vision Transformer 与 CNN 结合）

边缘计算与实时推理

EfficientNet通过同时优化网络的深度、宽度和分辨率，显著提升了模型的性能和效率。

- 昇腾 CANN 与边缘设备：
 - 通过 ATC 工具将模型转换为.om 格式，在 Orange Pi Alpro（昇腾 310B）实现高效推理（如 GoogLeNet 图像分类、ResNet 性别分类案例）
 - 优势：低延迟、本地化部署，适用于无网络或高实时性场景

卷积

卷积概念

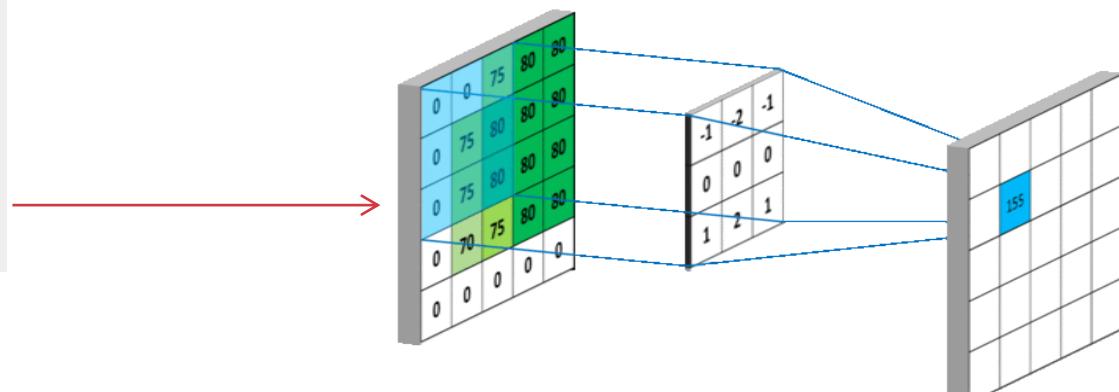
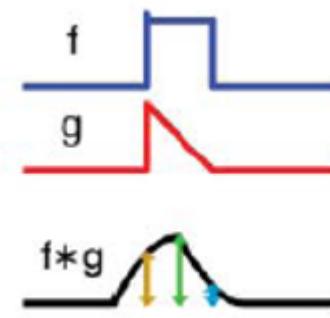
卷积(Convolution)是通过两个函数 f 和 g 生成第三个函数的一种数学运算，其本质是一种特殊的积分变换，表征函数 f 与 g 经过翻转和平移的重叠部分函数值乘积对重叠长度的积分

卷积操作是指将一个可移动的小窗口与图像进行逐元素相乘然后相加的操作
(这个小窗口其实是一组固定的权重，它可以被看作是卷积核)

这个操作的名称“卷积”，源自于这种元素级相乘和求和的过程

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau$$

$$\begin{aligned}y[n] &= f[n] * g[n] \\&= \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m] \\&= \sum_{m=-\infty}^{\infty} g[m] \cdot f[n-m]\end{aligned}$$

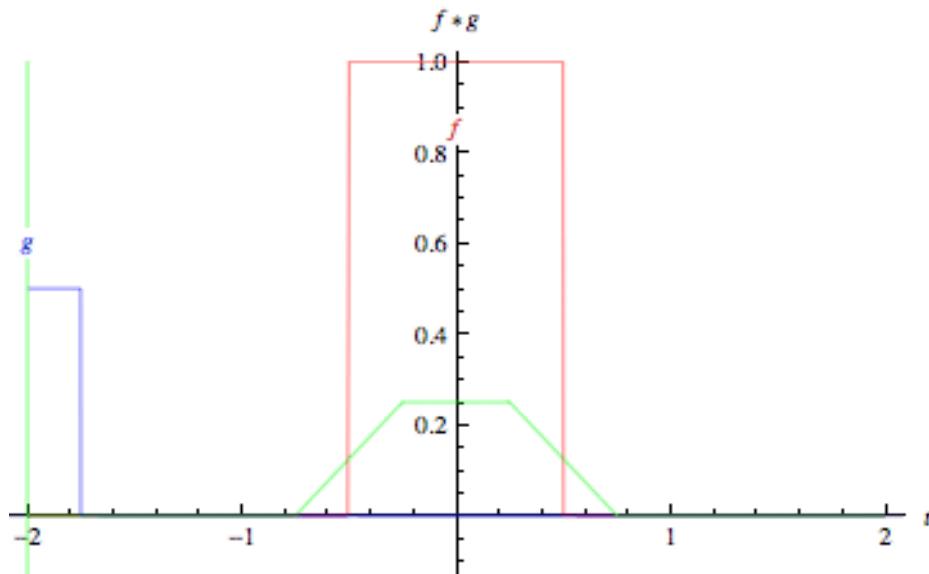
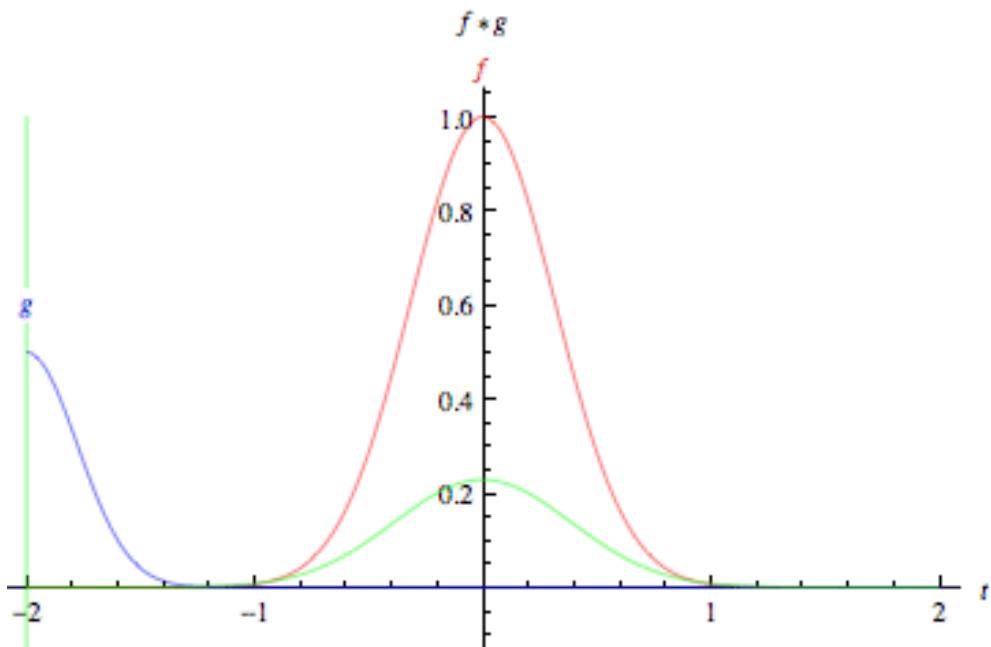


输入图像

卷积核

特征图

卷积演示



卷积计算

无填充步长=1

1	2	3
4	5	6
7	8	9



0	1
2	3



25	

$$0 \times 1 + 1 \times 2 + 2 \times 4 + 3 \times 5 = 25$$

1	2	3
4	5	6
7	8	9



0	1
2	3



25	31

$$0 \times 2 + 1 \times 3 + 2 \times 5 + 3 \times 6 = 31$$

1	2	3
4	5	6
7	8	9



0	1
2	3



25	31
43	

$$0 \times 4 + 1 \times 5 + 2 \times 7 + 3 \times 8 = 43$$

1	2	3
4	5	6
7	8	9



0	1
2	3



25	31
43	49

$$0 \times 5 + 1 \times 6 + 2 \times 8 + 3 \times 9 = 49$$



$$b[i, j] = \sum_{u,v} a[i + u, j + v] w[u, v]$$

a: 输入图片, b : 特征图, w : 卷积核参数

填充 Padding

填充是指在边缘像素点周围填充“0”（即0填充），使得输入图像的边缘像素也可以参与卷积计算。数据填充的主要目的是确保卷积核能够覆盖输入图像的边缘区域，同时保持输出特征图的大小。

0	0	0	0	0
0	7	8	9	0
0	4	5	6	0
0	1	2	3	0
0	0	0	0	0

padding=1

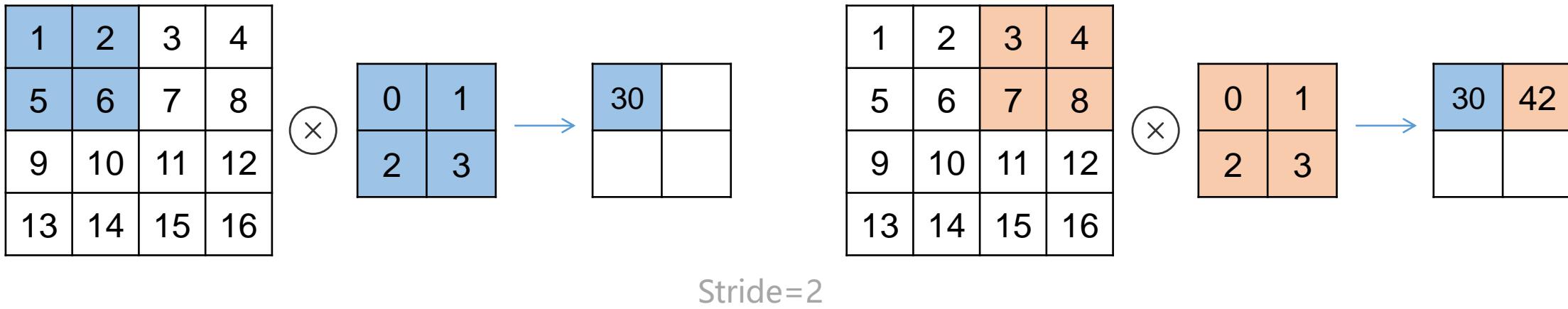
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	7	8	9	0	0
0	0	4	5	6	0	0
0	0	1	2	3	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

padding=2

步长 Stride

在卷积操作时，通常希望输出图像分辨率与输入图像分辨率相比会逐渐减少，即图像被约减。因此，可以通过改变卷积核在输入图像中移动步长大小来跳过一些像素，进行卷积滤波。

当Stride=1时，卷积核滑动跳过1个像素，这是最基本的单步滑动，也是标准的卷积模式。



当高和宽方向的步长分别为 S_h 和 S_w 时，输出特征图尺寸计算公式为：

- $H_{out} = \frac{H+2p_h-k_h}{S_h} + 1$
- $W_{out} = \frac{W+2p_w-k_w}{S_w} + 1$

特征图

特征图大小计算公式

- 输出特征图高度: $H_{out} = \left\lfloor \frac{H_{in} + 2 \times padding - kernel_h}{stride} \right\rfloor + 1$
- 输出特征图宽度: $W_{out} = \left\lfloor \frac{W_{in} + 2 \times padding - kernel_w}{stride} \right\rfloor + 1$

不同卷积下的公式变化

- Valid卷积 (padding=0)
 - 输出特征图高度: $H_{out} = \left\lfloor \frac{H_{in} - kernel_h}{stride} \right\rfloor + 1$
 - 输出特征图宽度: $W_{out} = \left\lfloor \frac{W_{in} - kernel_w}{stride} \right\rfloor + 1$

特征图

- Same卷积的目标是Same卷积的目标是保持输出尺寸与输入尺寸的比例为 1/stride

- 当步长为1: $\text{padding} = \frac{\text{kernel}-1}{2}$

- 当步长不为1: 使用公式 $H_{out} = \left\lceil \frac{H_{in}}{\text{stride}} \right\rceil, W_{out} = \left\lceil \frac{W_{in}}{\text{stride}} \right\rceil$

$$\text{padding} = \left\lfloor \frac{(\text{stride} - 1) \times H_{in} - \text{stride} + \text{kernel}}{2} \right\rfloor$$

示例计算

- 假设输入特征图大小为 32×32 , 卷积核大小为 3×3 , 步长为 1, 填充为 1

- 输出特征图高度: $H_{out} = \left\lceil \frac{32+2 \times 1-3}{1} \right\rceil + 1 = 32$

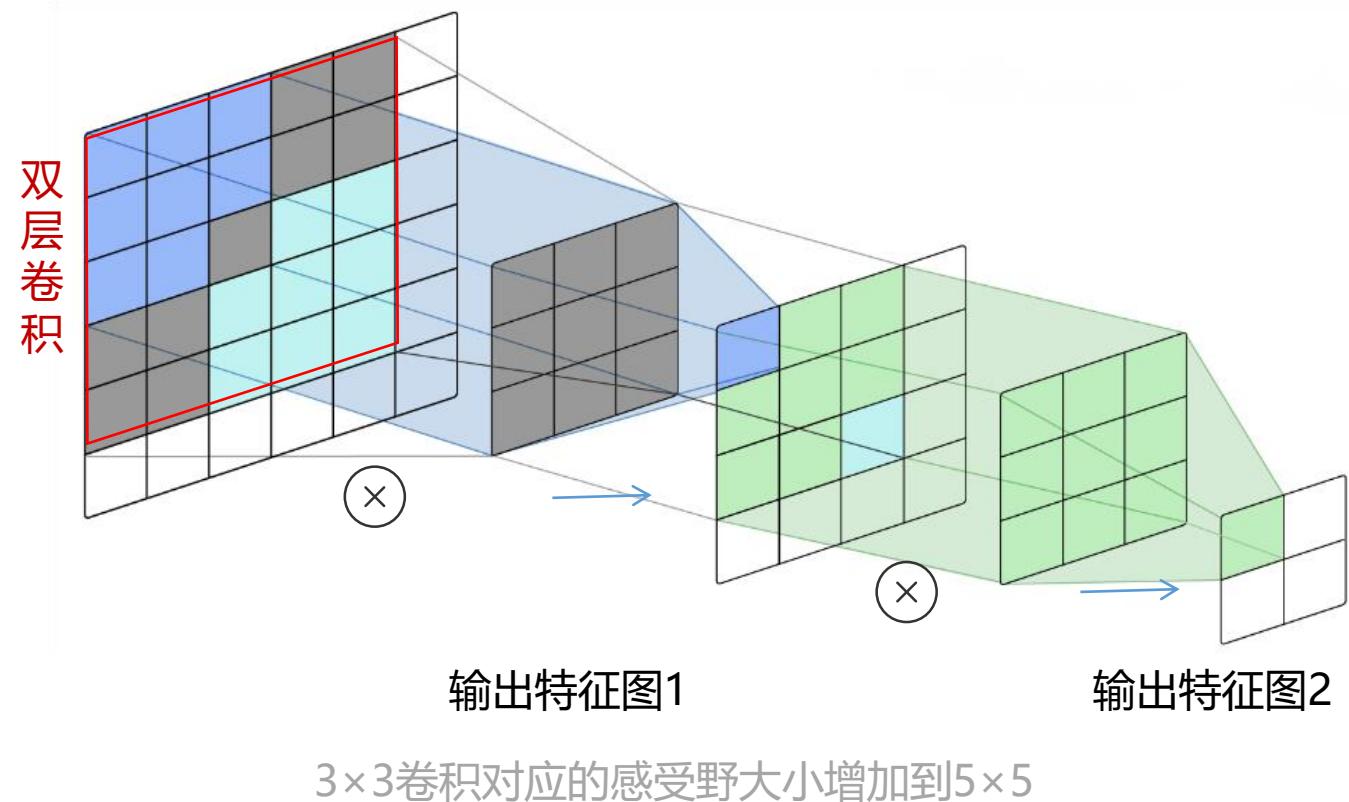
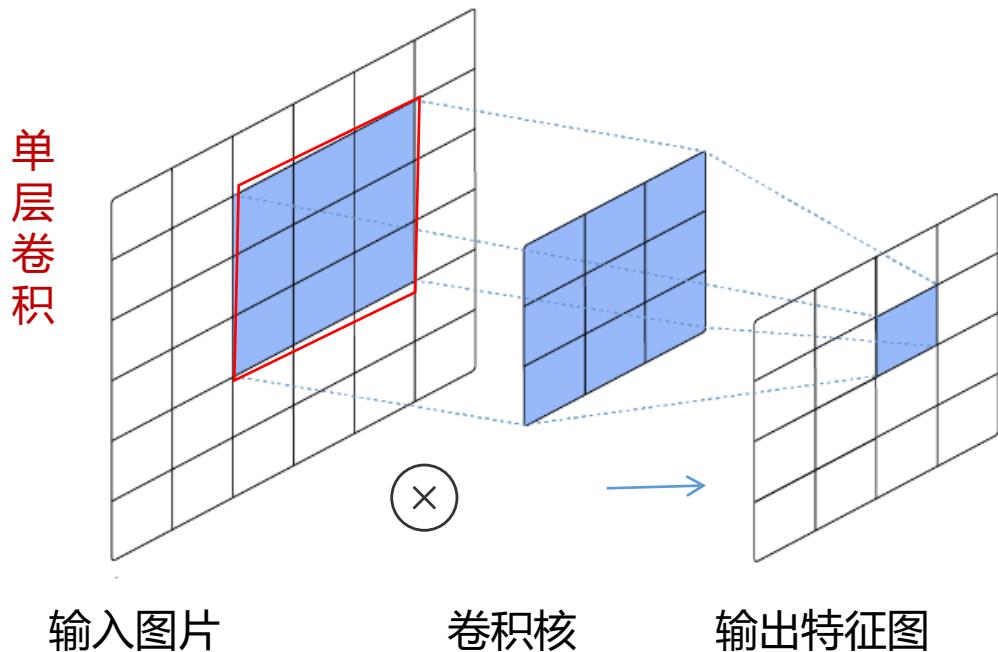
- 输出特征图宽度: $W_{out} = \left\lceil \frac{32+2 \times 1-3}{1} \right\rceil + 1 = 32$

输出的特征图大小为 32×32

感受野机制

感受野

在卷积所得结果中，每个特征图像素点取值依赖于输入图像中的某个区域，该区域被称为感受野。在卷积神经网络中，感受野是特征图上的点对应输入图像上的区域。



不同卷积核差别 $3 \times 3, 5 \times 5, 7 \times 7$

感受野

- 3×3 : 感受野小，主要用于捕获局部细节
- 5×5 : 感受野中等，可以捕获稍大的局部模式和上下文关系
- 7×7 : 感受野较大，能够感知更大范围的特征，适合融合局部与全局的信息

累计感受野：

- 两层 3×3 卷积的有效感受野为 5×5
- 三层 3×3 卷积的有效感受野为 7×7
- 直接使用 7×7 的卷积核，则一次操作即可获得较大的感受野

计算成本 参数量 = $(k \times k \times Cin + 1) \times Cout$ 【假设输入通道 $Cin=64$, 输出通道 $Cout=128$ 】

- 3×3 : 参数量 = $(3 \times 3 \times 64 + 1) \times 128 = 73,856$
- 5×5 : 参数量 = $(5 \times 5 \times 64 + 1) \times 128 = 204,928$
- 7×7 : 参数量 = $(7 \times 7 \times 64 + 1) \times 128 = 401,536$



较大的卷积核会显著增加计算成本和
参数量

不同卷积核差别 3×3 、 5×5 、 7×7

特征提取能力

3×3 :

- 优点：能够有效捕获局部细节特征，如边缘、纹理等
- 缺点：感受野较小，难以感知更大的全局信息

5×5 :

- 优点：可以捕获更大的上下文关系，提取中等规模的模式
- 缺点：相比 3×3 ，计算成本更高

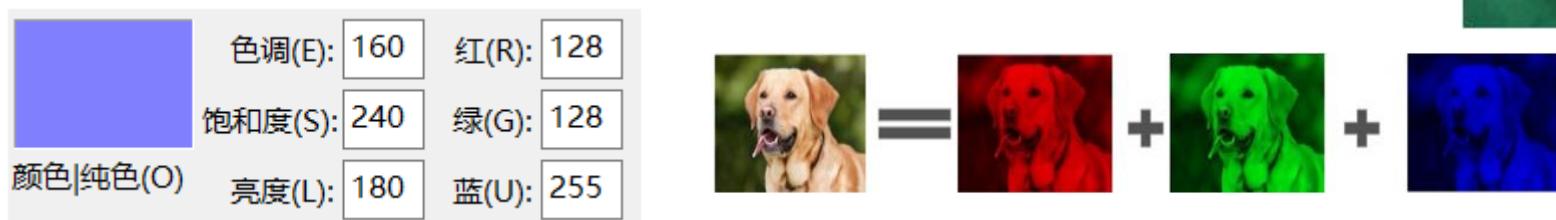
7×7 :

- 优点：适合捕获全局特征和更长距离的依赖关系，尤其是处理空间上下文时（如语义分割或注意力机制）
- 缺点：容易丢失细节信息，计算量大

图像的编码

RGB三通道编码

- 原理：计算机通过红 (Red, R) 、绿 (Green, G) 、蓝 (Blue, B) 三个通道的像素值叠加表示彩色图像，每个通道取值范围为 0~255，对应像素亮度 (0 为黑色， 255 为纯色)
- 单像素编码：用三维向量 (R, G, B) 表示，如白色为 (255, 255, 255)，黑色为 (0, 0, 0)
- 图像矩阵：一幅 $H \times W$ 的彩色图可表示为三维张量 $[H, W, 3]$ ，每个通道对应二维矩阵



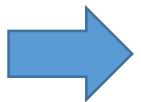
灰度图像编码（单通道）

- 将彩色图转换为单通道图像，像素值范围为 0~255 (0 为黑色， 255 为白色) ，忽略色彩信息，仅保留亮度特征
- 数学定义：灰度值Gray由RGB三通道值通过加权或平均计算得到

图像的编码示例



彩色图编码（三通道）



```
[[[168 170 180]
 [168 170 180]
 [168 170 180]
 ...
 [161 172 176]
 [161 172 176]
 [161 172 176]]
 [[168 170 180]
 [168 170 180]
 [168 170 180]
 ...
 [161 172 176]
```

灰度图（单通道）



颜色空间

颜色空间是描述颜色的数学模型，用于标准化颜色的定义、存储和显示。不同的颜色空间适用于不同的应用场景。

场景

网页/显示器显示

推荐颜色空间

视频压缩

sRGB

图像编辑（调色）

YUV/YCrCb

印刷

HSV/HSL

颜色科学（精确匹配）

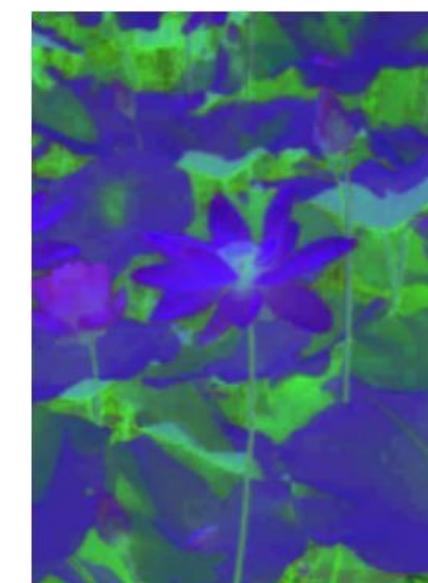
CMYK

计算机视觉（目标检测）

Lab/XYZ

RGB/HSV

```
import cv2  
# RGB转HSV  
rgb_img = cv2.imread("a.jpg")  
hsv_img = cv2.cvtColor(rgb_img, cv2.COLOR_RGB2HSV) # RGB转灰度  
gray_img = cv2.cvtColor(rgb_img, cv2.COLOR_RGB2GRAY)
```



卷积操作

RGB三通道卷积的融合计算

- 对于彩色图像，需要对每个通道分别进行卷积操作，然后将结果相加。假设输入图像为 $I(x, y, c)$ ，卷积核为 $K(k, l, c)$ ，输出特征图为 $O(i, j)$ ，则计算公式为：
- $$O(i, j) = \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} \sum_{c=0}^{C-1} I(i + k, j + l, c) K(k, l, c)$$
- 通过通道融合，可以将不同通道的信息整合在一起，提取更丰富的特征，这种融合方式使得模型能够更好地理解图像中的颜色和纹理信息，从而在各种任务中表现出色。

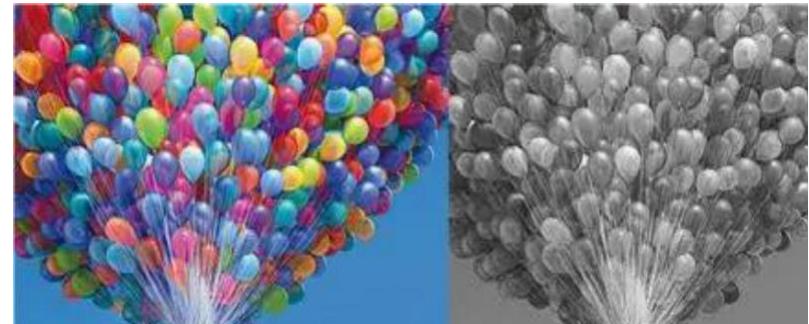
卷积的功能

- 特征提取：提取图像的边缘、纹理、形状等特征
- 噪声抑制：通过卷积核的设计，可以平滑图像，抑制噪声
- 图像增强：通过特定的卷积核，可以增强图像的某些特征，如边缘增强

彩色图与灰度图的相互转换

1. 彩色图→灰度图（灰度化）

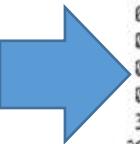
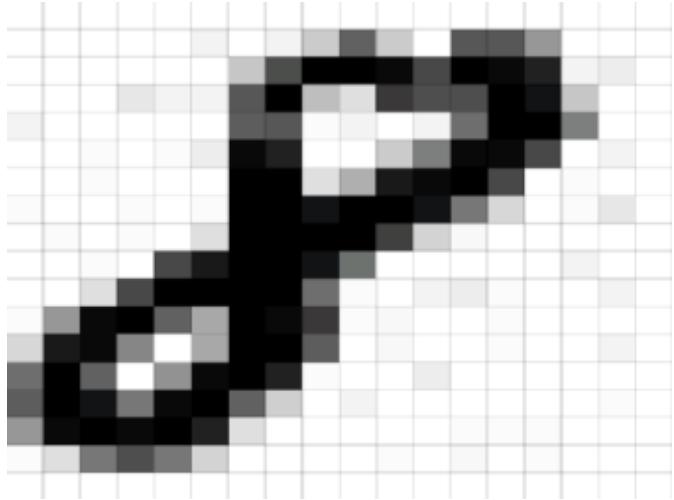
- 平均值法: $\text{Gray} = \frac{R+G+B}{3}$
- 加权平均法: $\text{Gray} = 0.3R + 0.59G + 0.11B$



2. 灰度图→彩色图（伪彩色化）

- 通过映射关系为灰度值分配色彩，常用于增强视觉对比度或可视化特征图
- 热图映射：低温区域（低灰度值）映射为蓝色，高温区域（高灰度值）映射为红色

灰度图编码示例

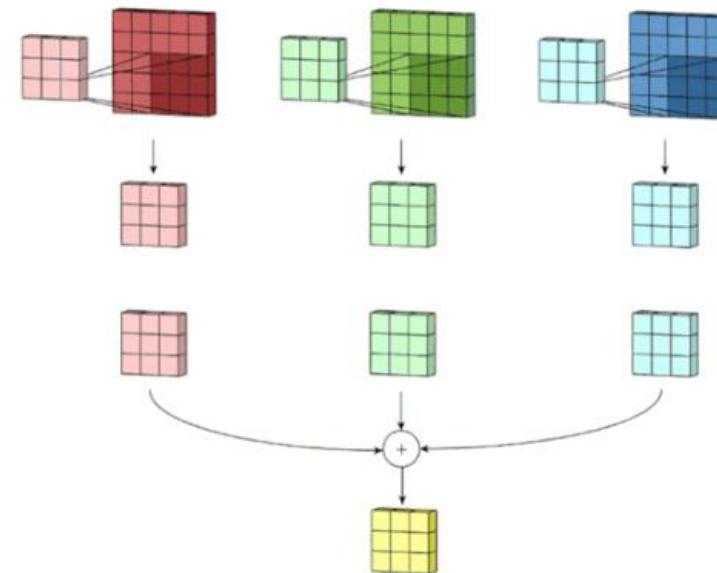
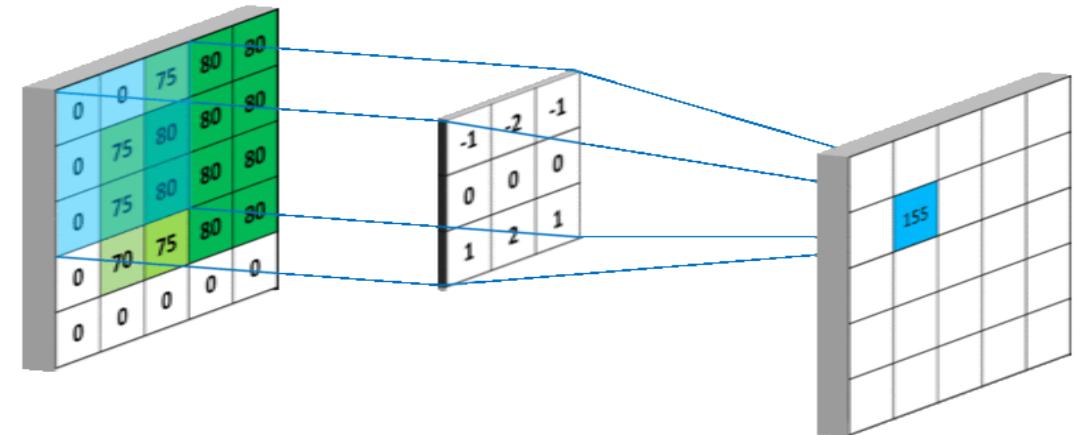


0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	1	12	8	11	39	137	37	0	152	147	84	0	0	0
0	0	1	0	0	0	41	160	250	255	235	162	255	238	286	11	13	0
0	0	0	15	9	9	150	251	45	21	184	159	154	255	233	40	0	0
10	0	0	0	0	0	145	146	3	10	0	11	124	253	255	107	0	0
0	0	3	0	4	15	236	216	0	0	38	109	247	240	169	0	11	0
1	0	2	0	0	0	253	253	23	62	224	241	255	164	0	5	0	0
6	0	0	4	0	3	252	250	228	255	255	234	112	28	0	2	17	0
0	2	1	4	0	21	255	253	251	255	172	31	8	0	1	0	0	0
0	0	4	0	163	225	251	255	229	120	0	0	0	0	0	11	0	0
0	0	21	162	255	255	254	255	126	6	0	10	14	6	0	0	9	0
3	79	242	255	141	66	255	245	189	7	8	0	0	5	0	0	0	0
26	221	237	98	0	67	251	255	144	0	8	0	0	7	0	0	11	0
125	255	141	0	87	244	255	208	3	0	0	13	0	1	0	1	0	0
145	248	228	116	235	255	141	34	0	11	0	1	0	0	0	1	3	0
85	237	253	246	255	210	21	1	0	1	0	0	6	2	4	0	0	0
6	23	112	157	114	32	8	0	0	0	2	0	8	2	7	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

卷积类型

标准卷积

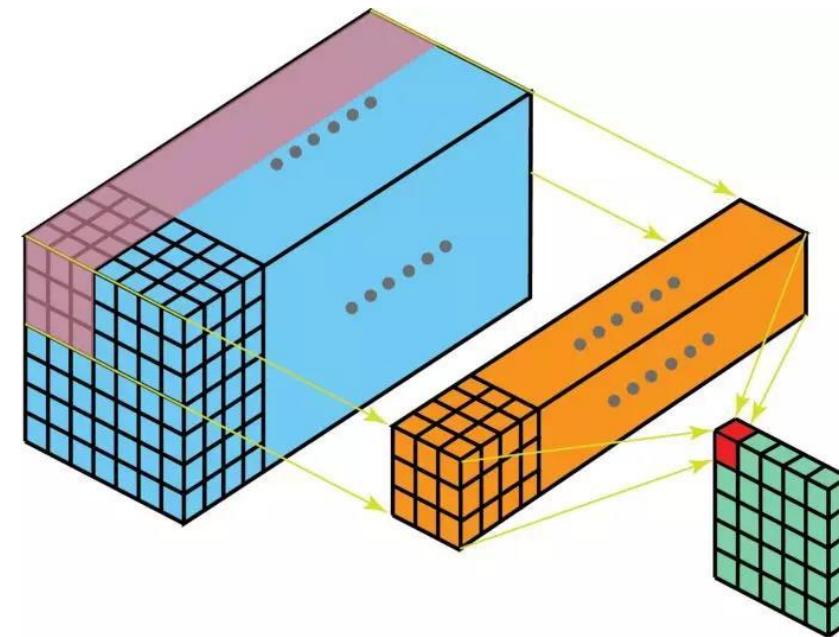
最基本的卷积操作，直接对输入图像进行卷积。标准卷积通过在输入图像上滑动卷积核，计算加权和，得到输出特征图。



卷积类型

三维卷积

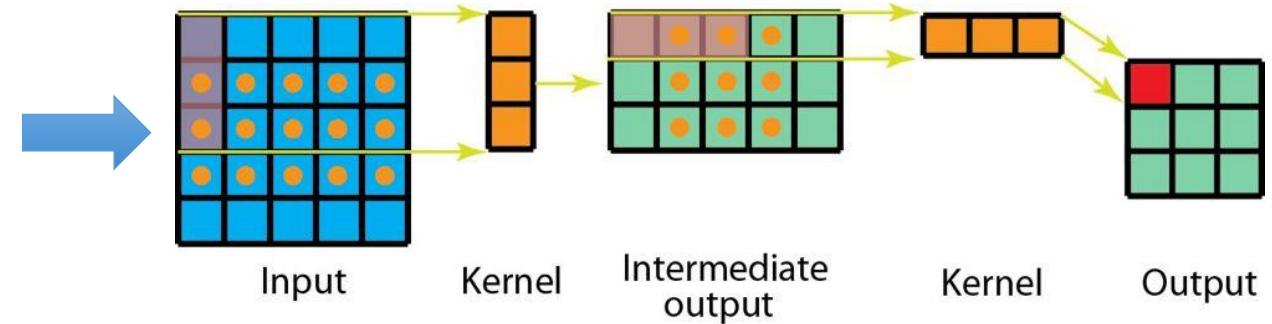
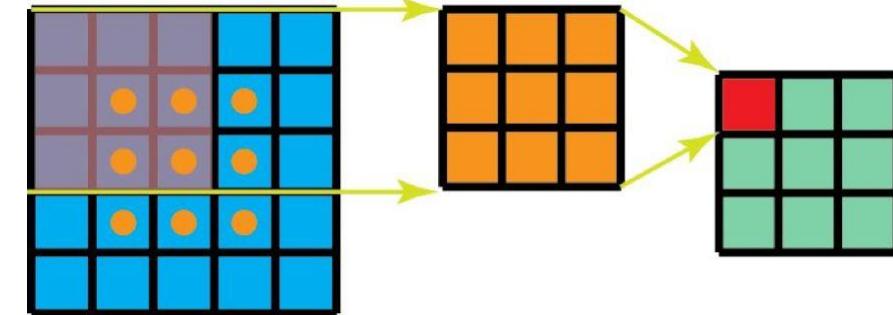
三维卷积用于处理**三维数据**（如视频、医学影像、时空序列），通过滑动3D Kernel在输入数据的**长、宽、深度**三个维度上提取特征。



卷积类型

可分卷积

- 可分卷积是将一个卷积分解为两个单独的运算。节省成本，但并非所有的核都能分成两个更小的核。



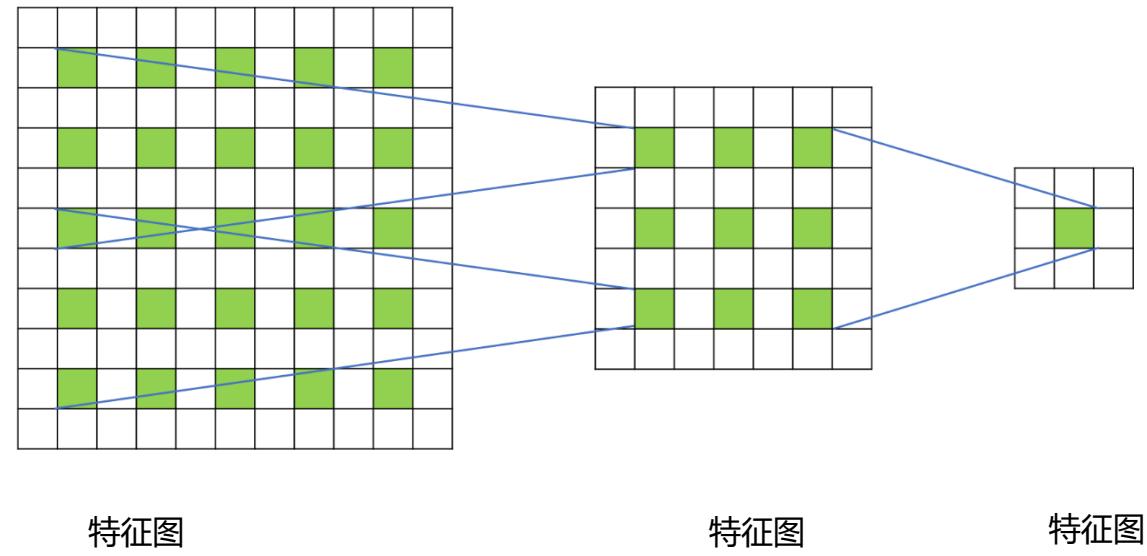
卷积类型

空洞卷积

通过在卷积核中插入空洞，扩大感受野，不增加计算量。空洞卷积通过在卷积核中插入间隔，使得卷积核能够覆盖更大的输入区域，从而扩大感受野。空洞卷积是一种上采样，没有参数参与学习，不会学习新的特征。**注意与反卷积的差别。**

获取 long-ranged information，不利于小物体的检测和语义分割。

Dilation rate: 插入空洞的数量，取值2时在卷积核的元素之间补充一个空洞。



特征图

特征图

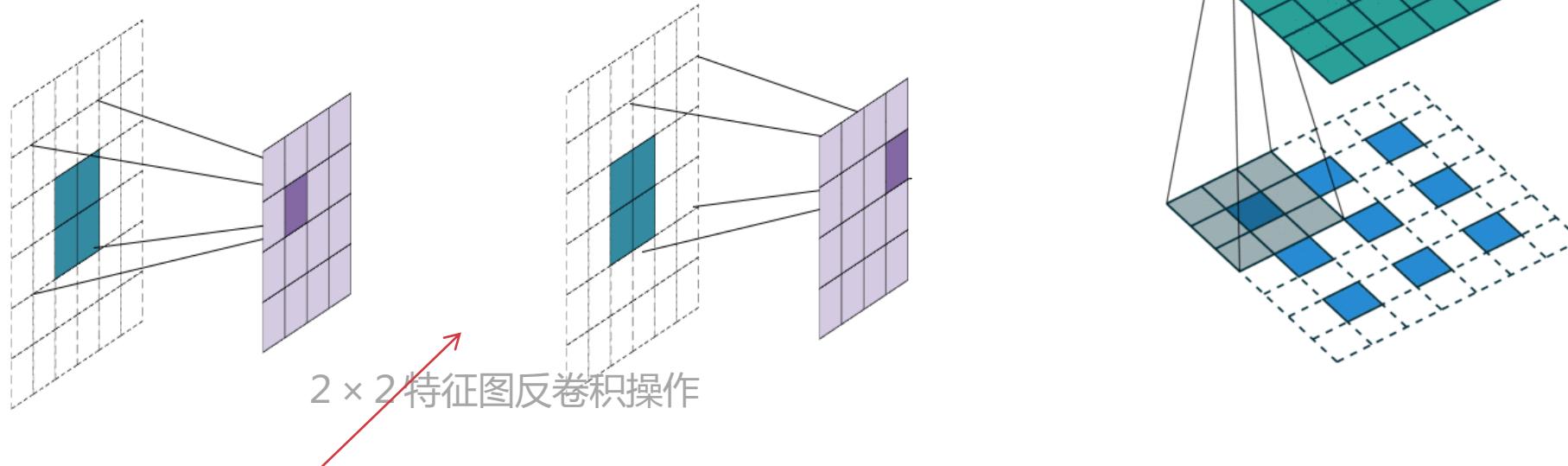
特征图

反卷积 deconvolution

反卷积与上采样是由区别的：反卷积有一个**可训练的卷积核**。

在低分辨率输入特征图的像素之间插入“0”或进行特定的填充，然后对填充后的图进行一个标准的卷积操作。这个卷积核在训练过程中会被优化。上采样一般是属于无参数的操作。

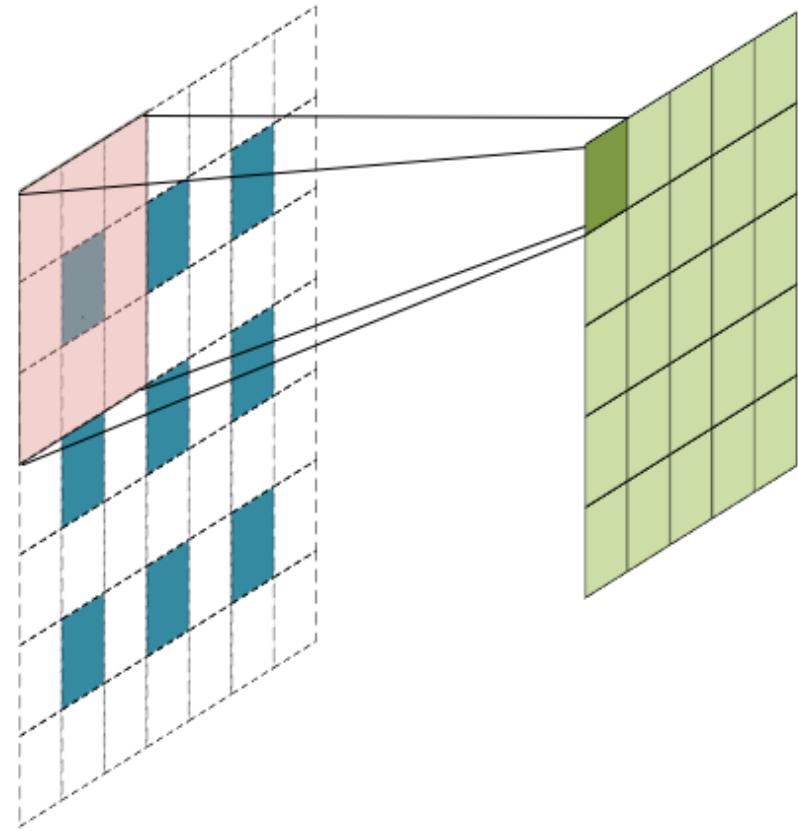
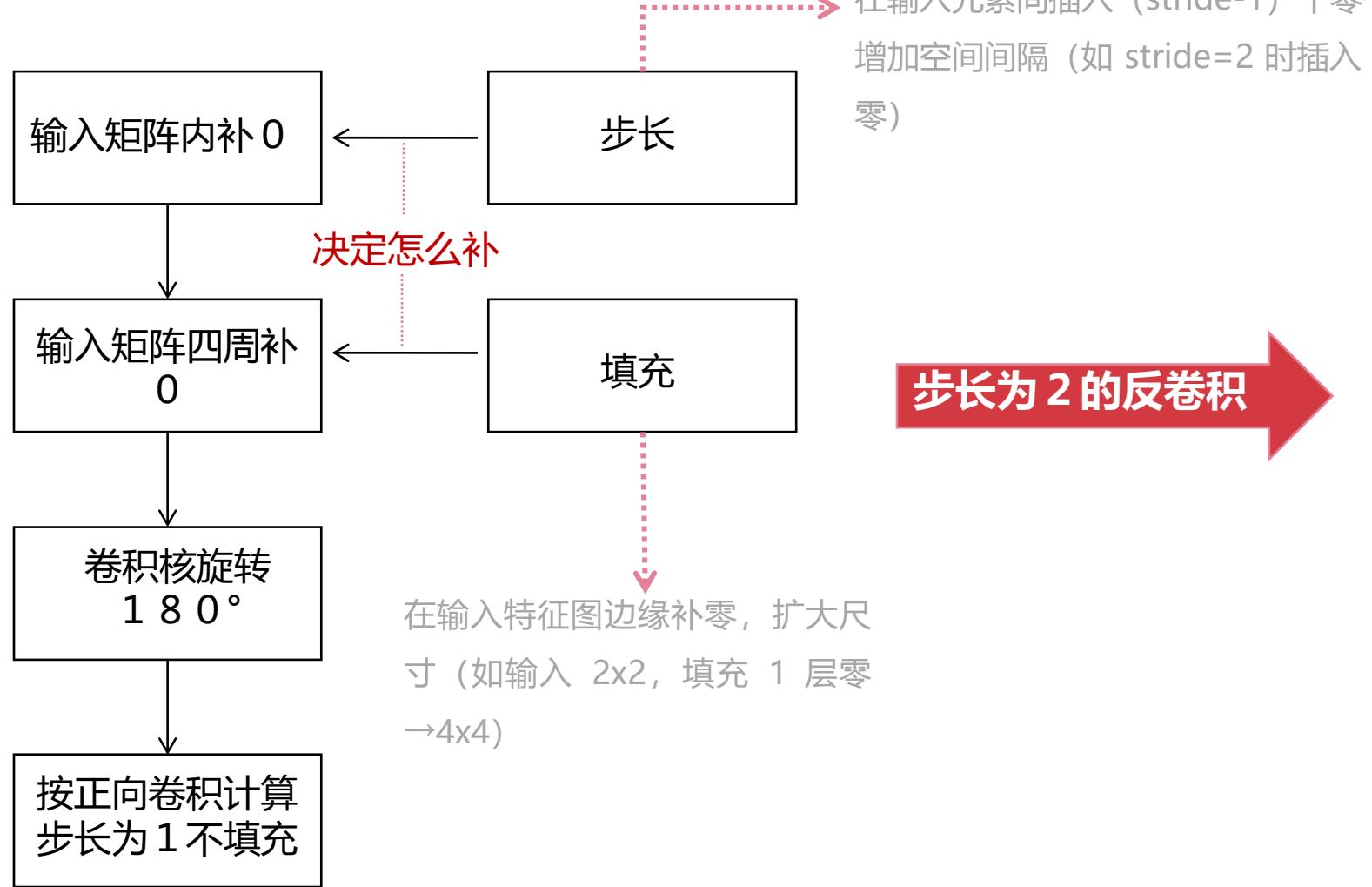
反卷积又称转置卷积，是卷积的逆操作，**核心思想**是将低维特征图（如 $H \times W$ ）映射到高维空间（如 $2H \times 2W$ ），常用于图像重建、上采样。



2×2的特征图首先通过填充变为6×6的输入图像特征，再通过3×3的卷积核逐步进行卷积计算，最后得到4×4的特征图

反卷积 deconvolution

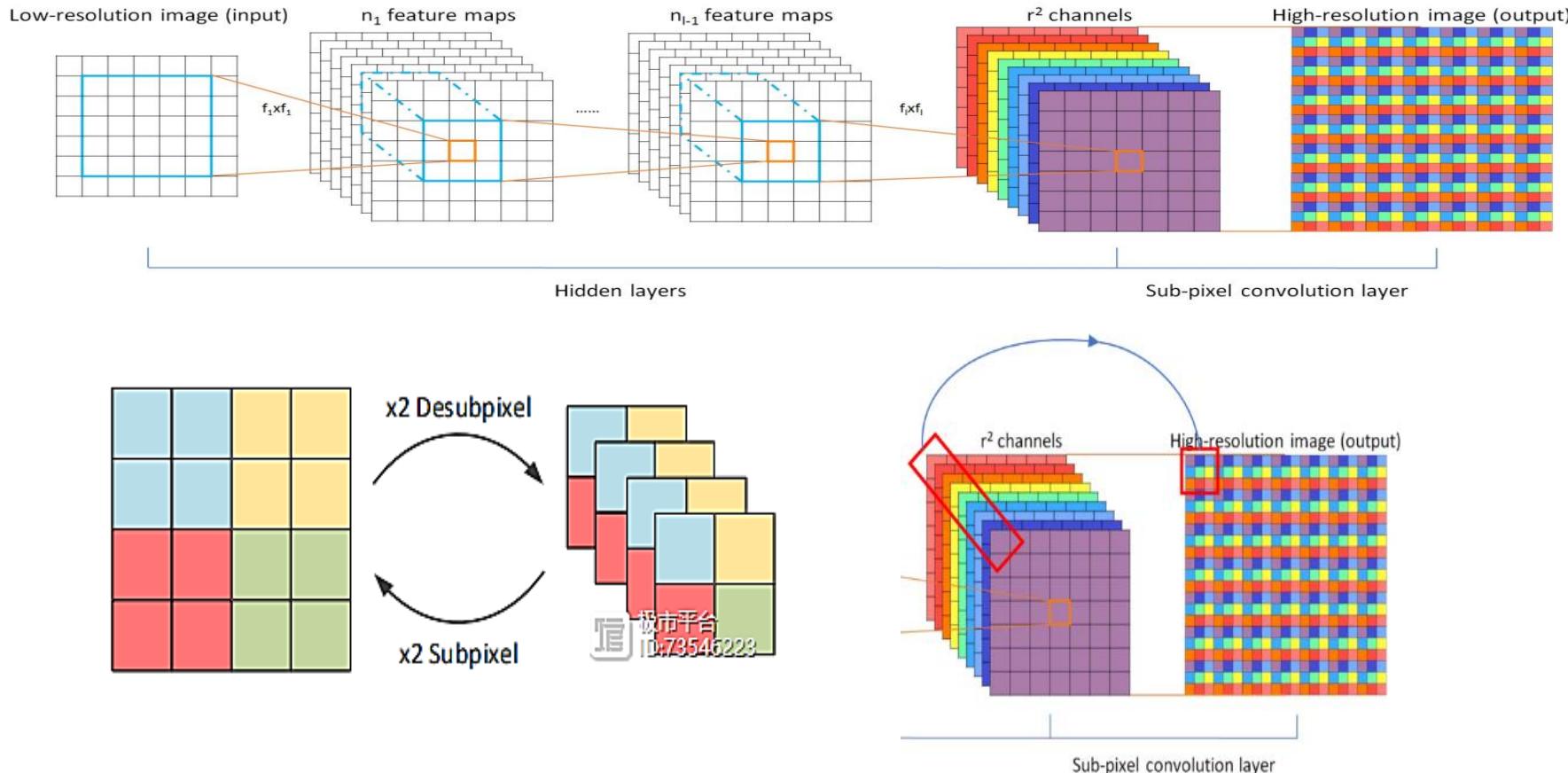
反卷积计算流程



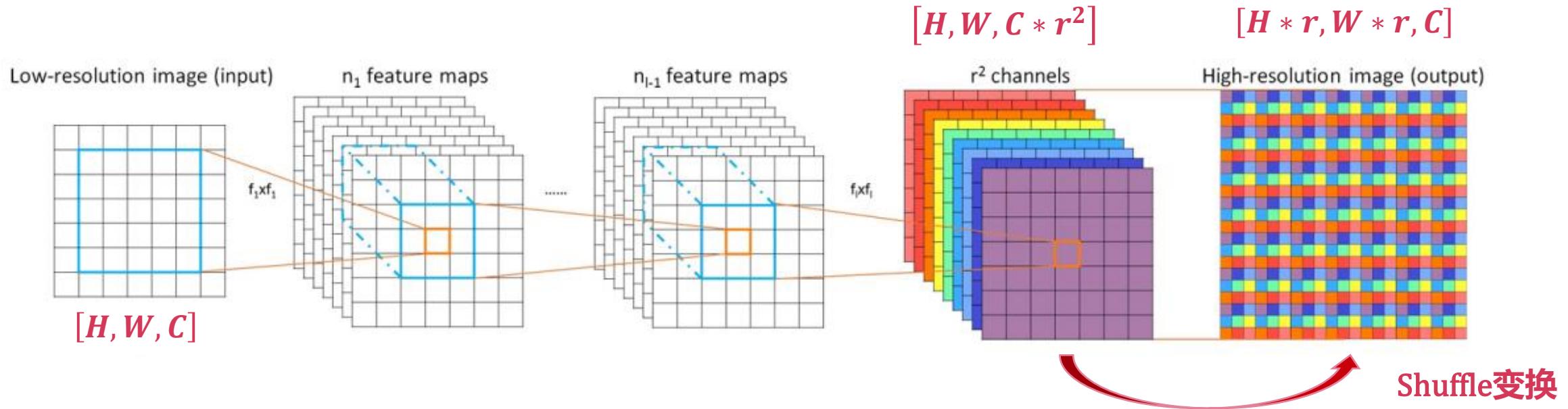
步长为 2 的反卷积

PixelShuffle 亚像素卷积

相比反卷积，PixelShuffle克服了反卷积的易产生棋盘格的问题
PixelShuffle是一种经典的上采样方法，现已广泛应用在如图像分割等计算机视觉问题上，和反卷积一起成为了神经网络中最常用的两种上采样技术。



PixelShuffle 亚像素卷积



超分辨率图像生成过程: 首先对一个大小为 $[H, W, C]$ 的低分辨率图像(Input)进行 $l - 1$ 次卷积，获得一个特征图 $[H, W, C * r^2]$ ，这里的 r 为设置的期望上采样倍数。然后对特征图做 shuffle 变换获得一张大小为 $[H * r, W * r, C]$ 的超分辨率图像(output)

池化操作pooling

池化的作用

池化层的引入是仿照人的视觉系统对视觉输入对象进行降维和抽象，它主要有以下几种作用：

- 平移不变性：输出结果对输入的平移基本保持不变
- 尺度不变性：池化操作就相当于图像的resize，图像压缩时去掉的信息只是一些无关紧要的信息，而留下的信息则是具有尺度不变性的特征，是最能表达图像的特征
- 特征降维(下采样)：把冗余信息去除，把最重要的特征抽取出来
- 池化层会不断地减小数据的空间大小，因此参数的数量和计算量也会下降，在一定程度上控制了过拟合
- 实现非线性 (组合ReLU)
- 扩大感受野

常见的池化操作

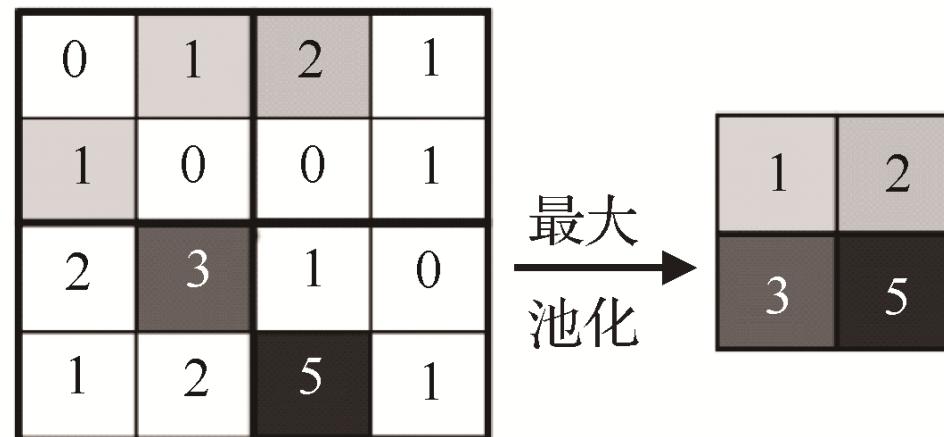
最大池化 (Max Pooling)

选择池化区域中的最大值作为输出，能够突出图像中的显著特征。例如，对于一个 2×2 的池化区域，输出该区域中的最大值。

- 特点：能够保留图像中的纹理和边缘特征，对噪声和小的扰动具有一定的鲁棒性。例如，在边缘检测任务中，最大池化可以更好地保留边缘信息。
- 假设输入特征图尺寸为 $H \times W$ ，池化窗口大小为 $k \times k$ ，步幅为 s ，则输出特征图的尺寸为

$$H' = \left\lfloor \frac{H - k}{s} \right\rfloor + 1$$

$$W' = \left\lfloor \frac{W - k}{s} \right\rfloor + 1$$

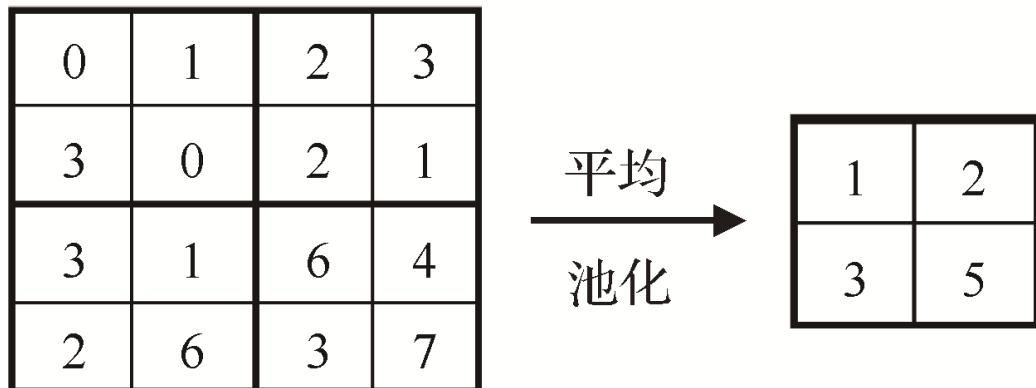


常见的池化操作

平均池化 (Mean Pooling)

计算池化区域中的平均值作为输出，能够平滑特征，保留更多的背景信息。

- 特点：能够保留图像的**整体结构信息**，适合用于提取**背景特征**。例如，在图像分类任务中，平均池化可以更好地保留图像的整体结构信息



举例

假设有一输入特征图 X , 尺寸为 4×4 , 使用 2×2 的池化窗口且步幅为2:

$$X = \begin{bmatrix} 1 & 3 & 2 & 4 \\ 5 & 6 & 8 & 7 \\ 3 & 2 & 1 & 4 \\ 5 & 3 & 2 & 6 \end{bmatrix}$$

最大池化: 输出特征图 $Y = \begin{bmatrix} 1 & 3 \\ 5 & 6 \end{bmatrix}$

窗口1: $\begin{bmatrix} 1 & 3 \\ 5 & 6 \end{bmatrix}$, 最大值6

窗口2: $\begin{bmatrix} 2 & 4 \\ 8 & 7 \end{bmatrix}$, 最大值8

窗口3: $\begin{bmatrix} 3 & 2 \\ 5 & 3 \end{bmatrix}$, 最大值5

窗口4: $\begin{bmatrix} 1 & 4 \\ 2 & 6 \end{bmatrix}$, 最大值6

平均池化: 输出特征图 $Y = \begin{bmatrix} 3.75 & 5.25 \\ 3.25 & 3.25 \end{bmatrix}$

窗口1: $\begin{bmatrix} 1 & 3 \\ 5 & 6 \end{bmatrix}$, 平均值3.75

窗口2: $\begin{bmatrix} 2 & 4 \\ 8 & 7 \end{bmatrix}$, 平均值5.25

窗口3: $\begin{bmatrix} 3 & 2 \\ 5 & 3 \end{bmatrix}$, 平均值3.25

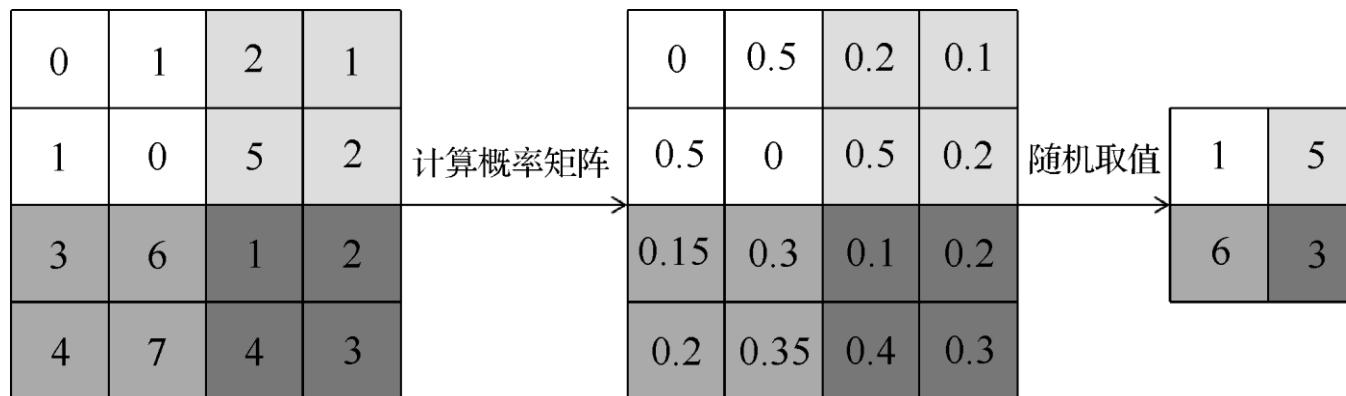
窗口4: $\begin{bmatrix} 1 & 4 \\ 2 & 6 \end{bmatrix}$, 平均值3.25

常见的池化操作

随机池化 (Stochastic Pooling)

按照池化区域中像素值的概率分布随机选择一个值作为输出。每个像素值被选中的概率与其值成正比，值越大被选中的概率越高。

- 特点：随机池化可以看作一种正则化方式，能够增加模型的泛化能力。例如，在训练过程中，随机池化可以防止模型过拟合，提高模型的泛化能力。

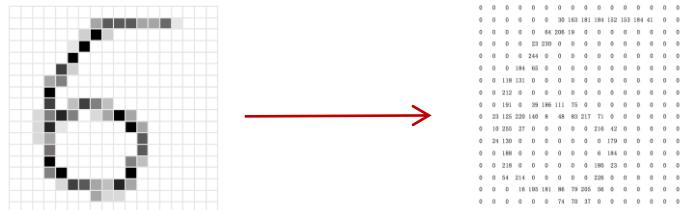


完整的卷积神经网络

01

输入层

输入图像，并将图像转换为其对应的由像素值构成的二维矩阵，并将此二维矩阵存储，等待后面几层的操作



02

卷积层

将输入信号在某一特征上加强，从而实现特征的提取，也可以排除干扰因素，从而降低特征的噪声

03

激活函数

输入特征图经过卷积层后，需要输入给激活函数，进行非线性变换得到激活输出

04

池化层

池化有以下主要功能：①能够缩小输入特征图的维度，使参数数量和运算量减少，在一定程度上可以避免过拟合；②增强网络对输入图像中的微小变化的稳健性

05

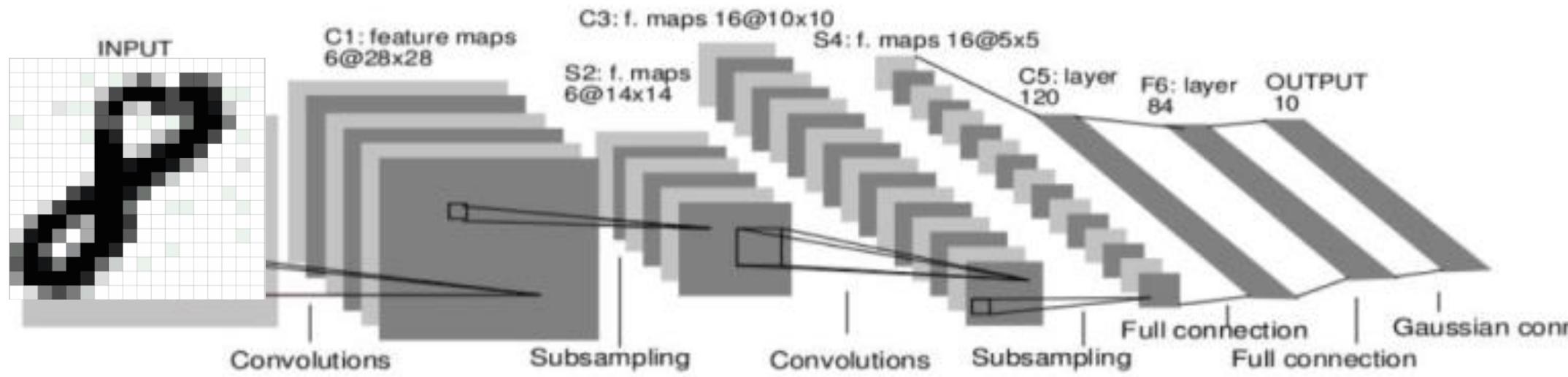
全连接层

综合低层的各个卷积层特征，学习特征间的非线性组合，利用全连接层将特征结合到一起

06

输出层

除了输出结果外，还有另一项任务是进行反向传播训练，依次向后进行梯度传递，计算相应的损失函数，并重新更新权重



卷积神经网络优化与训练

权重初始化 (Weight Initialization)

打破对称性，避免神经元输出相同值

常见方法

- 随机初始化：正态分布/均匀分布（如TruncatedNormal）
- Xavier初始化：保持输入输出方差一致，适用于 Sigmoid/Tanh 激活
 - $W \sim N(0, \frac{2}{n_{in} + n_{out}})$
- He初始化：针对 ReLU 设计
 - $W \sim N(0, \frac{2}{n_{in}})$

n_{in}, n_{out} 表示输入输出层神经元个数

参数设置建议

- 参数值（张量的通道数、特征图尺寸、批量大小）设为2的幂，如64、128，优化内存管理，提升计算效率

卷积神经网络优化与训练

前向传播 (Forward Propagation)

流程

- 输入数据 → 逐层计算 → 输出预测值

核心操作

- 卷积层：提取空间特征
- 激活层：引入非线性
- 池化层：降维
- 全连接层：特征整合

损失函数计算 (Loss Calculation)

目标

- 衡量预测值与真实值的差异

分类任务：交叉熵损失 (Cross-Entropy Loss) ; 回归任务：均方误差 (MSE Loss)

卷积神经网络优化与训练

反向传播与权重更新 (Backpropagation & Optimization)

原理

- 通过链式法则计算梯度，使用优化器更新权重

关键组件：

- 梯度计算：MindSpore 自动微分
- 优化器
 - 随机梯度下降 (SGD)
 - Adam：自适应学习率

更新公式： $W_{t+1} = W_t - \eta \cdot \nabla L(W_t)$

反向传播训练

Step 1: 用随机数初始化所有的滤波器和参数/权重

Step 2: 网络将训练图片作为输入，执行前向步骤（卷积，ReLU，池化以及全连接层的前向传播）并计算每个类别的对应输出概率

- 假设图的输出概率是[0.2, 0.4, 0.1, 0.3]

Step 3: 计算输出层的总误差

- 总误差 = $\frac{1}{2} \sum (\text{目标概率} - \text{输出概率})^2$

Step 4: 反向传播算法计算误差相对于所有权重的梯度，并用梯度下降法更新所有的滤波器/权重和参数的值，以使输出误差最小化

- 权重的调整程度与其对总误差的贡献成正比
- 当同一图像再次被输入，这次的输出概率可能是[0.1, 0.1, 0.7, 0.1]，与目标[0, 0, 1, 0]更接近了，这说明神经网络已经学习分类特定图片了，学习的方式是调整权重/滤波器以降低输出误差

超参数调优策略

01

学习率 (Learning Rate) 调优

核心目标：平衡收敛速度
与最优解搜索

策略类型	典型方法	适用场景
固定学习率	恒定值 (如 0.01)	简单模型或小数据集
衰减学习率	指数衰减、余弦衰减、阶梯衰减	防止后期震荡，提升精度
自适应学习率	Adam、RMSprop、Adagrad	复杂模型或大规模数据

超参数调优策略

02

正则化 (Regularization) 防过拟合

正则化是指在机器学习和统计建模中的一种技术，用于控制模型的复杂度，防止模型在训练数据上过度拟合 (overfitting)。

正则化通过在模型的损失函数中引入额外的惩罚项，来对模型的参数进行约束，从而降低模型的复杂度，这个额外的惩罚通常与模型参数的大小或者数量相关。

L1:

$$L'(\theta) = L(\theta) + \lambda \sum_{i=1}^n |\theta_i|$$

L2:

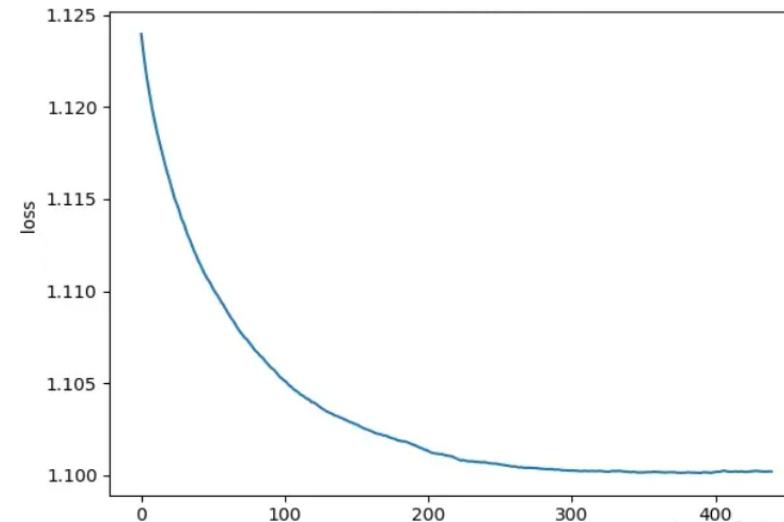
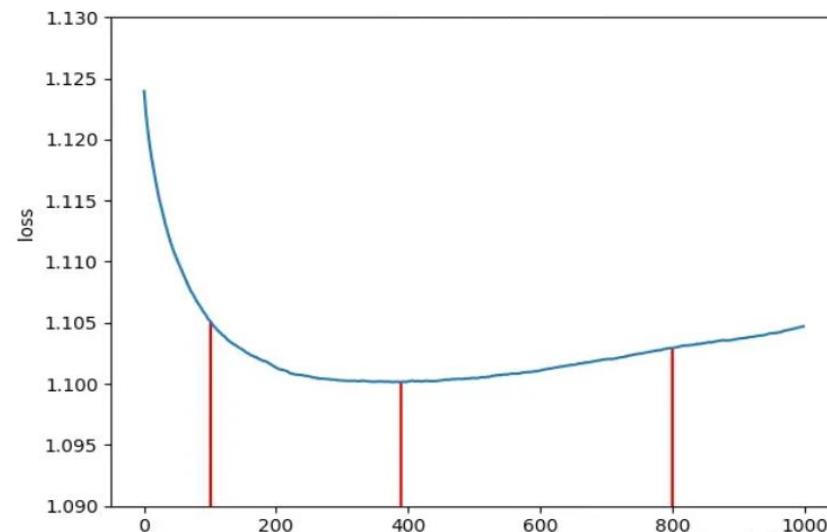
$$L'(\theta) = L(\theta) + \lambda \sum_{i=1}^n \theta_i^2$$

防过拟合--Dropout

- DNNs是以概率 p 舍弃部分神经元，其它神经元以概率 $q=1-p$ 被保留，舍去的神经元的输出都被设置为零
- Dropout在实践中能很好工作是因为其在训练阶段阻止神经元的共适应

防过拟合--早停机制

- 一个epoch结束时（一个epoch即对所有训练数据的一轮遍历）计算validation data的total loss，当loss不再降低时，就停止训练，这样可以防止过拟合



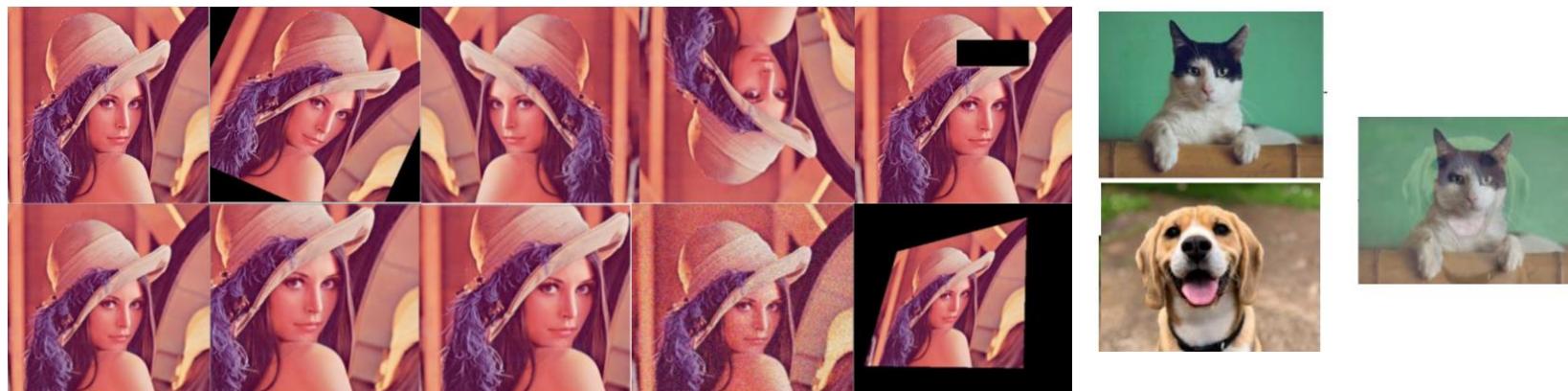
超参数调优策略

03

数据增强 (Data Augmentation)

核心作用：扩大数据集多样性，提升模型泛化能力

操作类型	示例	目标场景
几何变换	随机裁剪、翻转、旋转	增强平移 / 旋转不变性
颜色变换	亮度、对比度调整	适应光照变化
混合增强	Mixup、Cutout	抑制过拟合，提升鲁棒性



其他关键超参数

批量大小 (Batch Size)

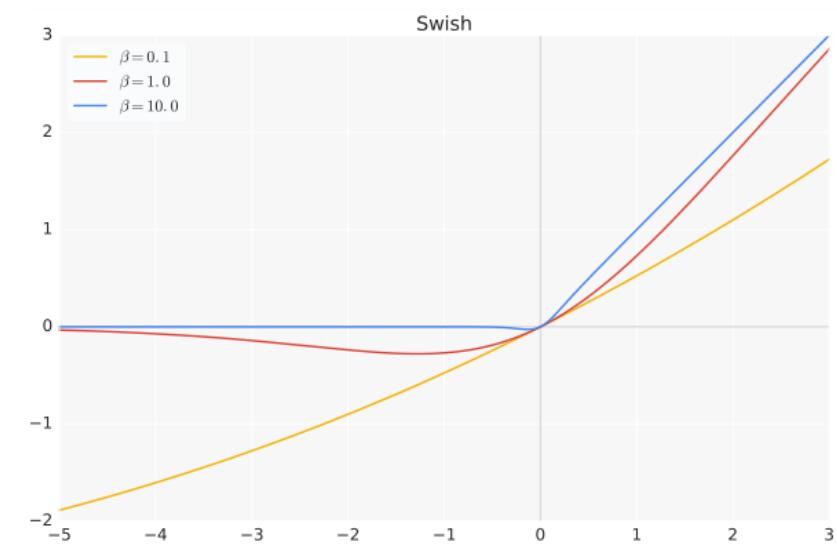
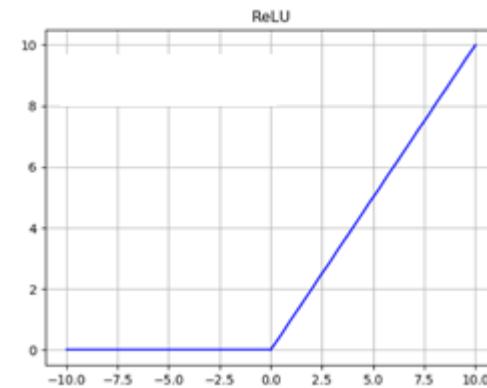
- 小批量：收敛稳定，适合显存有限场景
- 大批量：加速训练，但需更大显存

网络深度与宽度

- 深度：层数增加（如 ResNet-50 vs LeNet），提取更高级特征
- 宽度：卷积核数量（如从 64→128 通道），提升特征表达能力

激活函数

- ReLU：缓解梯度消失，默认首选
- Swish：平滑非线性，适合复杂模型



CNNCTC 图像文本识别

任务

- 基于 MindSpore 框架搭建 CNNCTC 模型，实现图像文本识别的训练与离线推理。

目标

- 掌握 MindSpore 框架搭建
- 学习香橙派 Alpro 开发板的离线模型转换与推理实现

实验环境

- [Mindspore](#)
- [ModelArts](#)
- 香橙派Alpro开发板

模型下载

训练代码下载

- 进入MindSpore官网，下载CNNCTC案例的notebook文档：
 - <https://www.mindspore.cn/tutorials/application/zh-CN/r2.2/cv/cnnctc.html>

实验环境搭建

- 对于平台环境的搭建可以参考《ModelArts云环境搭建指南》
- 实验数据处理所需的存储空间比较大，故在环境搭建时，“云硬盘EVS”这项的设置可稍大些，建议设置为100GB，如图所示：



ModelArts云环境
搭建指南.docx



- 按上述指南进入ModelArts开发平台后，要将MindSpore版本升级最新版
 - 在终端运行指令

```
pip install mindspore
```

- 或下载whl包进行安装

模型训练 ModelArts

- 将之前下载好的训练代码文档上传到 ModelArts 开发平台

代码修正

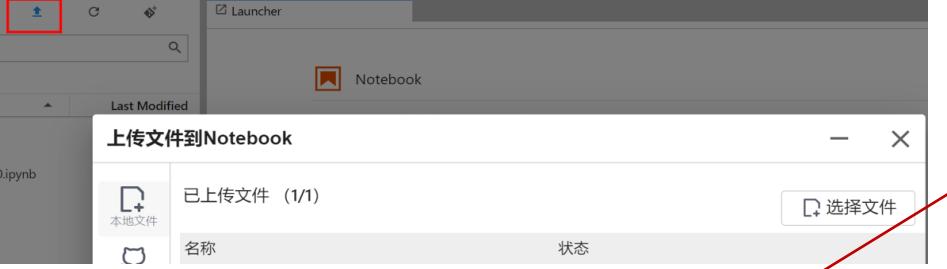
- 在数据集准备与加载模块添加数据下载权限

```
%env no_proxy='a.test.com,127.0.0.1,2.2.2.2'
```

- 在模型训练模块修改算力相关参数
- 在训练前先改图模式为静态图 (ms.GRAPH_MODE)

训练并导出模型

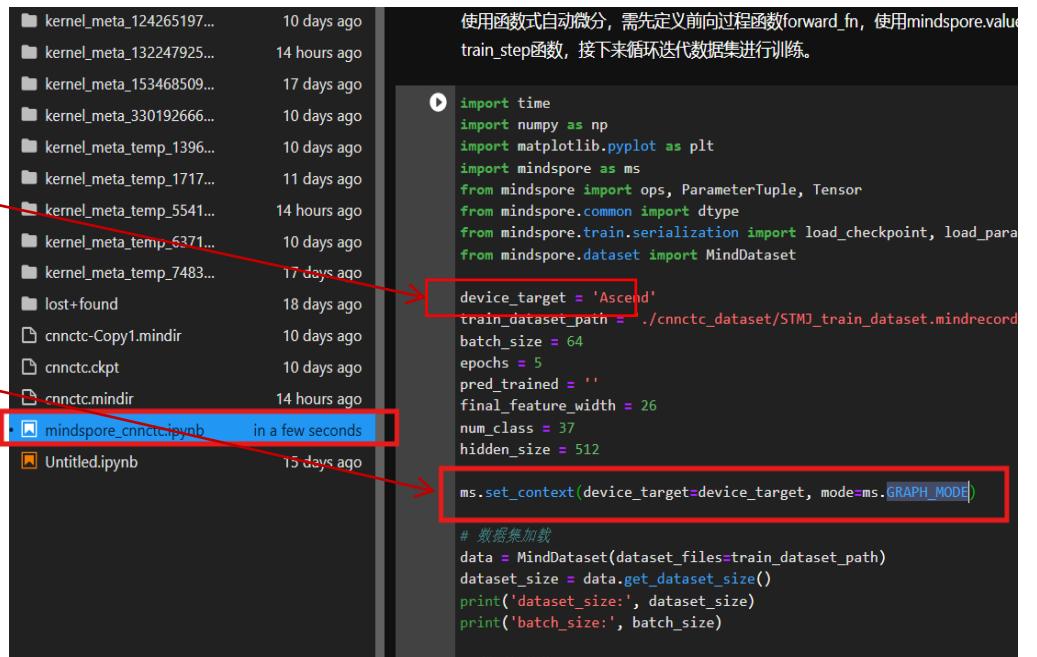
- 导出模型前切换为动态图 (ms.PYNATIVE_MODE)
- 训练完成后，通过ms.export导出 MINDIR 模型



```
%env no_proxy='a.test.com,127.0.0.1,2.2.2.2'
from download import download

url = "https://mindspore-website.obs.cn-north-4.myhuaweicloud.com/notebook/datasets/
download(url, ".", kind="tar.gz", replace=True)
```

模型构建



使用函数式自动微分，需先定义前向过程函数forward_fn，使用mindspore.value.train_step函数，接下来循环迭代数据集进行训练。

```
import time
import numpy as np
import matplotlib.pyplot as plt
import mindspore as ms
from mindspore import ops, ParameterTuple, Tensor
from mindspore.common import dtype
from mindspore.train.serialization import load_checkpoint, load_params_into_net
from mindspore.dataset import MindDataset

device_target = 'Ascend'
train_dataset_path = './cnnctc_dataset/STMJ_train_dataset.mindrecord'
batch_size = 64
epochs = 5
pred_trained = ''
final_feature_width = 26
num_class = 37
hidden_size = 512

ms.set_context(device_target=device_target, mode=ms.GRAPH_MODE)

# 数据集加载
data = MindDataset(dataset_files=train_dataset_path)
dataset_size = data.get_dataset_size()
print('dataset_size:', dataset_size)
print('batch_size:', batch_size)
```

开发板环境准备

- 登录香橙派，升级 MindSpore 至 最新 版本
- 参考《手把手教你搭建Orange Pi AI Pro开发环境》实验手册



手把手教你搭建Orange Pi Alpro开发

推理过程

- 下载项目代码

```
cd /home/HwHiAiUser/samples/notebooks/  
git clone https://github.com/mindspore-courses/orange-pi-mindspore.git
```

- 上传mindir模型文件至指定目录
- 在 “/home/HwHiAiUser/samples/notebooks” 目录下运行如下命令，生成om模型文件

```
#获取bash.sh文件  
wget https://mindspore-courses.obs.cn-north-4.myhuaweicloud.com/orange-pi-mindspore/package/bash.sh
```

模型推理过程 香橙派 Alpro

模型转换 (MINDIR→OM)

#执行bash.sh文件

①

```
source bash.sh /home/HwHiAiUser/samples/notebooks/cnnctc2.2.14/cnnctc.mindir cnnctc
```

②

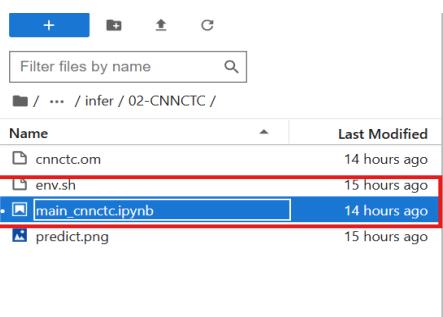
bash.sh文件执行时需要传入两个参数：

- 第一个参数是开发板上存放的MINDIR文件的绝对路径
- 第二个参数是生成的om文件的名称

推理执行

- 修改start_notebook.sh中的 IP 为192.168.137.100，启动 JupyterLab 环境
- 运行main_cnnctc.ipynb前，注释 OM 下载代码，执行推理

```
start_notebook.sh
1 . /usr/local/Ascend/ascend-toolkit/set_env.sh
2 export PYTHONPATH=/usr/local/Ascend/thirdpart/aarch64/acclite:$PYTHONPATH
3
4 if [ $# -eq 1 ];then
5     jupyter lab --ip $1 --allow-root --no-browser
6 else
7     jupyter lab --ip 192.168.137.100 --allow-root --no-browser
8 fi
```



A screenshot of a Jupyter Notebook cell. The code is as follows:

```
# 推理过程
preds_size = np.array([model_predict.shape[1]])
preds_index = np.argmax(model_predict, 2)
preds_index = np.reshape(preds_index, [-1])
preds_str = converter.decode(preds_index, preds_size)
print('Predict: ', preds_str)

infer use time:11.1453317260742ms
Predict: ['parking']
```

The output of the code is shown below the cell.



模型输出：Predict: ['parking']

https://www.mindspore.cn/tutorials/application/zh-CN/r2.2/index.html

[M]^s 昕思
MindSpore

安装 学习 文档 产品全景 资源 社区 活动与资讯 昕思大模型平台 HOT

教程 2.2 ▾ 初学入门 应用实践 深度开发

"教程" 内搜索 教程 (2.2) > 应用实践

欢迎查看MindSpore教程

计算机视觉 ResNet50图像分类 ResNet50迁移学习 Vision Transformer图像分类 CNN+CTC图像文本识别 FCN图像语义分割 ShuffleNet图像分类 SSD目标检测

自然语言处理 > RNN实现情感分类 LSTM+CRF序列标注

生成式 >

应用实践

计算机视觉

- ResNet50图像分类
- ResNet50迁移学习
- Vision Transformer图像分类
- CNN+CTC图像文本识别
- FCN图像语义分割
- ShuffleNet图像分类
- SSD目标检测

自然语言处理

- RNN实现情感分类
- LSTM+CRF序列标注

生成式

Thank you.

把数字世界带入每个人、每个家庭、
每个组织，构建万物互联的智能世界。

Bring digital to every person, home, and
organization for a fully connected,
intelligent world.

Copyright©2023 Huawei Technologies Co., Ltd.
All Rights Reserved.

The information in this document may contain predictive statements including, without limitation, statements regarding the future financial and operating results, future product portfolio, new technology, etc. There are a number of factors that could cause actual results and developments to differ materially from those expressed or implied in the predictive statements. Therefore, such information is provided for reference purpose only and constitutes neither an offer nor an acceptance. Huawei may change the information at any time without notice.

