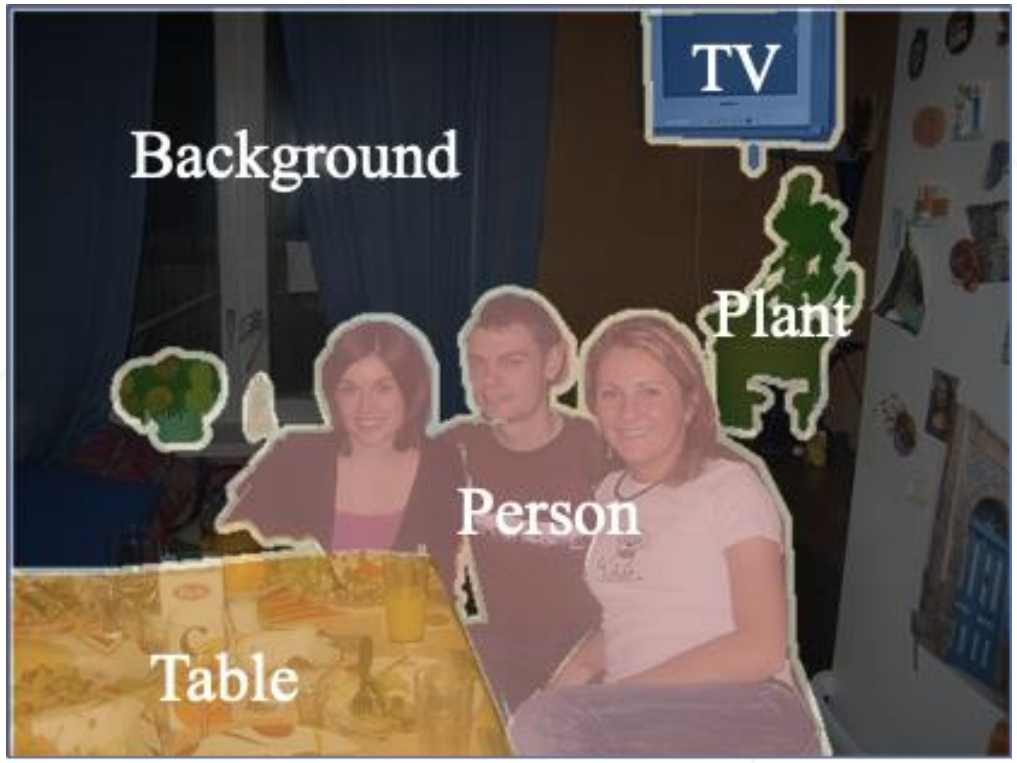
OpenMMLab

通用视觉框架OpenMMLab

第5讲 语义分割与MMSegmentation

什么是语义分割



任务：　将图像按照物体的类别分割成不同的区域

等价于：　对每个像素进行分类

应用：无人驾驶汽车

自动驾驶车辆，会将行人，其他车辆，行车道，人行道、交通标志、房屋、草地与树木等等按照类别在图像中分割出来，从而辅助车辆对道路的情况进行识别与认知。

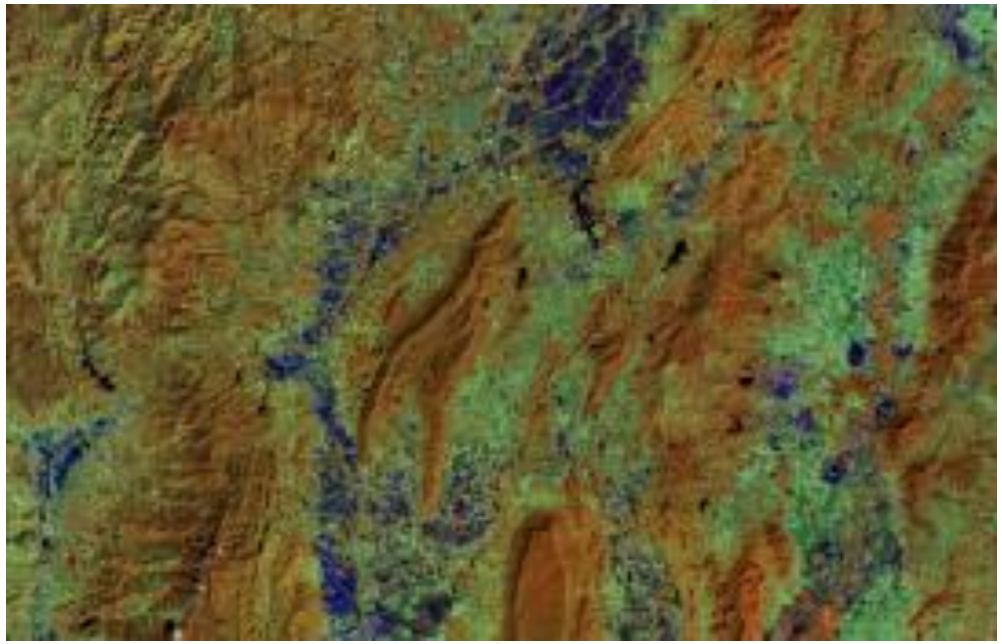
应用：人像分割

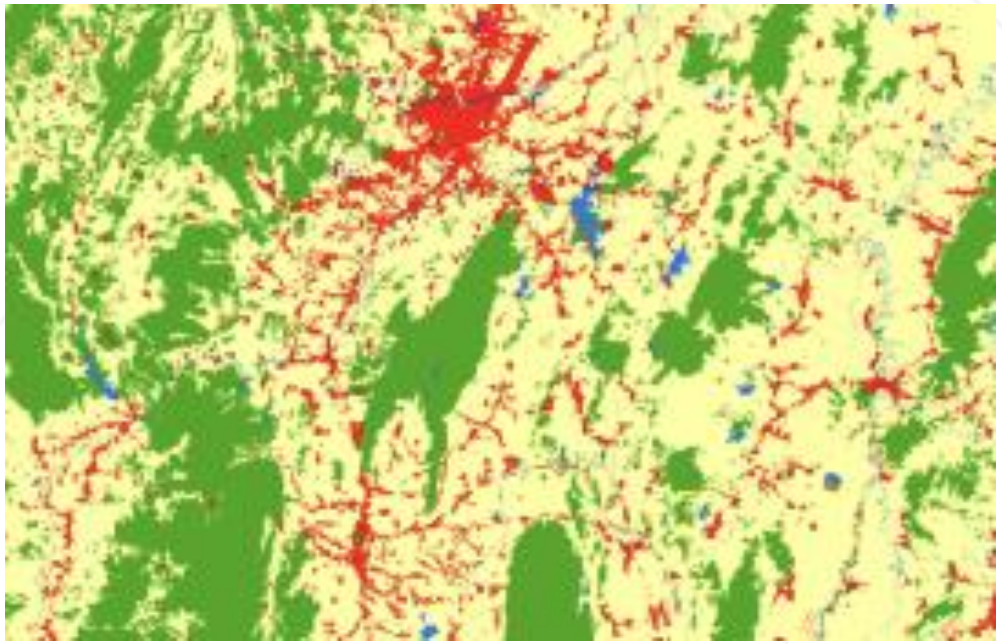
实时替换视频的背景

在智慧互娱和智能会议场景中，可以通过这种方法增加交互的多样性

图片来源：https://ai.googleblog.com/2018/03/mobile-real-time-video-segmentation.html

应用：智能遥感

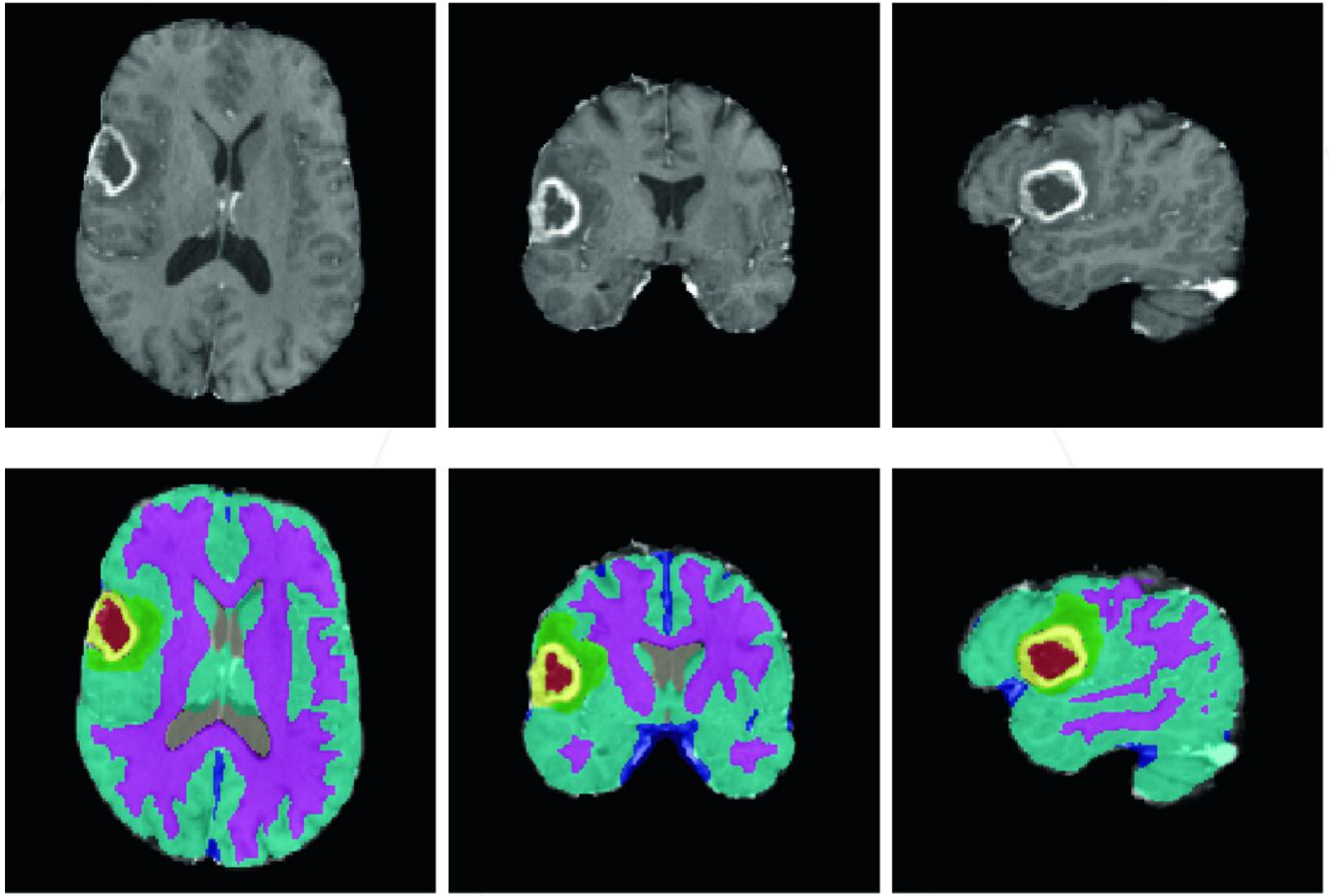




分辨地表物体的类别，通过右侧分割之后的图像可以看到，红色的部分属于湖泊水流。通过智能遥感能够监测不同季节地表水域的变化，从而辅助农业生产，以及旱灾洪灾的预测等等。

应用：医疗影像分析

通过图像分割技术，辅助进行医疗诊断。如右图，识别脑部肿瘤异物的位置。



语义分割 vs 实例分割 vs 全景分割

语义分割

仅考虑像素的类别

不分割同一类的不同实体

实例分割

分割不同的实体

仅考虑前景物体

全景分割

背景仅考虑类别

前景需要区分实体

内容概要

本节内容：

- 语义分割的基本思路
- 深度学习下的语义分割模型
  - 全卷积网络
  - 空洞卷积与 DeepLab 模型
  - 上下文信息与 PSPNet 模型
- 分割模型的评估方法
- 实践 MMSegmentation

基本思路：按颜色分割

先验知识

物体内部颜色相近，物体交界颜色变化

基于图像处理方法
按照颜色分割

需要额外手段确定物体类别

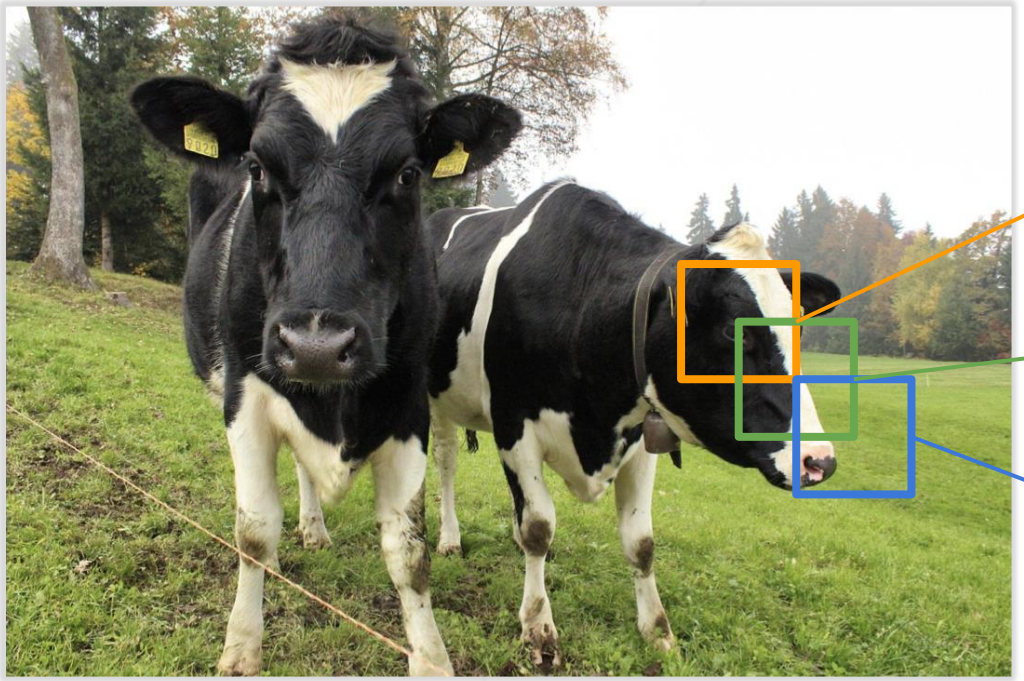
问题

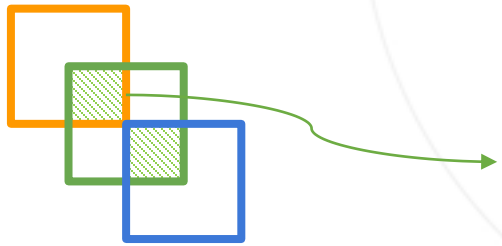
先验知识不完全准确：
不同物体颜色可能相近，物体内也会包含多种颜色

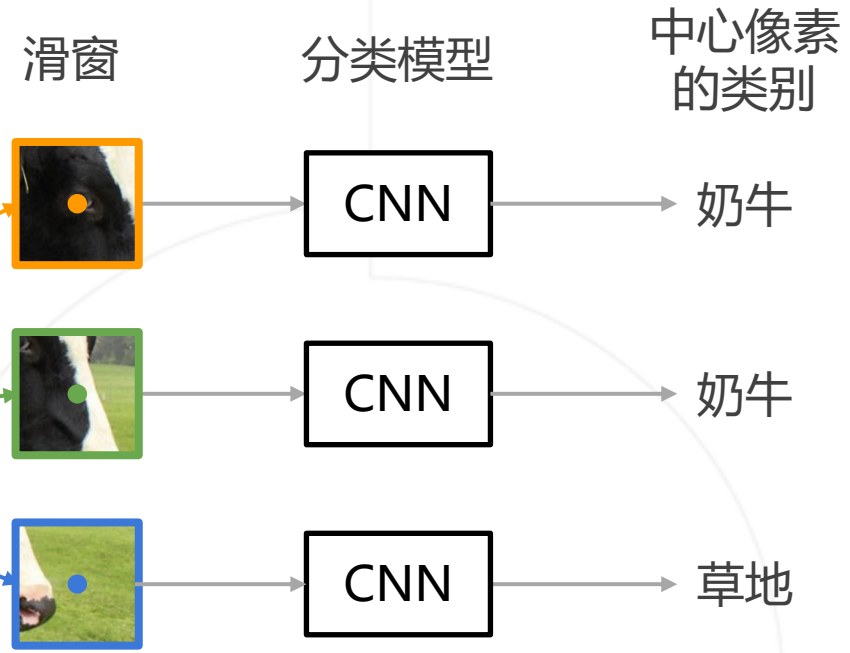
最终性能依赖初步分割结果

Farabet et al, “Learning Hierarchical Features for Scene Labeling,” TPAMI2013

基本思路：逐像素分类



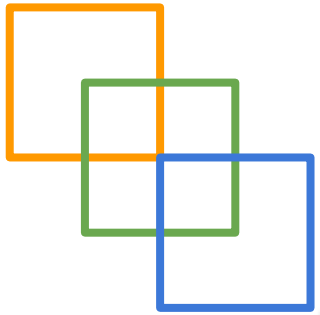




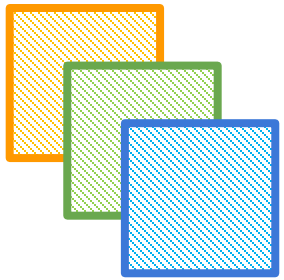
优势：可以充分利用已有的图像分类模型

问题：效率低下，重叠区域重复计算卷积

复用卷积计算

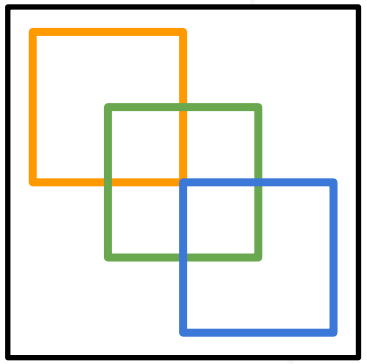


在滑窗上计算卷积

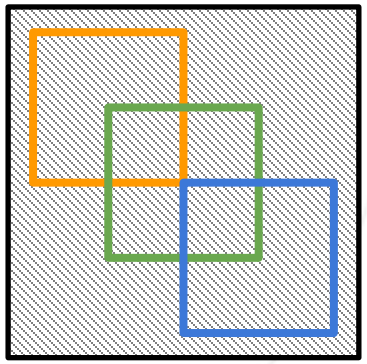


同样的卷积核

计算复用
结果相同

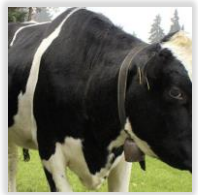


在原图上计算卷积

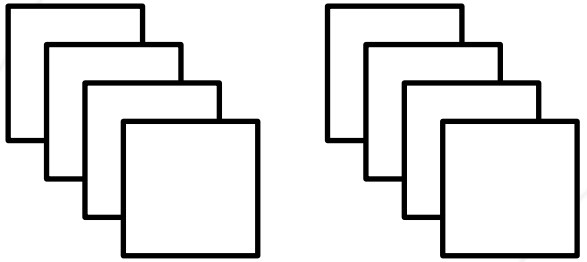


复用卷积计算

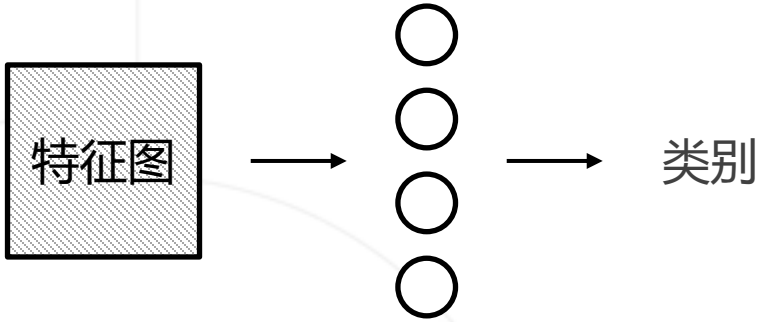
固定大小的区块



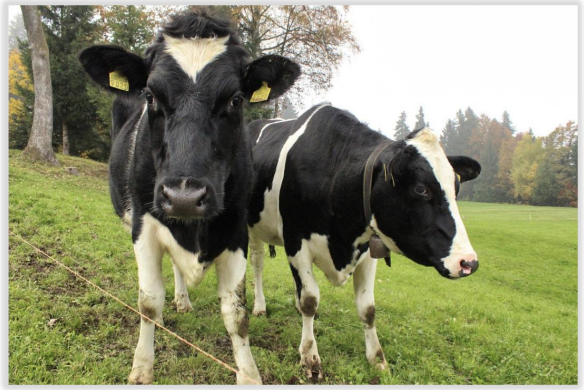
卷积层



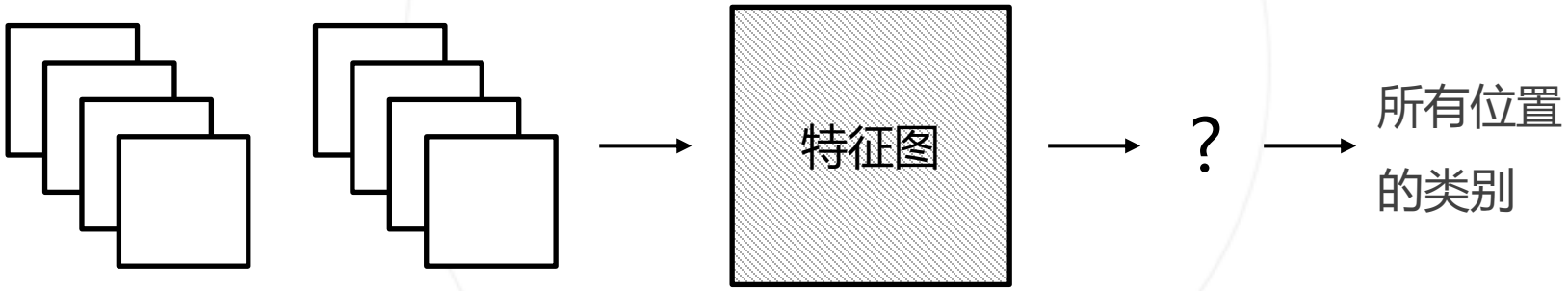
全连接层



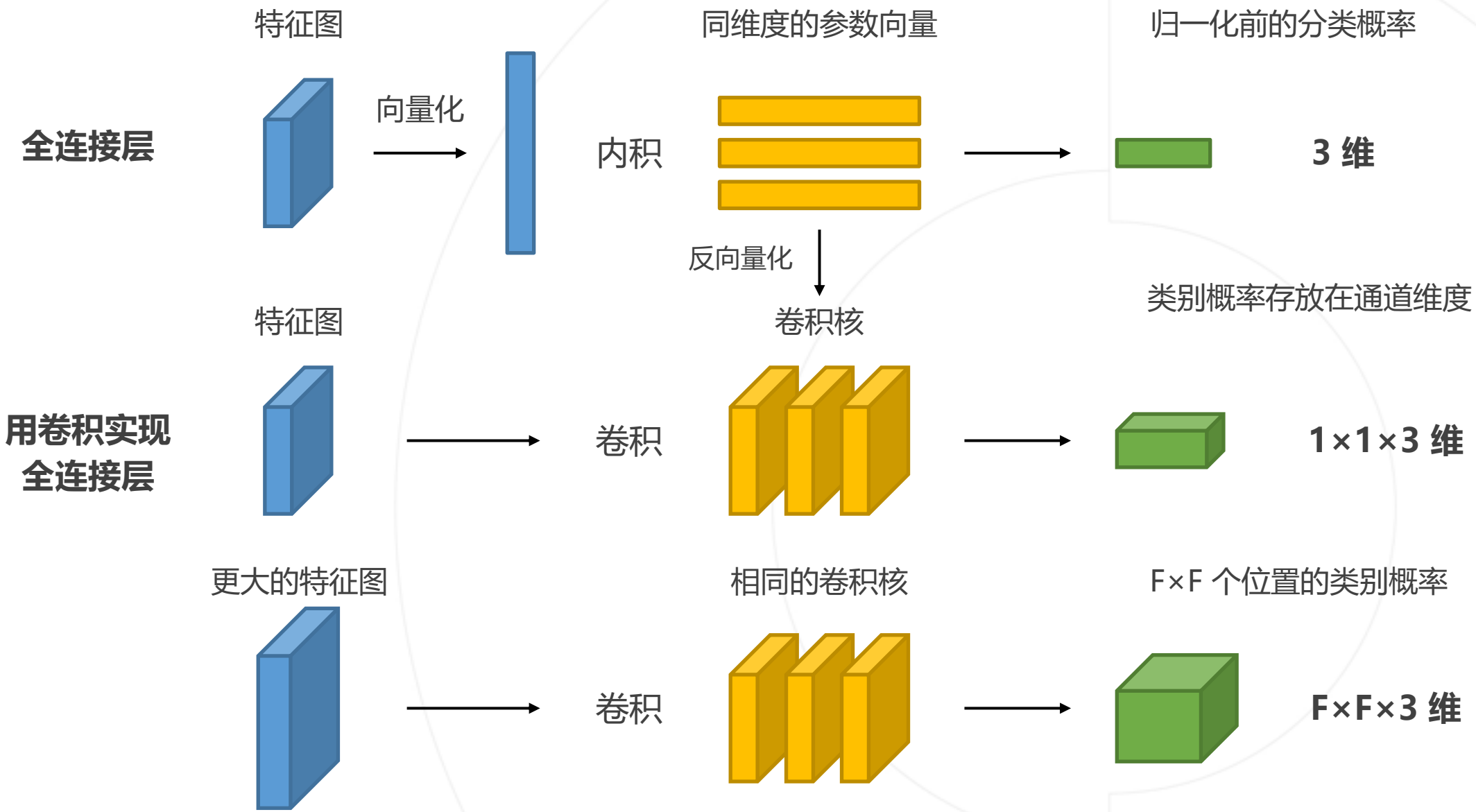
任意大小的原图



问题：全连接层要求固定输入大小



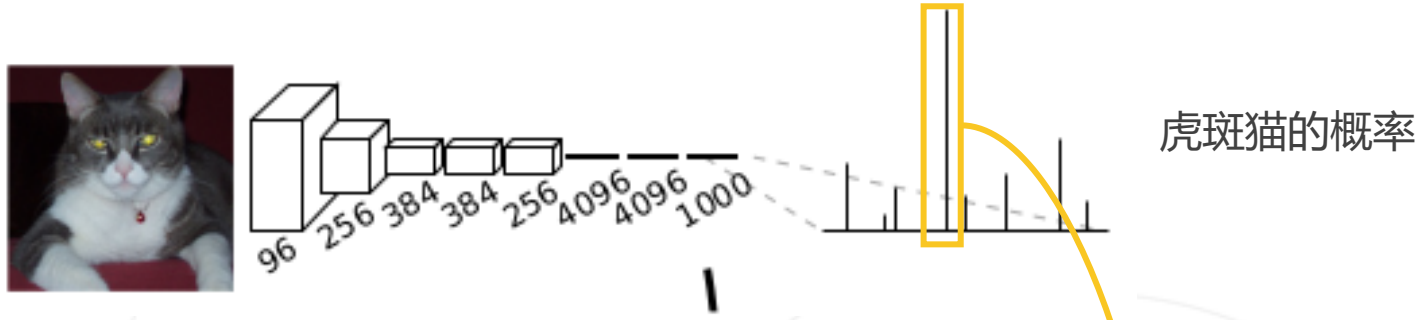
全连接层的卷积化


特征图
同维度的参数向量
归一化前的分类概率
全连接层
向量化
内积
3 维
反向量化
类别概率存放在通道维度
特征图
卷积核
用卷积实现
全连接层
卷积
1×1×3 维
更大的特征图
相同的卷积核
F×F 个位置的类别概率
卷积
F×F×3 维

全卷积网络 Fully Convolutional Network 2015

图像分类

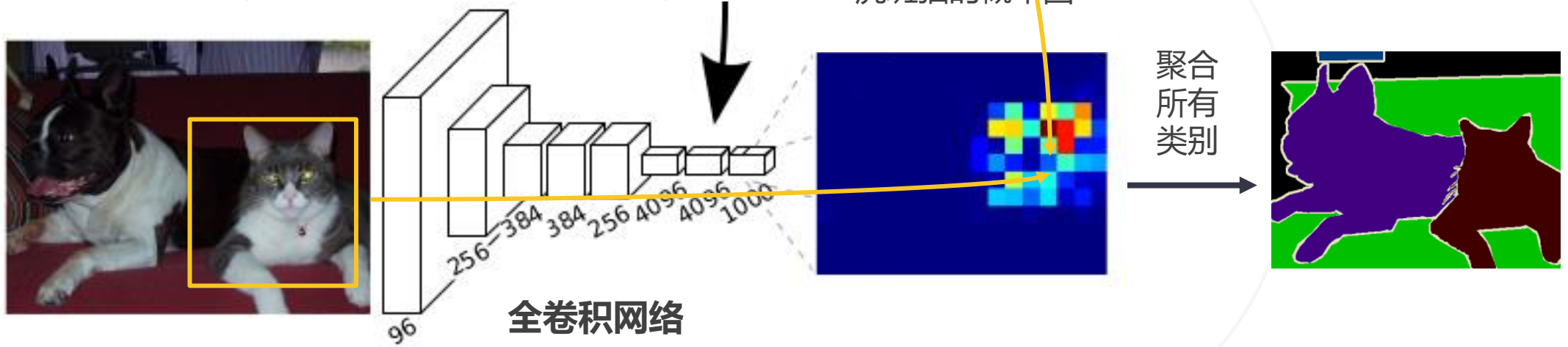
固定尺寸输入



全连接层
卷积化

语义分割

任意尺寸输入

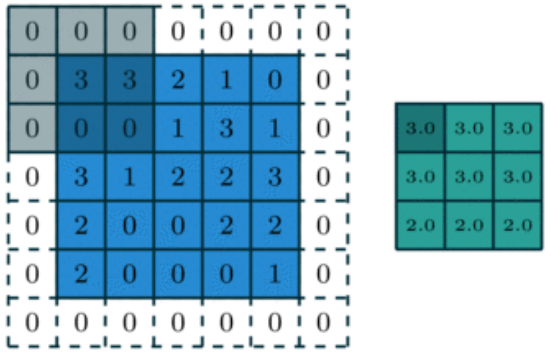


Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

预测图的升采样

问题：

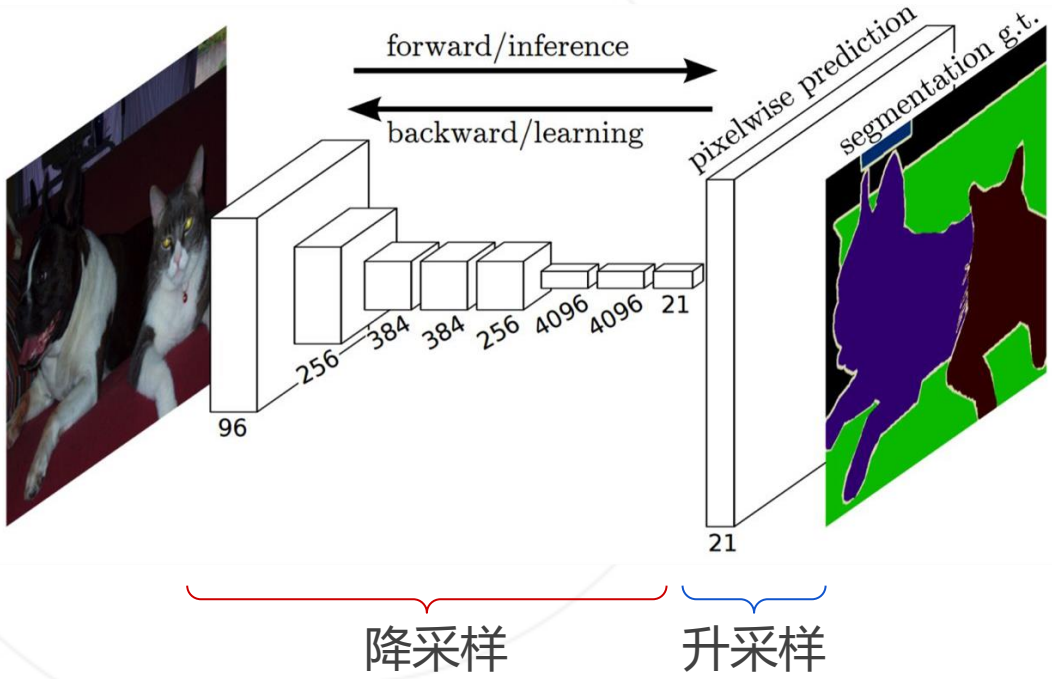
图像分类模型使用降采样层（步长卷积或池化）获得高层次特征，导致全卷积网络输出尺寸小于原图，而分割要求同尺寸输出



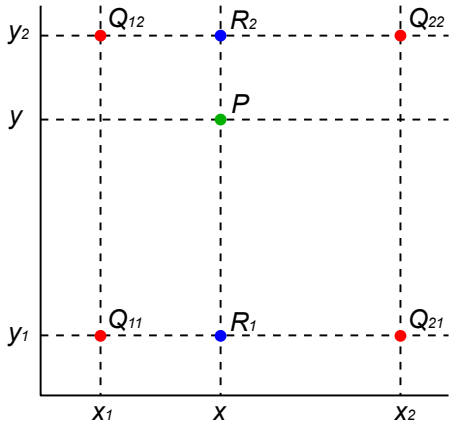
解决方法：

对预测的分割图升采样，恢复原图分辨率，升采样方案：

1. 双线性插值
2. 转置卷积：可学习的升采样层



双线性插值 Bilinear Interpolation



$$f(x,y) = \frac{1}{(x_2-x_1)(y_2-y_1)} \begin{bmatrix} x_2-x & x-x_1 \end{bmatrix} \begin{bmatrix} f(Q_{11}) & f(Q_{12}) \\ f(Q_{21}) & f(Q_{22}) \end{bmatrix} \begin{bmatrix} y_2-y \\ y-y_1 \end{bmatrix}.$$

1	2	3
4	5	6
7	8	9

1.0	1.5	2.0	2.5	3.0
2.5	3.0	3.5	4.0	4.5
4.0	4.5	5.0	5.5	6.0
5.5	6.0	6.5	7.0	7.5
7.0	7.5	8.0	8.5	9.0

卷积实现双线性插值

1	2	3
4	5	6
7	8	9

零插值

1.0		2.0		3.0
4.0		5.0		6.0
7.0		8.0		9.0

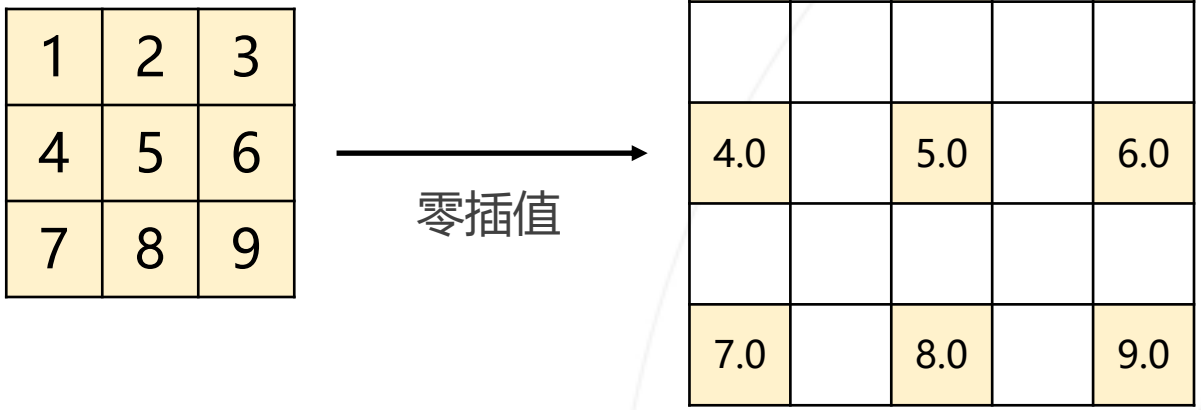
卷积

1.0	1.5	2.0	2.5	3.0
2.5	3.0	3.5	4.0	4.5
4.0	4.5	5.0	5.5	6.0
5.5	6.0	6.5	7.0	7.5
7.0	7.5	8.0	8.5	9.0

双线性插值对应的卷积核

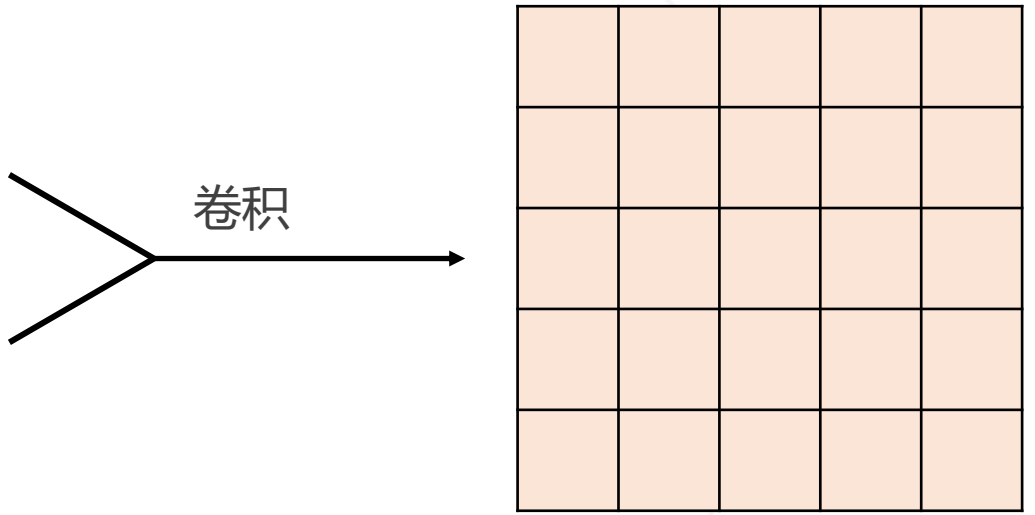
0.25	0.5	0.25
0.5	1	0.5
0.25	0.5	0.25

转置卷积 Transposed Convolution



别名：

- 升卷积 Upconvolution
- 反卷积 Deconvolution
  - **但数学上和卷积并不是逆运算！**

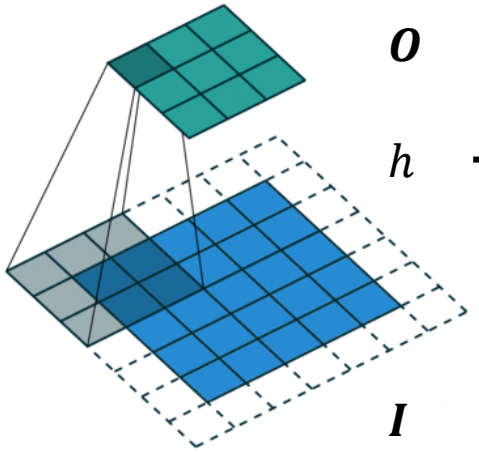


可学习的卷积核

a_{11}	a_{12}	a_{13}
a_{21}	a_{22}	a_{23}
a_{31}	a_{32}	a_{33}

转置卷积 Transposed Convolution

卷积



$$O = I * h$$

卷积核

$$h = \begin{bmatrix} w_{00} & w_{01} & w_{02} \\ w_{10} & w_{11} & w_{12} \\ w_{20} & w_{21} & w_{22} \end{bmatrix}$$

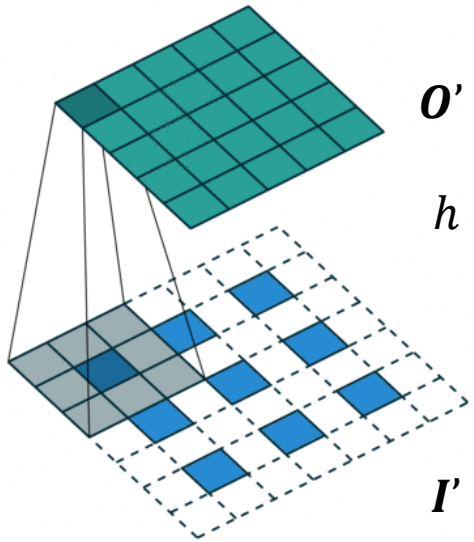
$$C = \begin{pmatrix} w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 & 0 & 0 & 0 & 0 \\ 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2.1} & w_{2.2} & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2} & & \end{pmatrix}$$

将卷积运算表示成矩阵乘法

$$O = I * h = C\,I$$

$$O' = I' \star h = C^{\mathrm{T}} I'$$

转置卷积

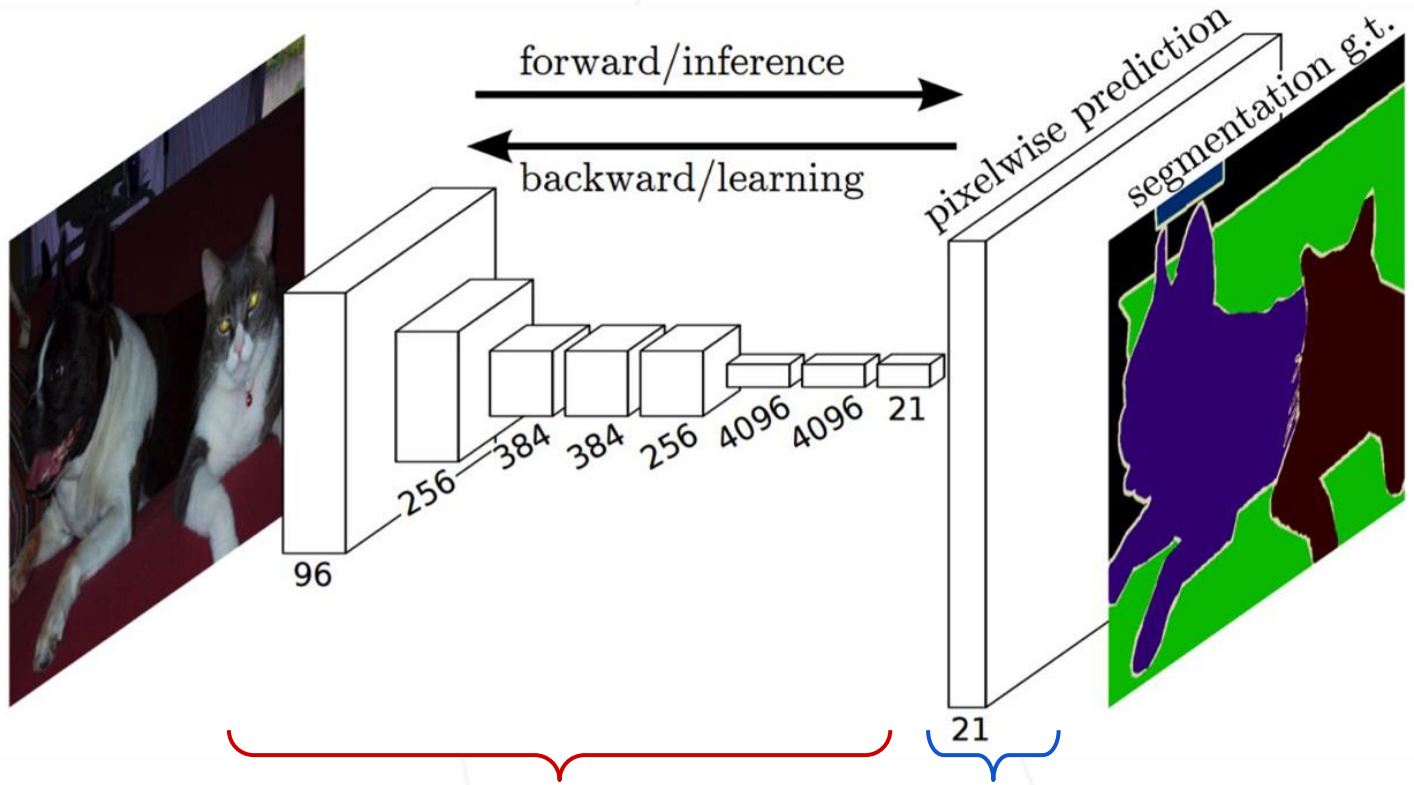


只在形状上有互逆的关系，数值结果没有互逆关系

全卷积网络的预测与训练

图像

每像素的
类别预测



卷积化改造的
图像分类网络

升采样层

逐像素计算交叉
熵损失并求和

训练损失

反传更新网络参数

端到端学习的模型

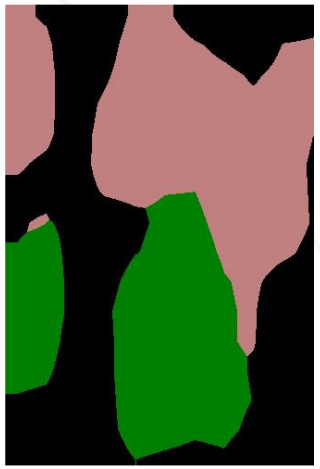
Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

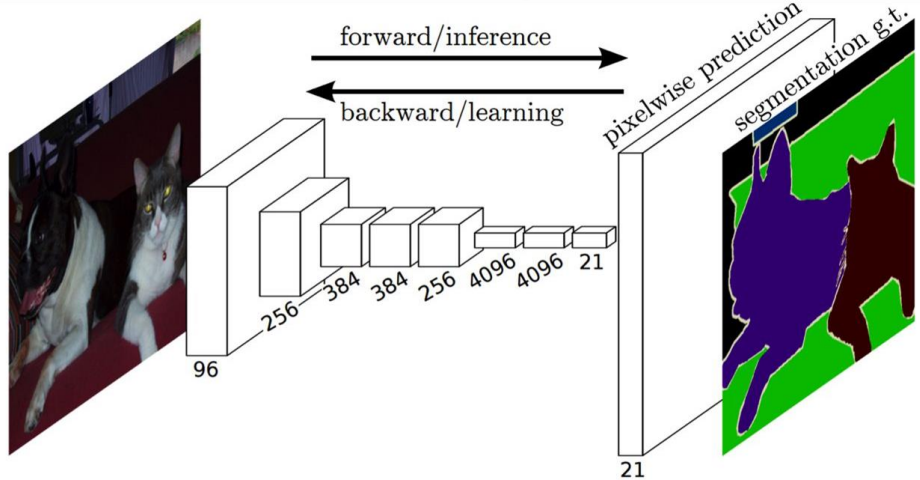
基于多层级特征的上采样

问题：基于顶层特征预测，再升采样 32 倍得到的预测图较为粗糙。

分析：高层特征经过多次降采样，细节丢失严重。

解决思路：结合低层次和高层次特征图。





细节信息丰富
语义信息贫乏

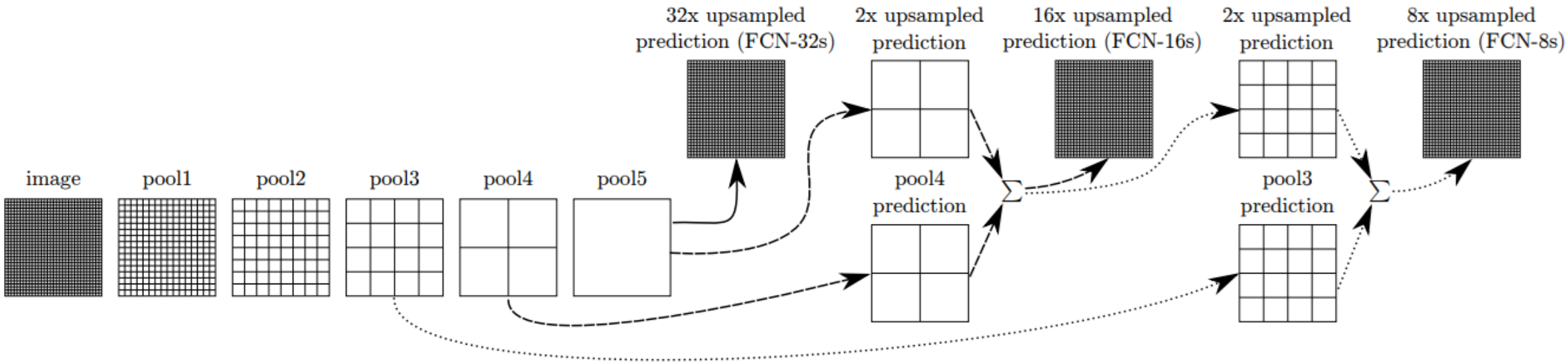
语义信息丰富
细节信息贫乏

Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

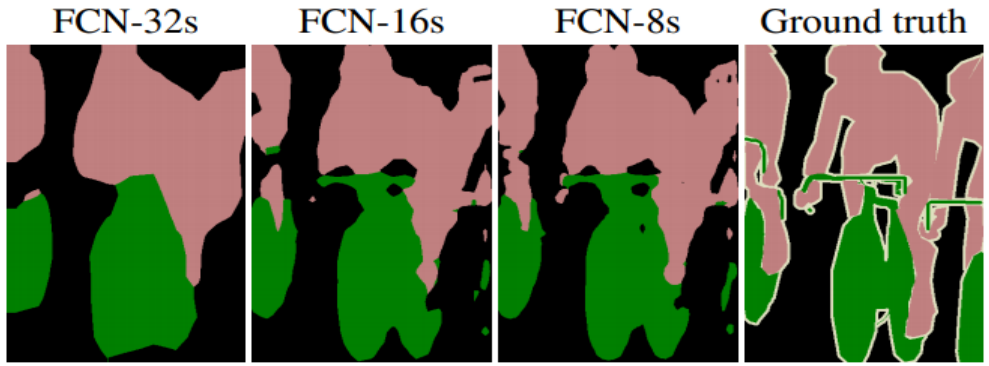
基于多层级特征的上采样

解决方案 FCN：

基于低层次和高层次特征图分别产生类别预测，升采样到原图大小，再平均得到最终结果



细节信息丰富
语义信息贫乏

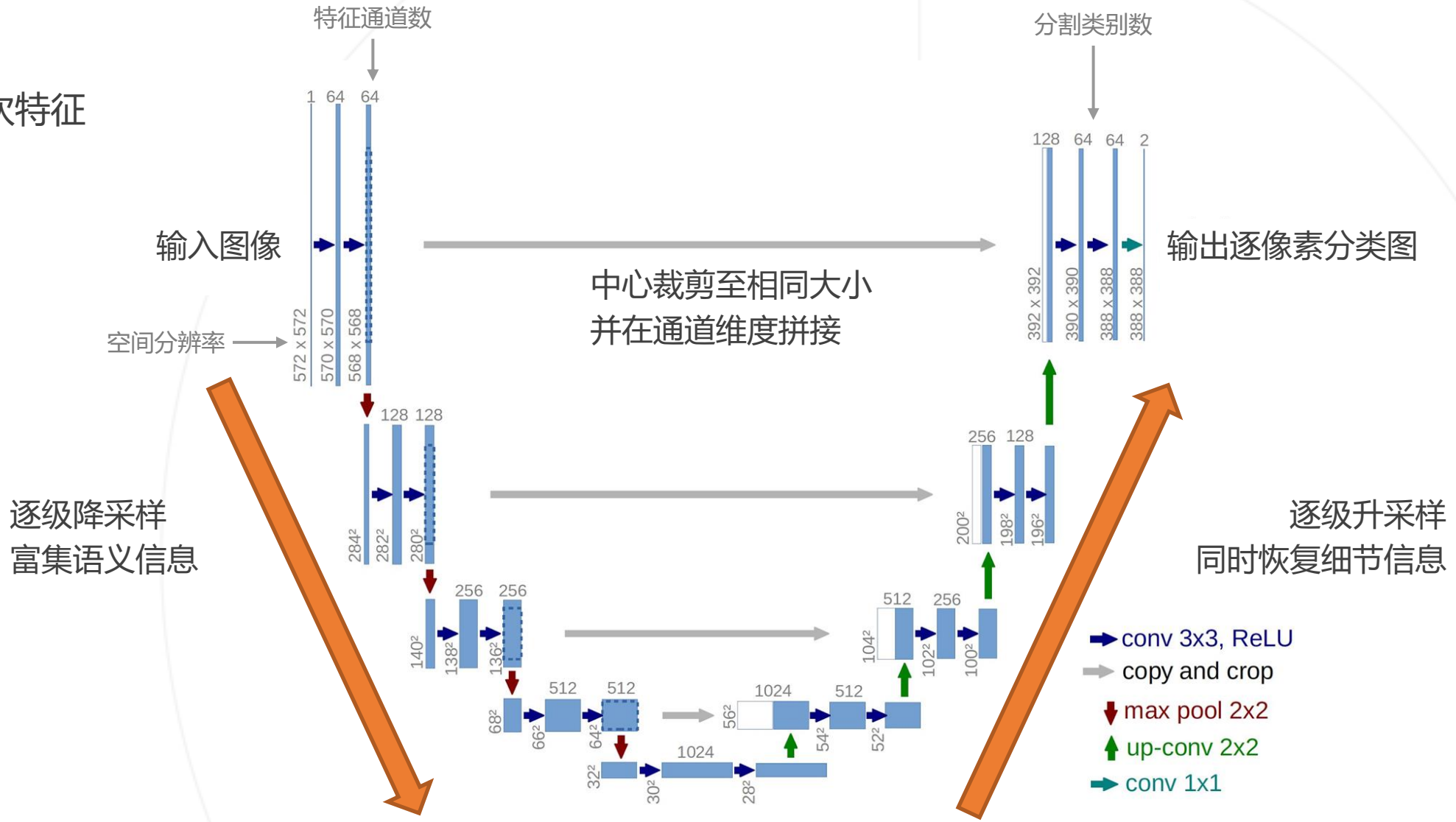


Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

UNet 2015

解决方案 UNet：

逐级融合高低层次特征



Ronneberger et al, “U-Net: Convolutional Networks for Biomedical Image Segmentation”, MICCAI 2015

上下文信息

有歧义的区域

船?

汽车?

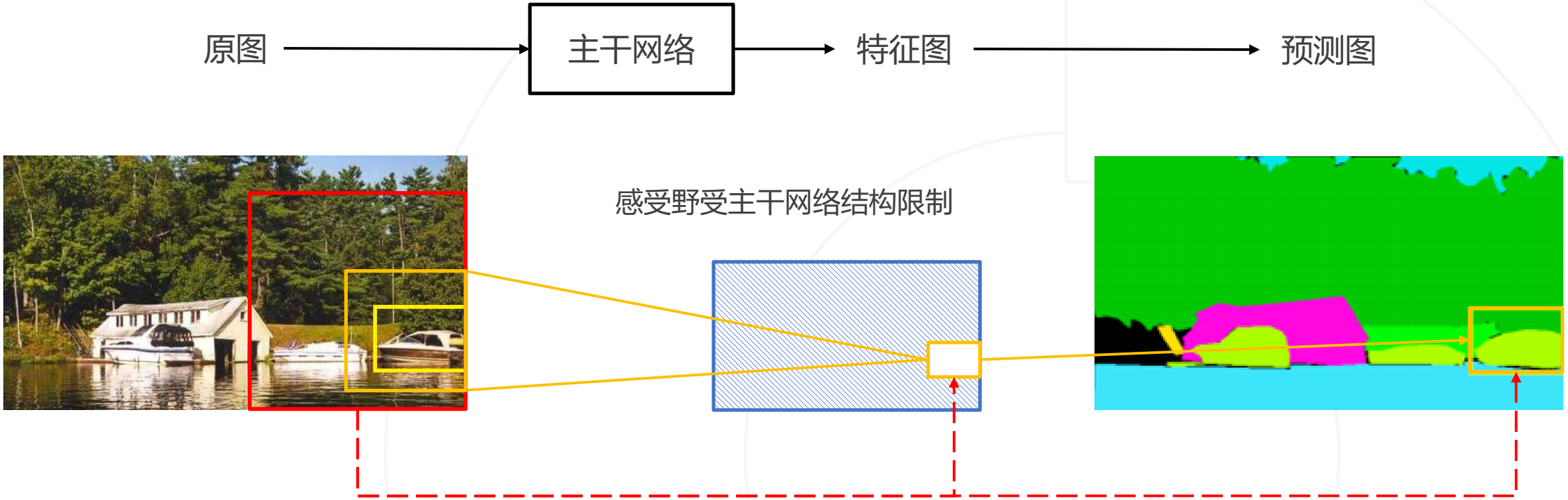
枕头?

被子?

上下文的重要性

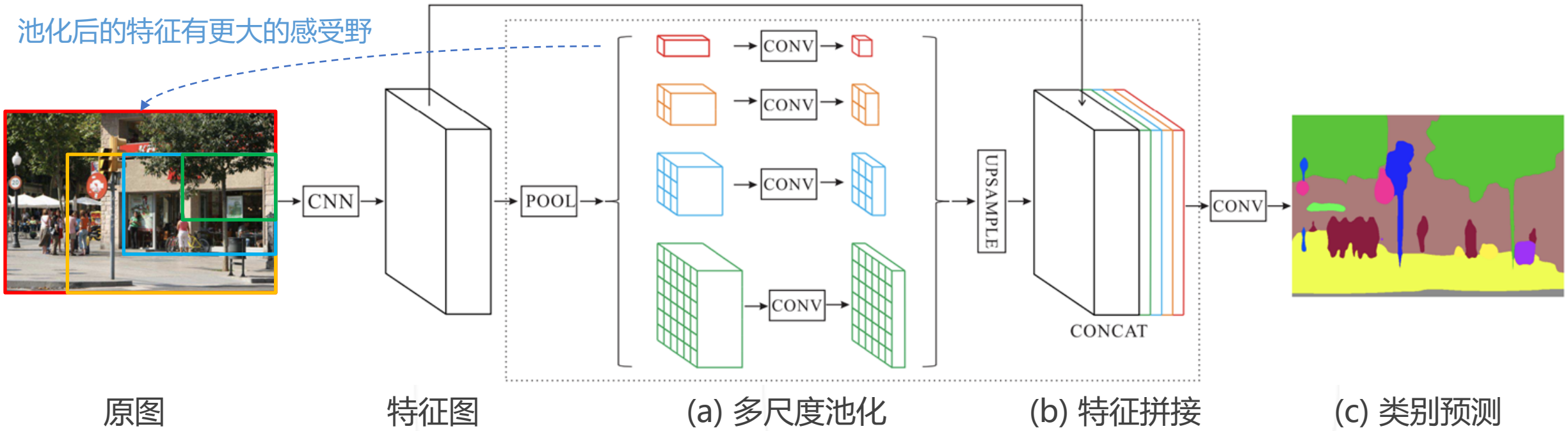
图像周围的内容（也称上下文）可以帮助我们做出更准确的判断。

获取上下文信息



方案：增加感受野更大的网络分支，
将上下文信息导入局部预测中

PSPNet 2016



原图　　特征图　　(a) 多尺度池化　　(b) 特征拼接　　(c) 类别预测

(a) 对特征图进行不同尺度的池化，得到不同尺度的上下文特征

(b) 上下文特征经过通道压缩和空间上采样之后拼接回原特征图 → 同时包含局部和上下文特征

(c) 基于融合的特征产生预测图

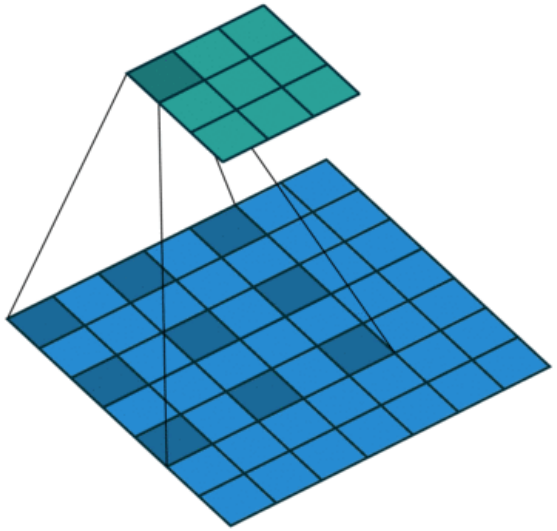
空洞卷积与 DeepLab 系列算法

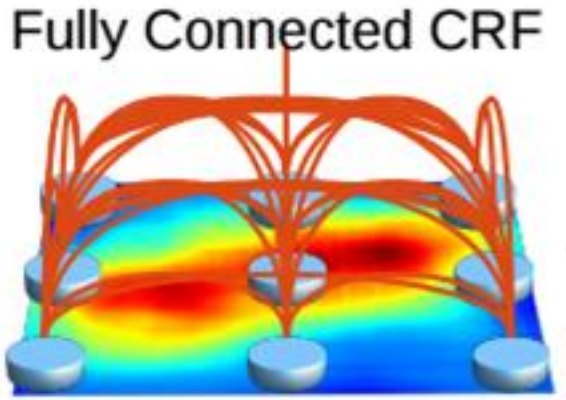
DeepLab 系列

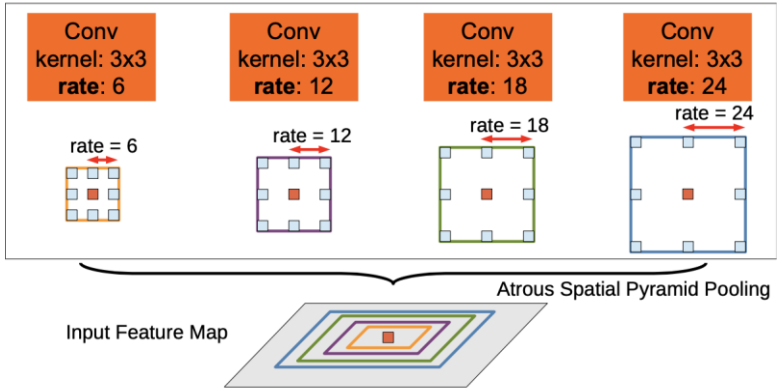
DeepLab 是语义分割的又一系列工作，其主要贡献为：

- 使用空洞卷积解决网络中的下采样问题
- 使用条件随机场 CRF 作为后处理手段，精细化分割图
- 使用多尺度的空洞卷积（ASPP 模块）捕捉上下文信息

DeepLab v1 发表于 2014 年，后于 2016、2017、2018 年提出 v2、v3、v3+ 版本。







空洞卷积解决下采样问题

图像分类模型中的下采样层使输出尺寸变小

如果将池化层和卷积中的步长去掉：

- 可以减少下采样的次数；
- 特征图就会变大，需要对应增大卷积核，以维持相同的感受野，但会增加大量参数
- 使用空洞卷积（Dilated Convolution/Atrous Convolution），在不增加参数的情况下增大感受野

空洞卷积解决下采样问题

标准卷积

特征图

1.0		2.0		3.0
4.0		5.0		6.0
7.0		8.0		9.0

下采样

1.0	2.0	3.0
4.0	5.0	6.0
7.0	8.0	9.0

卷积运算

卷积核

a_{11}	a_{12}	a_{13}
a_{21}	a_{22}	a_{23}
a_{31}	a_{32}	a_{33}

结果

空洞卷积

1.0		2.0		3.0
4.0		5.0		6.0
7.0		8.0		9.0

特征图不变
膨胀卷积核

再进行卷积运算

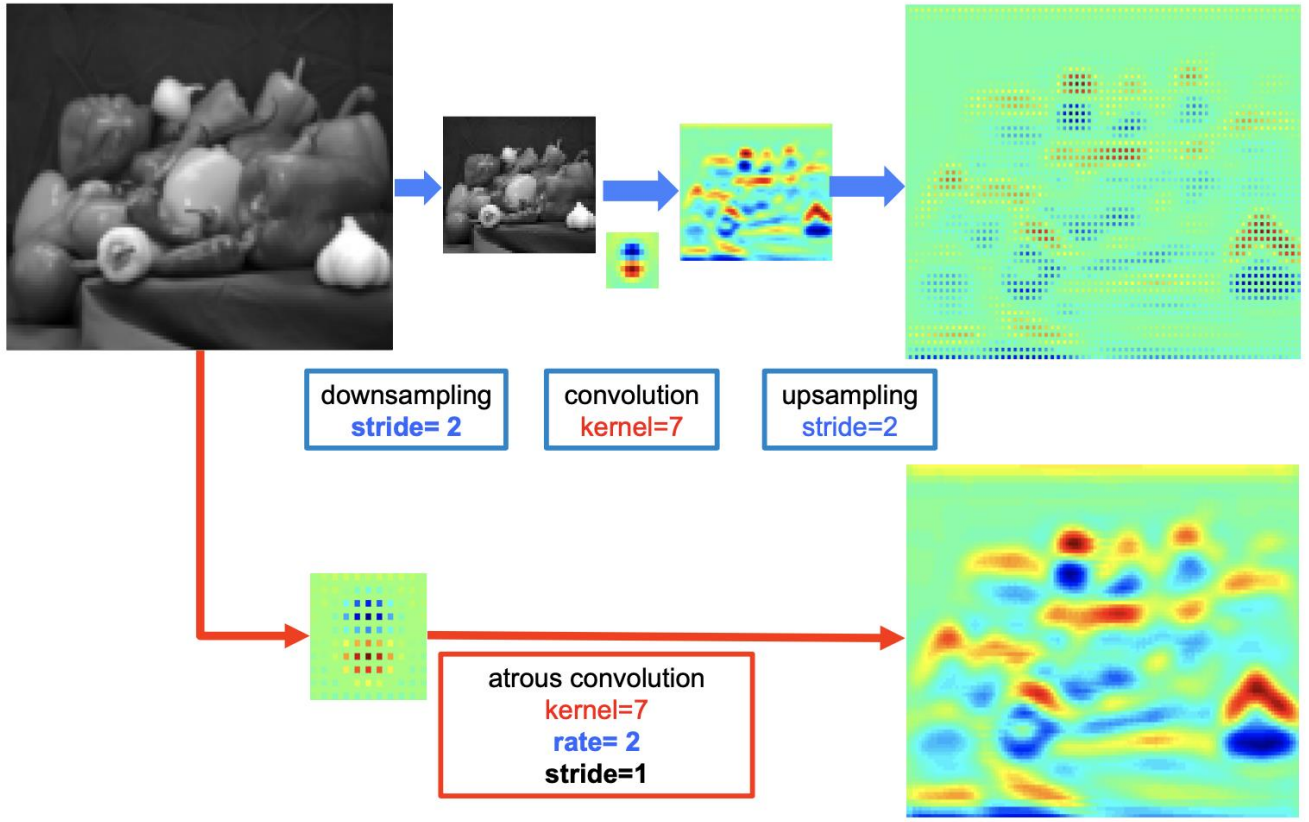
膨胀卷积核
不产生额外参数

a_{11}		a_{12}		a_{13}
a_{21}		a_{22}		a_{23}
a_{31}		a_{32}		a_{33}

相同的计算结果

下采样加标准卷积
等价于空洞卷积

空洞卷积和下采样



使用升采样方案得到的特征图只有原图 1/4 位置的响应，需要配合插值

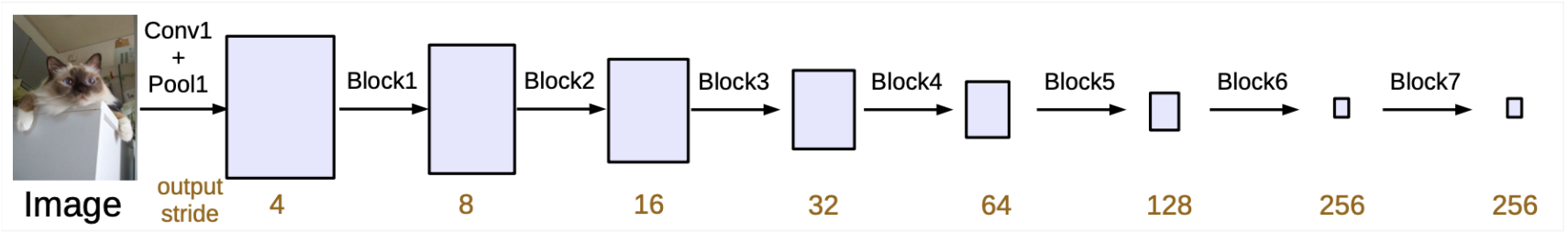
使用空洞卷积可以得到相同分辨率的特征图，且无需额外插值操作

rate=膨胀倍率，即参数之间的距离

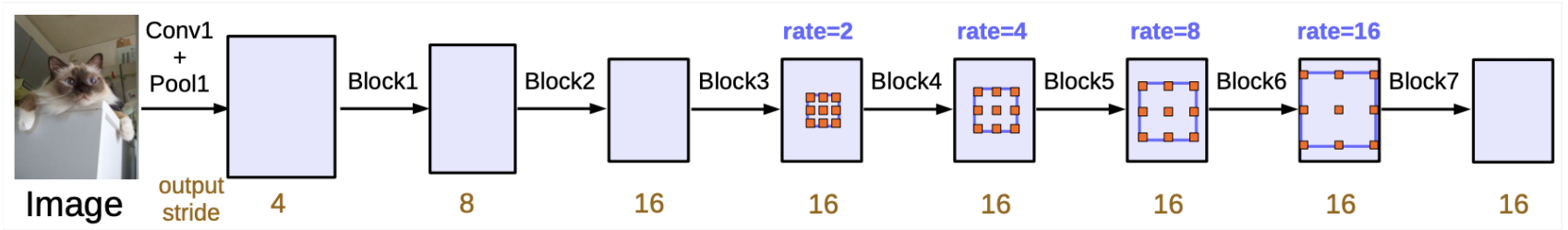
DeepLab 模型

DeepLab 在图像分类网络的基础上做了修改：

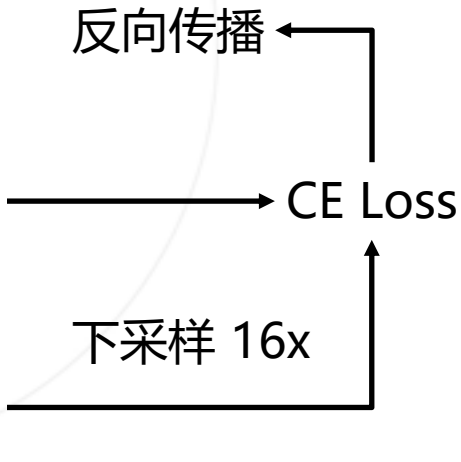
- 去除分类模型中的后半部分的下采样层
- 后续的卷积层改为膨胀卷积，并且逐步增加rate来维持原网络的感受野



含有降采样的卷积网络



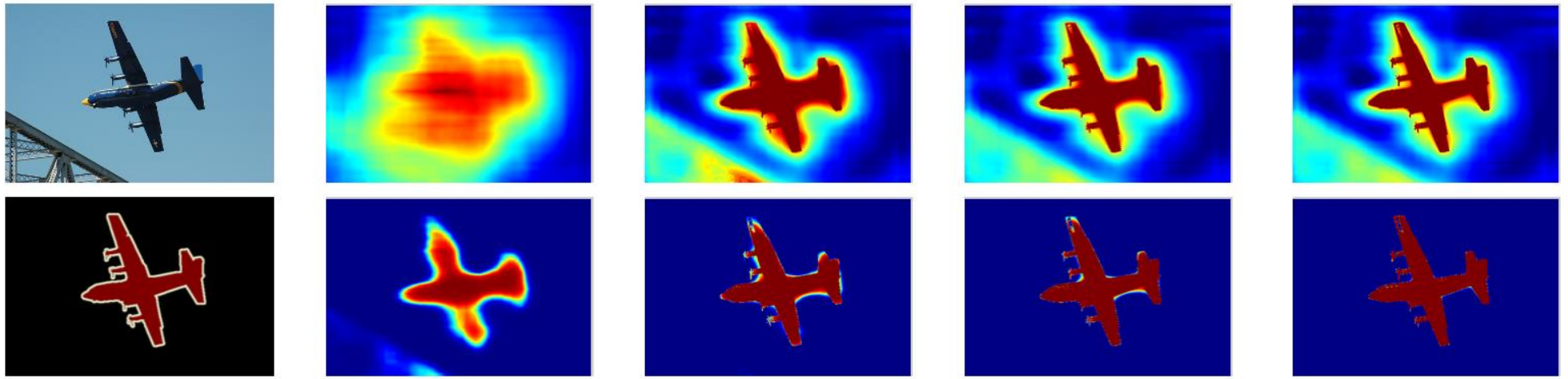
去除降采样并使用空洞卷积的网络



条件随机场 Conditional Random Field, CRF

模型直接输出的分割图较为粗糙，尤其在物体边界处不能产生很好的分割结果。

DeepLab v1&v2 使用条件随机场 (CRF) 作为后处理手段，结合原图颜色信息和神经网络预测的类别得到精细化分割结果。



条件随机场

CRF 是一种概率模型。DeepLab 使用 CRF 对分割结果进行建模，用能量函数用来表示分割结果优劣，通过最小化能量函数获得更好的分割结果。

能量函数 $$E(\boldsymbol{x}) = \sum_i \theta_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j)$$

x_i, x_j	特定像素的预测结果（向量化后只有1维坐标）
$\boldsymbol{x}$	全部像素的预测结果
$\theta_i(x_i)$	单个预测对能量函数的贡献
$\theta_{i,j}(x_i, x_j)$	一对预测对能量函数的贡献

能量函数的意义

能量函数 $$E(\boldsymbol{x}) = \sum_i \theta_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j)$$

鼓励后处理结果符合网络给出的结果

$$\theta_i(x_i) = -\log P(x_i)$$

$P(x_i)$ = 网络输出的对应类别的概率

鼓励产生更好的分割边界

$$\theta_{ij}(x_i, x_j) = \mu(x_i, x_j)\left[w_1 \exp\left(-\frac{||p_i - p_j||^2}{2\sigma_\alpha^2} - \frac{||I_i - I_j||^2}{2\sigma_\beta^2}\right) + w_2 \exp\left(-\frac{||p_i - p_j||^2}{2\sigma_\gamma^2}\right)\right]$$

仅当类别不同时产生惩罚 $\mu(x_i, x_j) = [x_i \neq x_j]$

位置相近且颜色相近时惩罚

鼓励仅在原图颜色边界处产生产生类别变化

位置相近时惩罚

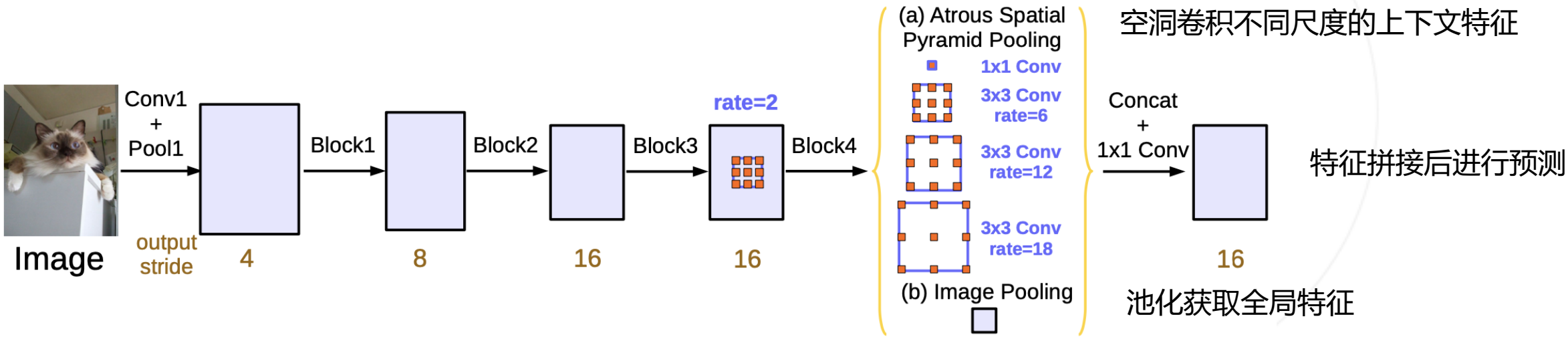
鼓励产生平滑的结果

空间金字塔池化 Atrous Spatial Pyramid Pooling ASPP

PSPNet 使用不同尺度的池化来获取不同尺度的上下文信息

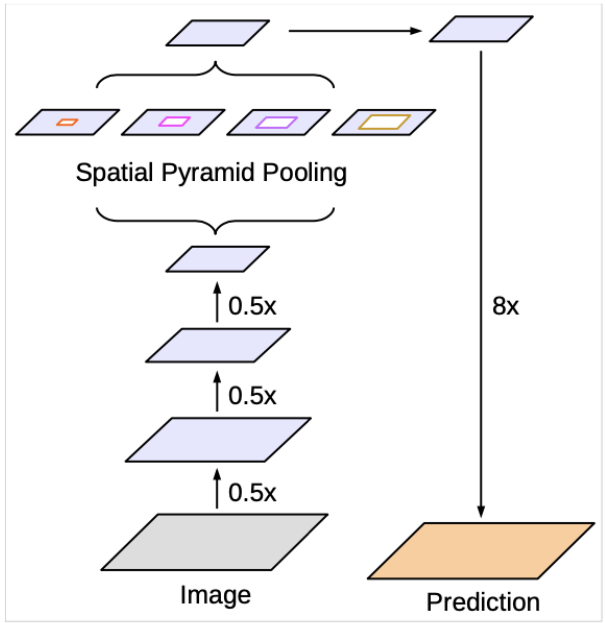
DeepLab v2 & v3 使用不同尺度的空洞卷积达到类似的效果

更大膨胀率的空洞卷积 ⟶ 更大的感受野 ⟶ 更多的上下文特征

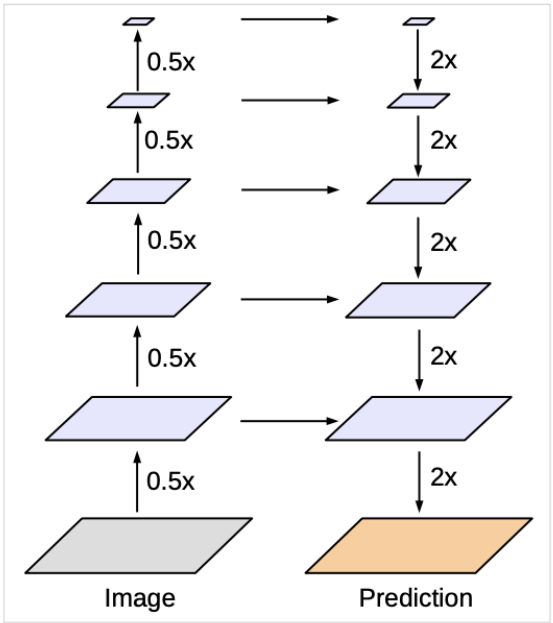


DeepLab v3+

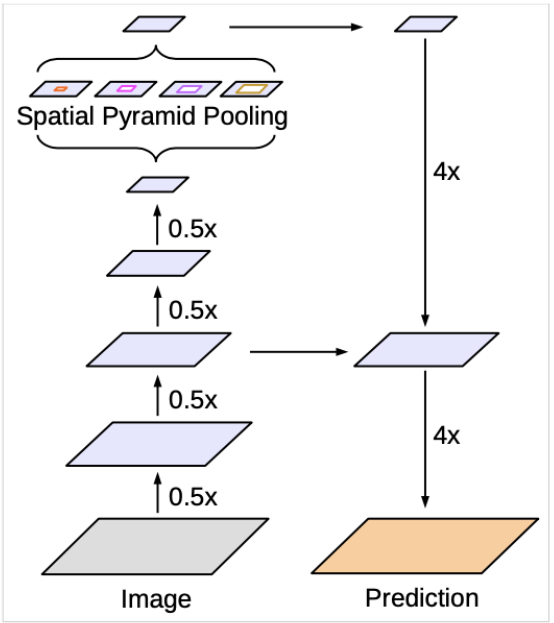
- DeepLab v2 / v3 模型使用 ASPP 捕捉上下文特征
- Encoder / Decoder 结构（如 UNet） 在上采样过程中融入低层次的特征图，以获得更精细的分割图
- DeepLab v3+ 将两种思路融合，在原有模型结构上增加了一个简单的 decoder 结构



(a) ASPP
DeepLab v2 / v3

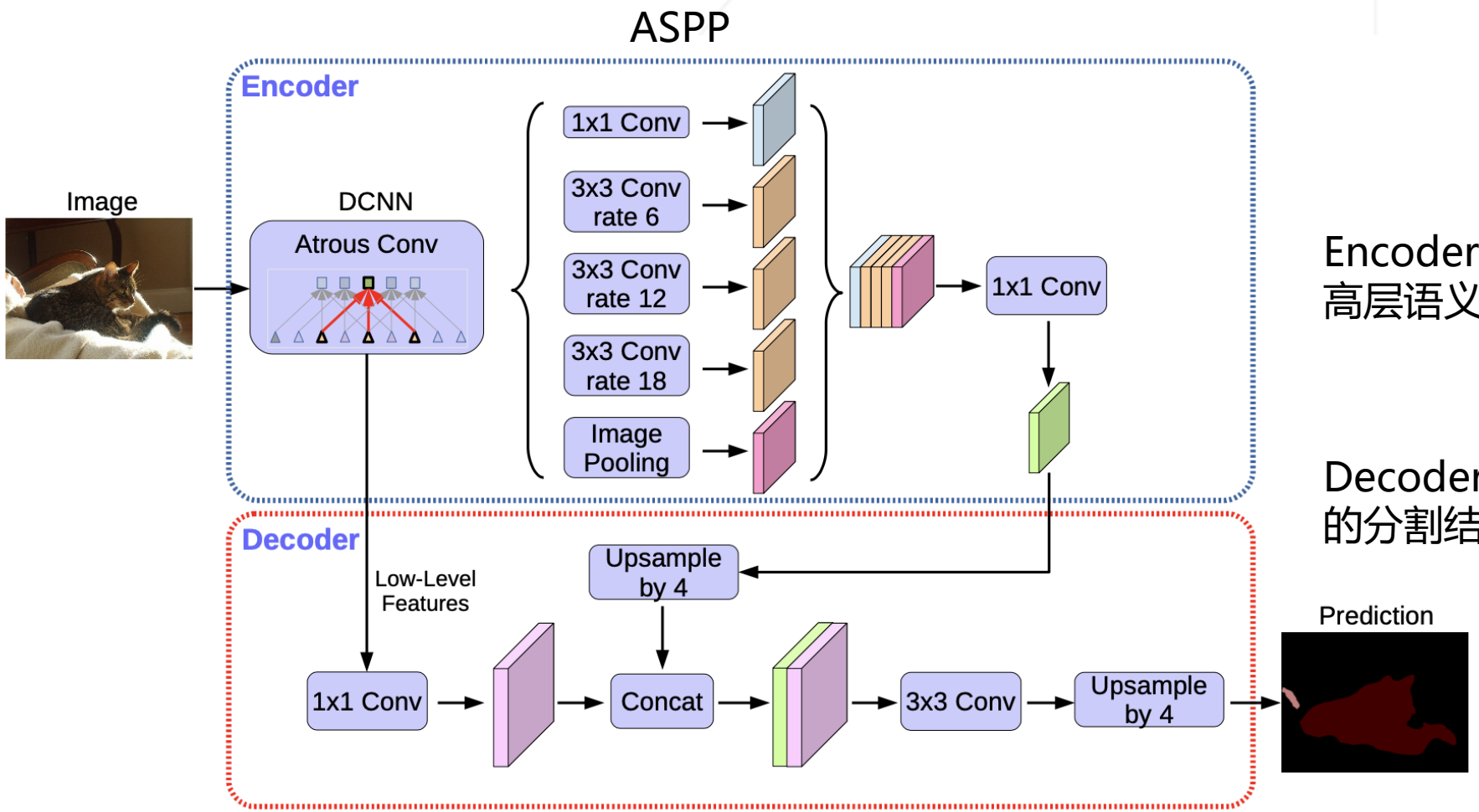


(b) Encoder / Decoder
UNet



(c) DeepLab v3+

DeepLab v3+



Encoder 通过 ASPP 产生多尺度的高层语义信息

Decoder 主要融合低层特征产生惊喜的分割结果

语义分割算法总结

基本思想 滑窗预测 → 全卷积网络 FCN

恢复网络中的降采样

朴素思路
双线性插值 → 可学习的升采样
转置卷积

FCN | UNet

更大感受野
空洞卷积

DeepLab 系列

恢复预测图的细节

分别基于高低层特征预测

FCN

融合高低层特征再预测

UNet | DeepLab v3+

CRF 后处理

DeepLab v1/v2

利用上下文信息

池化金字塔

PSPNet

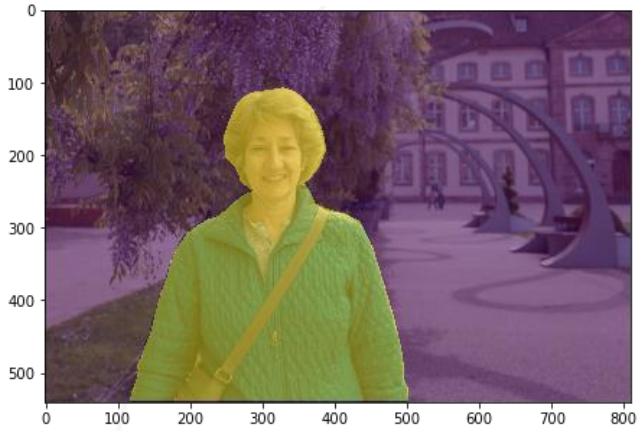
多尺度空洞卷积 ASPP

DeepLab v3/v3+

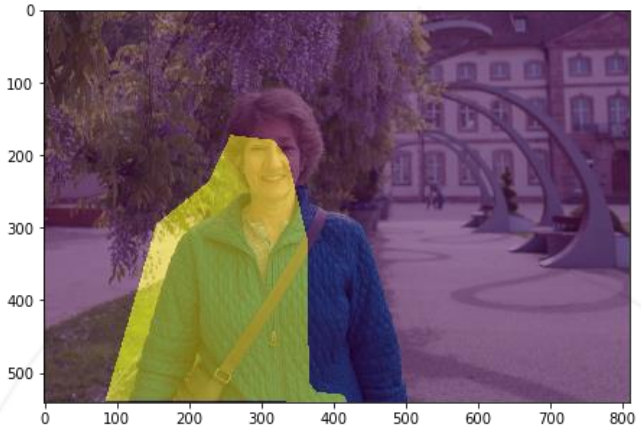
语义分割模型的评估

比较预测与真值

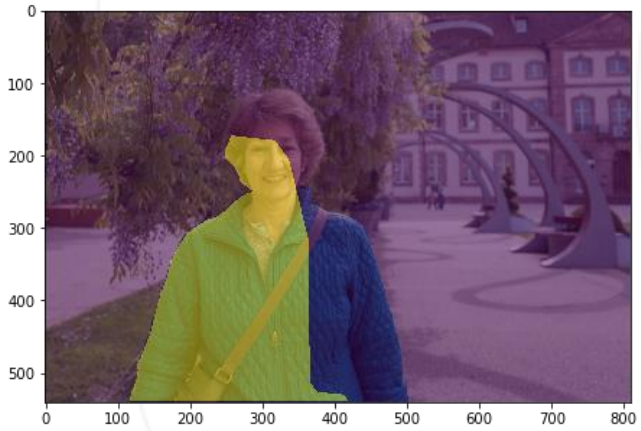
真实分割图



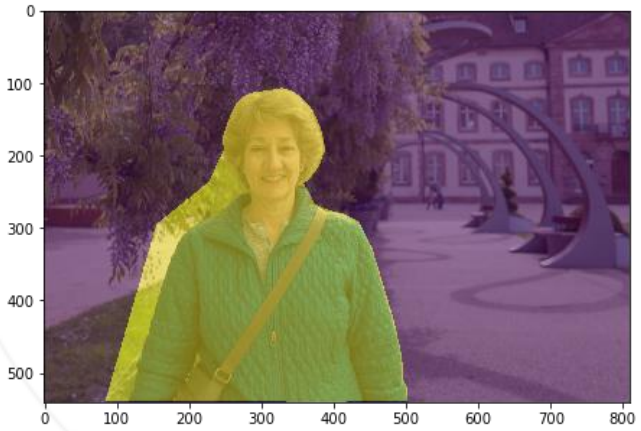
预测分割图



交集



并集



图片来源：https://www.jeremyjordan.me/evaluating-image-segmentation-models/

基于交并集的评估指标

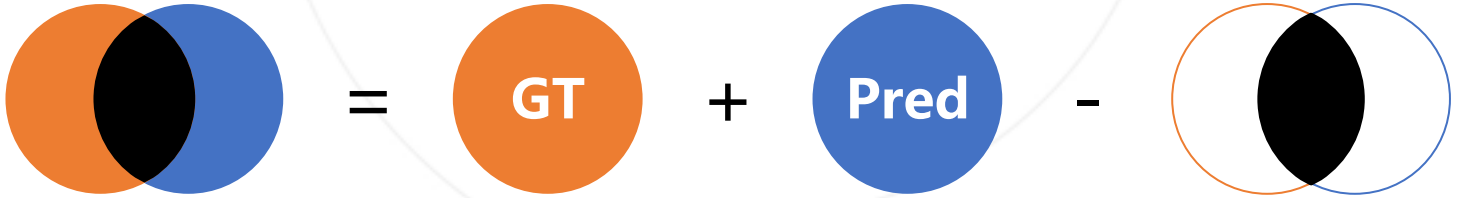
$$\text{Accuracy} = \frac{\text{面积比值}}{\text{GT}}$$

$$\text{IoU} = \frac{\text{面积比值}}{\text{GT} \cup \text{Pred}}$$

$$\text{Dice} = \frac{2 \times \text{面积比值}}{\text{GT} + \text{Pred}}$$

mAcc
mIoU
mDice
= 对每类计算指标再按类别平均

并集面积的计算方法



通用视觉框架OpenMMLab

实践5 语义分割工具包 MMSegmentation

内容概要

本节内容：

- MMSegmentation 项目概述
- MMSegmentation 的模块化设计
  - PSPNet 模型配置文件解读
- 数据集与数据流水线配置解读
- 常用优化器配置
- 代码实践
  - 使用预训练模型对单张图像进行推理
  - 使用自定义数据集训练语义分割模型

语义分割工具包 MMSegmentation

MMSegmentation

算法丰富

- 378 个预训练模型
- 27 篇论文复现

模块化设计

- 配置简便
- 容易拓展

统一超参

- 大量消融实验
- 支持公平对比

使用方便

- 训练工具
- 测试工具
- 推理 API

代码库：https://github.com/open-mmlab/mmsegmentation

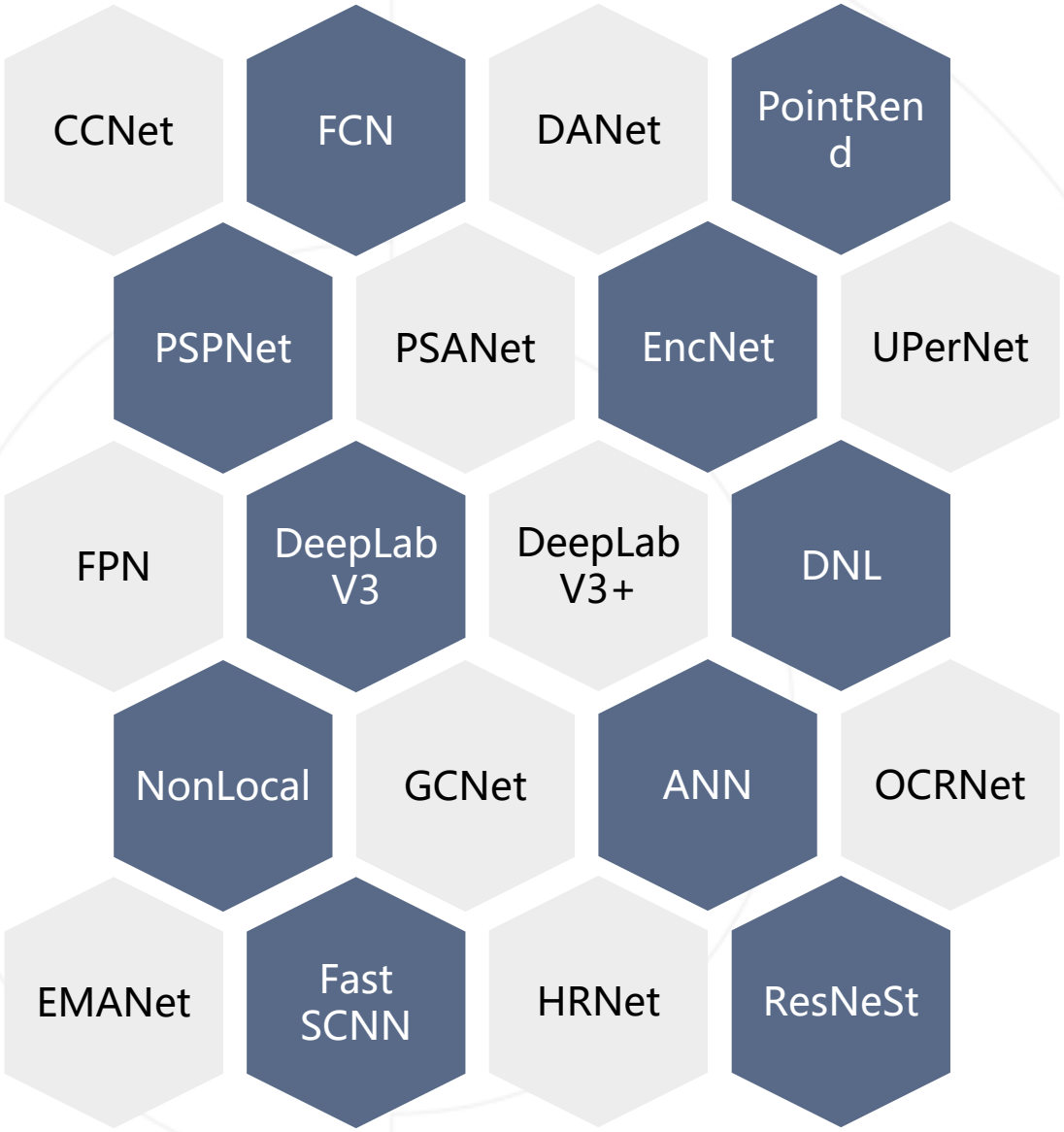
文档：https://mmsegmentation.readthedocs.io/en/latest/

算法丰富

OpenMMLab

20+ 算法

370+ 预训练模型

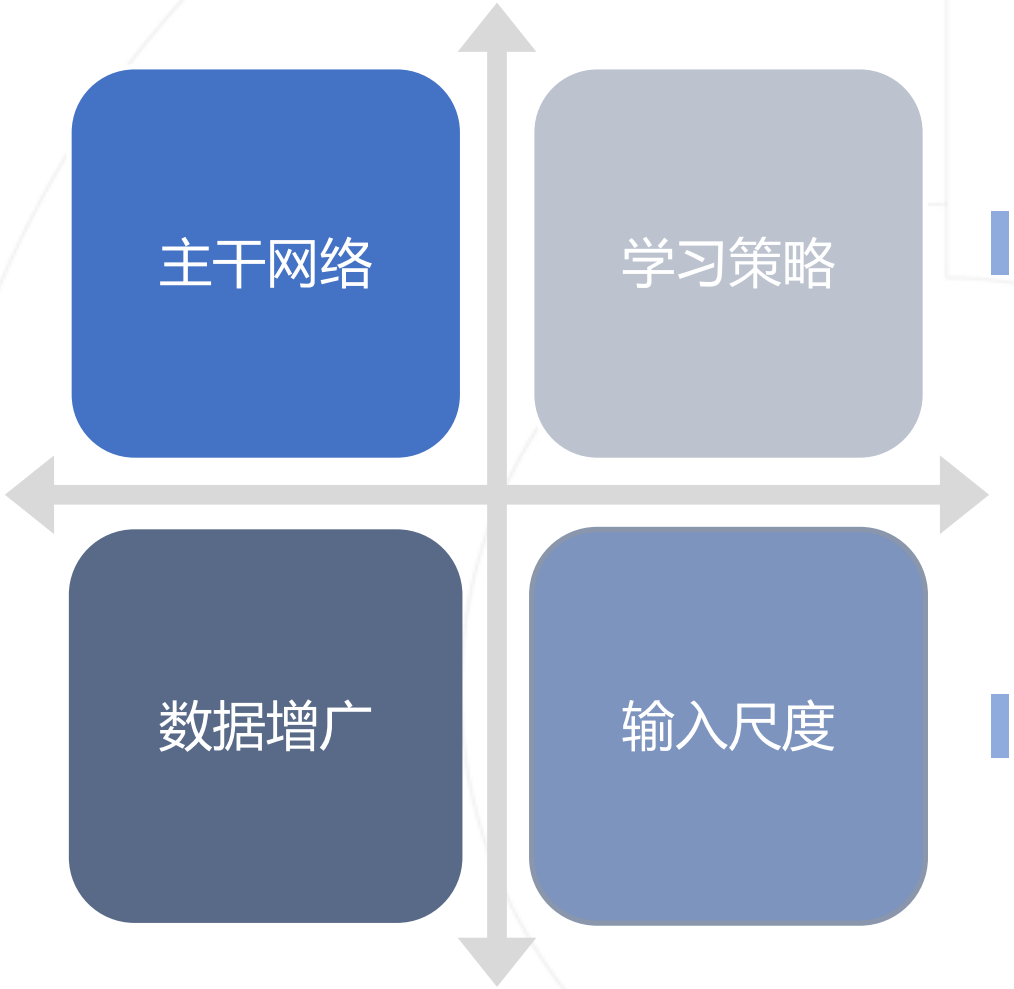


统一超参

OpenMMLab

	论文 A	论文 B	论文 C
1 骨干网络	• 标准 ResNet • PyTorch 官方预训练	• ResNet 变体 • 第三方预训练	• ResNet 变体 • 作者预训练
2 预处理	• 769x769 crop • 水平翻转	• 769x769 crop • 水平翻转 • 颜色扰动	• 512x1024 crop • 水平翻转 • 颜色扰动
3 核心算法	• 算法 A	• 算法 B	• 算法 C
4 训练策略	• 训练200 epoch • 初始学习率0.1 • Poly 学习率下降	• 训练120 iteration • 初始学习率0.01 • Step 学习率下降	• 训练100 epoch • 初始学习率0.01 • Poly 学习率下降
5 测试方式	• 多尺度 patch 测试	• 单尺度patch 测试	• 单尺度全图测试
	mIoU(A)	> mIoU(B)	> mIoU(C)

统一超参



预训练网络

空洞卷积

初始学习率

学习率下降策略

尺度放缩

区域剪裁

颜色扰动

水平翻转

Cityscapes: 769x769 & 512x1024

ADE20k: 512x512

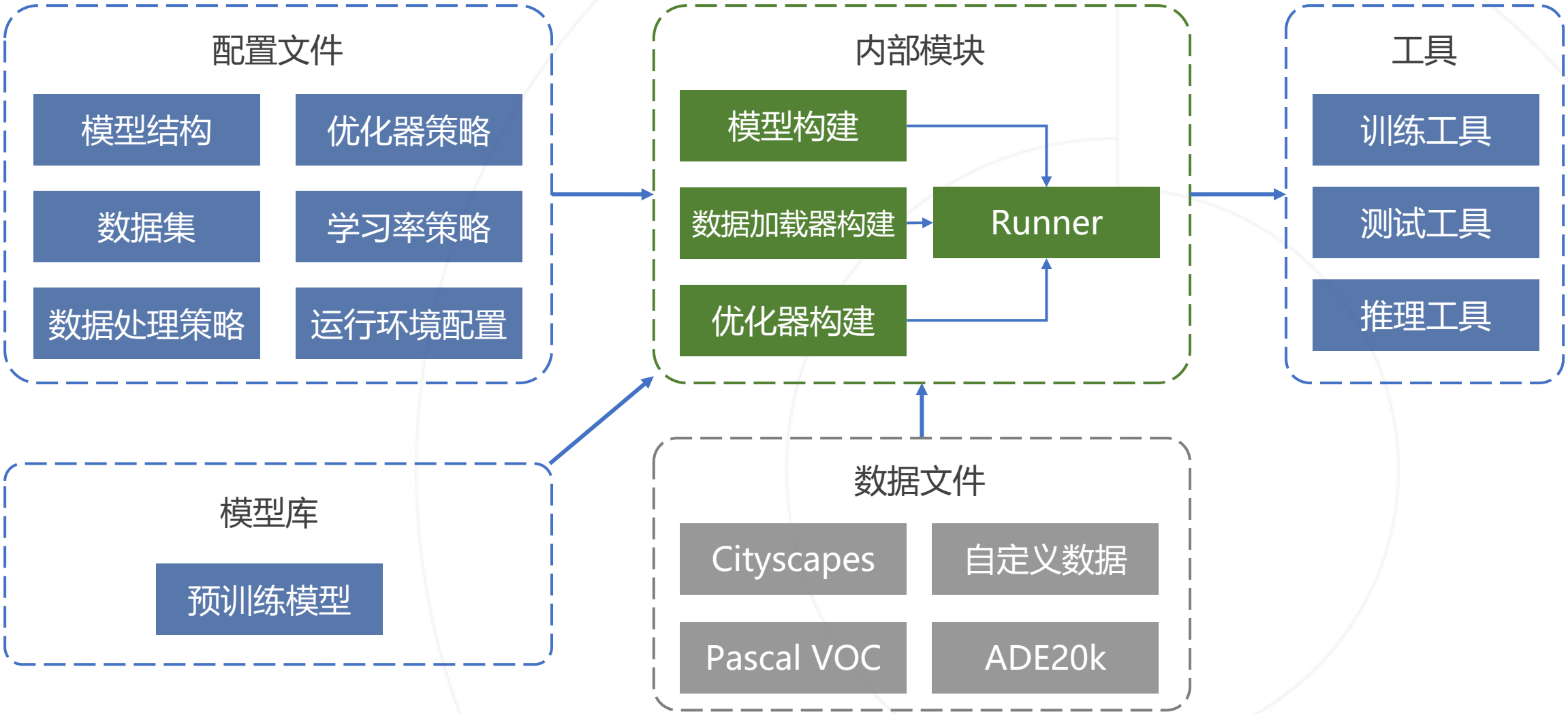
PascalVOC: 512x512

全面的性能标定

- 测试精度
- 推理速度
- 显存占用

Method	Backbone	Crop Size	Lr schd	Mem (GB)	Inf time (fps)	mIoU	mIoU(ms+flip)	download
PSPNet	R-50-D8	512x1024	40000	6.1	4.07	77.85	79.18	model | log
PSPNet	R-101-D8	512x1024	40000	9.6	2.68	78.34	79.74	model | log
PSPNet	R-50-D8	769x769	40000	6.9	1.76	78.26	79.88	model | log
PSPNet	R-101-D8	769x769	40000	10.9	1.15	79.08	80.28	model | log
PSPNet	R-50-D8	512x1024	80000	-	-	78.55	79.79	model | log
PSPNet	R-101-D8	512x1024	80000	-	-	79.76	81.01	model | log
PSPNet	R-50-D8	769x769	80000	-	-	79.59	80.69	model | log
PSPNet	R-101-D8	769x769	80000	-	-	79.77	81.06	model | log

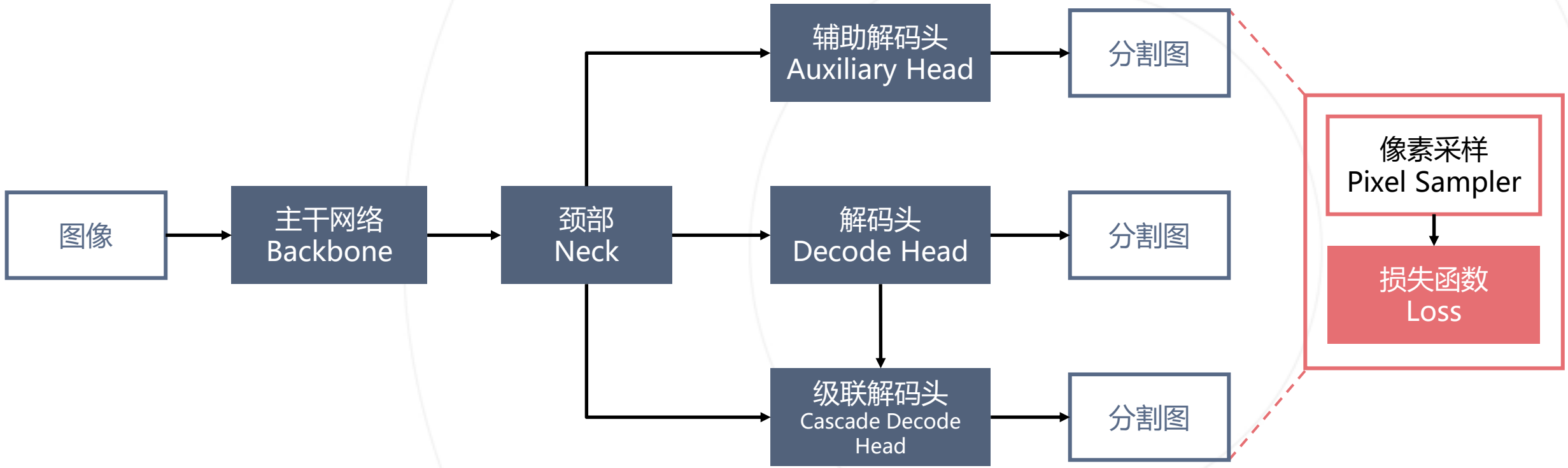
MMSegmentation 的项目结构



同时也适用于其他 OpenMMLab 工具包

分割模型的模块化设计

MMSegmentation 将分割模型统一拆解为如下模块，方便用户根据自己的需求进行组装和扩展。



分割模型的配置文件

```
model = dict(
    type='EncoderDecoder' OR 'CascadeEncoderDecoder',
    pretrained='open-mmlab://resnet50_v1c',
    backbone=dict(
        type='ResNetV1c',
        # ... more options),
    neck = None,
    decode_head=dict(
        type='PSPHead',
        # ... more options
        loss_decode=dict(
            type='CrossEntropyLoss', use_sigmoid=False, loss_weight=1.0)),
    auxiliary_head=dict(
        type='FCNHead',
        # ... more options
        loss_decode=dict(
            type='CrossEntropyLoss', use_sigmoid=False, loss_weight=0.4)),
    train_cfg=dict(...),
    test_cfg=dict(...))
```

分割模型的主体架构

主干网络

颈部

主解码头

损失函数

辅助解码头

损失函数

训练和测试的配置

主干网络的配置

主干网络输入图像，输出多层次的特征图

ResNet v1c 结构

输出全部级别的特征图
给颈部或者辅助解码头

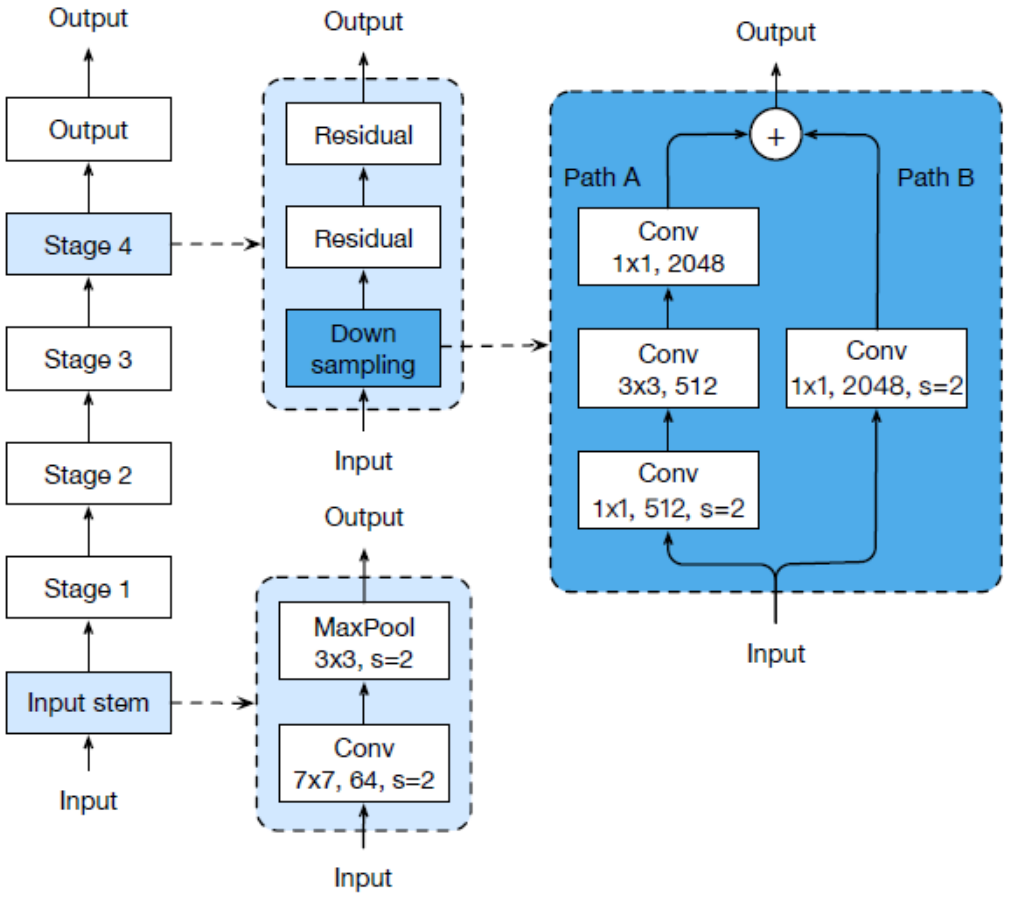
配置降采样和空洞卷积

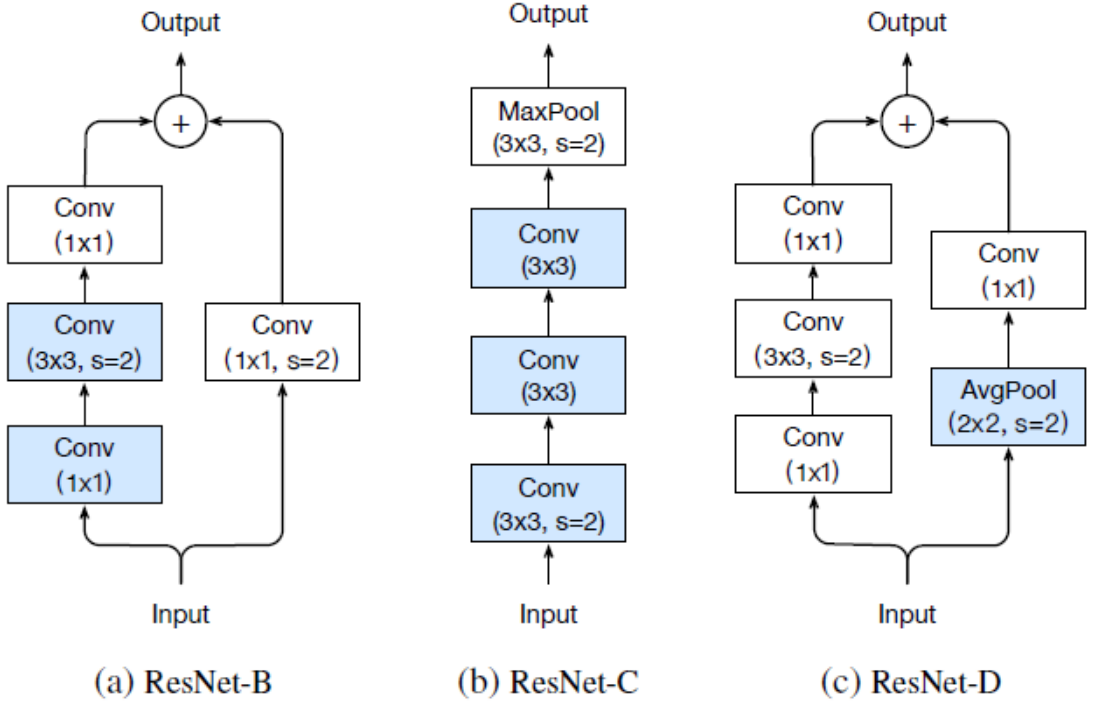
分割模型会使用 SyncBN
以增大 batch size

```
backbone=dict(
    type='ResNetV1c',
    depth=50,
    num_stages=4,
    out_indices=(0, 1, 2, 3),
    dilations=(1, 1, 2, 4),          增大空洞卷积倍率
    strides=(1, 2, 1, 1),            同时移除降采样
    norm_cfg= dict(type='SyncBN', requires_grad=True),
    norm_eval=False,
    style='pytorch',
    contract_dilation=True)
```

ResNet v1c

ResNet 50 层以上的模型在 BottleNeck 模块以及 stem 部分（即网络前几层）有一些变形：





图片来源：Bag of Tricks for Image Classification with Convolutional Neural Networks

主解码头的配置

主解码头从特征图预测分割图

使用 PSPNet 的解码头

以顶层特征图为输入

池化金字塔的尺度

预测类别数

逐像素交叉熵损失函数

```
decode_head=dict(
    type='PSPHead',
    in_channels=2048,
    in_index=3,
    channels=512,
    pool_scales=(1, 2, 3, 6),
    dropout_ratio=0.1,
    num_classes=19,
    norm_cfg= dict(type='SyncBN', requires_grad=True),
    align_corners=False,
    loss_decode=dict(
        type='CrossEntropyLoss', use_sigmoid=False, loss_weight=1.0)),
```

辅助解码头的配置

```
auxiliary_head=dict(
    type='FCNHead',                  # 使用 FCN 解码头
    in_channels=1024,                # 以低层次特征图为输入
    in_index=2,
    channels=256,                    # FCN 中卷积通道数
    num_convs=1,                     # FCN 中使用 1 层卷积
    concat_input=False,              # 是否拼接输入特征图用于预测
    dropout_ratio=0.1,
    num_classes=19,                  # 预测类别数
    norm_cfg=norm_cfg,
    align_corners=False,
    loss_decode=dict(                # 逐像素交叉熵损失函数
        type='CrossEntropyLoss', use_sigmoid=False, loss_weight=0.4)),  # 权重较小
```

辅助解码头鼓励学习更好的低层特征

不用于最终的预测

数据集配置

DataLoader 参数

数据集类型

数据集路径

batch size

worker 个数

训练集配置

图像文件目录

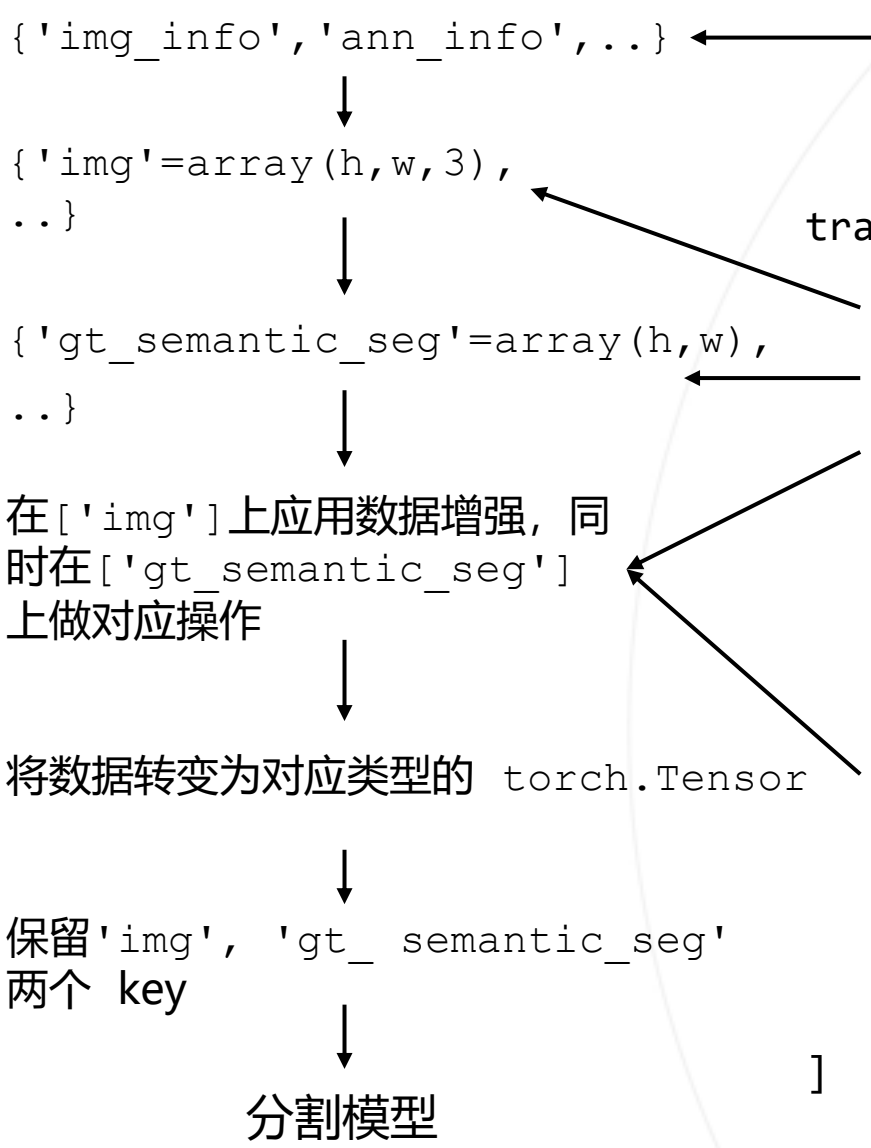
标注文件目录

训练数据的处理流水线

验证集、训练集的配置

```
dataset_type = 'CityscapesDataset'
data_root = 'data/cityscapes/'
data = dict(
    samples_per_gpu=2,
    workers_per_gpu=2,
    train=dict(
        type=dataset_type,
        data_root=data_root,
        img_dir='leftImg8bit/train',
        ann_dir='gtFine/train',
        pipeline=train_pipeline),
    val=dict(...),
    test=dict(...))
```

数据处理流水线




```
train_pipeline = [
    dict(type='LoadImageFromFile'),
    dict(type='LoadAnnotations'),
    dict(type='Resize', img_scale=(2048, 1024), ratio_range=(0.5, 2.0)),
    dict(type='RandomCrop', crop_size=crop_size, cat_max_ratio=0.75),
    dict(type='RandomFlip', prob=0.5),
    dict(type='PhotoMetricDistortion'),
    dict(type='Normalize', **img_norm_cfg),
    dict(type='Pad', size=crop_size, pad_val=0, seg_pad_val=255),
    dict(type='DefaultFormatBundle'),
    dict(type='Collect', keys=['img', 'gt_semantic_seg']),
]
```

背景类别

常用训练策略

MMSegmentation 默认的学习率策略

仅在总迭代次数上有所差别

学习率策略	20k	40k	80k	160k
总迭代数	2 万次	4 万次	8 万次	16 万次

使用多项式下降策略

基于迭代次数不是轮数

使用 mIoU 作为评估指标

```
# optimizer
optimizer = dict(type='SGD', lr=0.01, momentum=0.9, weight_decay=0.0005)
optimizer_config = dict()
# learning policy
lr_config = dict(policy='poly', power=0.9, min_lr=1e-4, by_epoch=False)
# runtime settings
runner = dict(type='IterBasedRunner', max_iters=20000)
checkpoint_config = dict(by_epoch=False, interval=2000)
evaluation = dict(interval=2000, metric='mIoU')
```

谢谢大家