

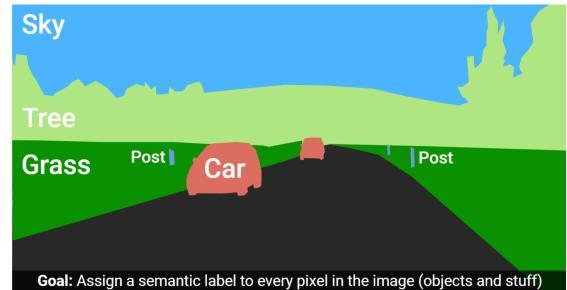
Lecture 10 - Recognition

在此之前，我们讨论的都是 CV 领域的几何相关问题；在这一章节中，我们会考虑一下 CV 领域的另一个子领域——识别领域，包括图像分类、语义分割、目标识别和实例分割

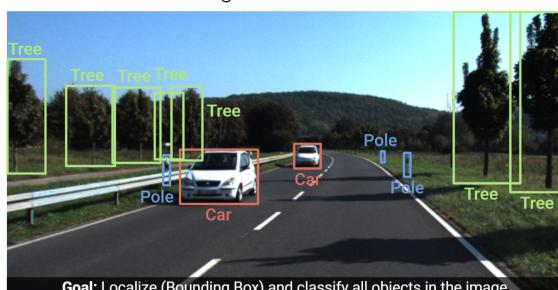
- 先简单介绍一下这 4 种问题的区别



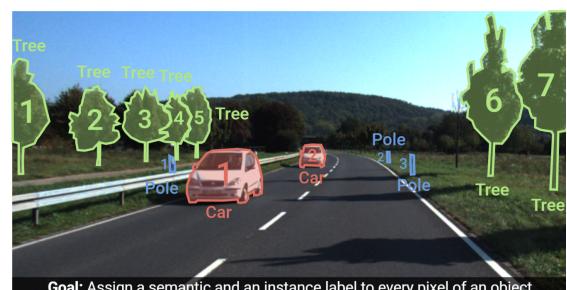
Image Classification



Semantic Segmentation



Object Detection

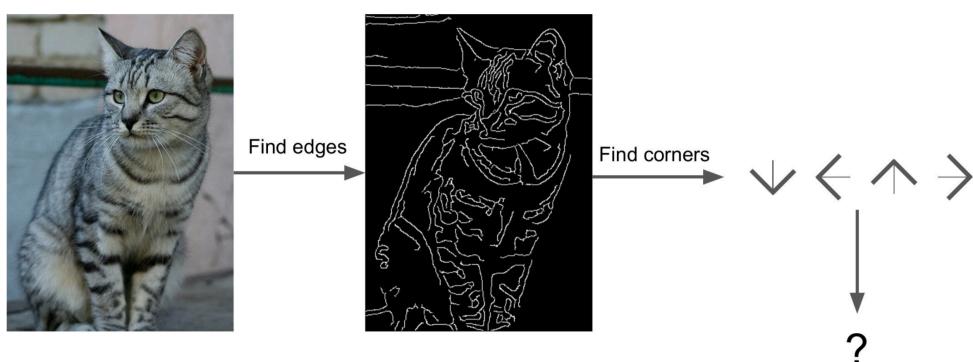


Instance Segmentation

- Image Classification (图像分类)**
 - 对于每一张图，我们希望为其分配一个离散的标签，例如上图，我们希望为其分配“道路场景”
 - Semantic Segmentation (语义分割)**
 - 对于图像中的每一个像素，我们希望为其分配一个语义标签
 - Object Detection (对象检测)**
 - 我们希望用一个 2D 的框来分类图像中的所有“对象”，而不包括背景
 - Instance Segmentation (实例分割)**
 - 有点类似于 Semantic Segmentation 和 Object Detection 的结合，在 Object Detection 的基础上，精确到像素级别；而和 Semantic Segmentation 不太一样，因为它对于每一个实例都可能会有一个编号，而不像 Semantic Segmentation ——你可以区分树和车，但是不能区分不同的树

10.1 Image Classification

- 关于图像分类任务
 - 它本身是一个非常有用的任务（例如手写数字的识别、大量图像中的关键字搜索）
 - 它的输入、输出规格很简单，并且具有标准的损失函数
 - 图像分类也是很久以来，CV 领域的困难的黄金等级 (gold-standard) 的任务
 - 它同样也是一个非常有用的高级代理任务——这意味着它可以用学习非常优良的表示，并转移到新的任务和领域
- 早期的图像分类任务尝试



- ■ 早期该领域的人们尝试在图像中寻找出所有的边界线，然后根据边界线的角度、形状等规律表示图像的类型标签，人工设计一种算法来解决该任务
- ■ 然而，利用形状、角度等信息表示，几乎是一件不可能的事，因此早期的该任务的研究是失败的，并导致该领域陷入了死循环——这也是为什么机器学习的出现成为了该领域后来发展的基础

10.1.1 Datasets

对于目前人工智能发展的最迅速的三大领域：NLP、CV、AR 来说，CV 领域的数据集的获取成本相对来说是最高的；而数据集又是一切学习方法的基础，因此我们首先来谈论该领域的数据集的发展是有必要的

- MNIST [LeCun, Bottou, Bengio and Haffner. Gradient-based learning applied to document recognition. IEEE, 1998.](#)



- ■ 这是在机器学习领域最受欢迎的数据集之一，它延伸出了很多变体，至今仍然在被使用
- 它基于美国国家标准与技术研究所，是由人口普查局的员工以及一些高中生手写得到的
- 28×28 像素的分辨率，有 6 万张标注的训练集，1 万张测试样本
- Caltech101 [Fei-Fei, Fergus and Perona: Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. CVPR, 2004.](#)



- ■ 首个主要用于目标识别的数据集，在 2003 年被收集
- 共有 101 个物体的类型，这也是为什么它被称为 Caltech101
- 这些图片主要是从 Google 搜索得到，并人工标注的
- 有一点需要主要的是，每一张图所代表的对象都位于图像的中心，因此不存在什么偏差（这可能降低学习模型的鲁棒性）
- ImageNet [Deng, Dong, Socher, Li, Li, Fei-Fei: ImageNet: A large-scale hierarchical image database. CVPR, 2009.](#)



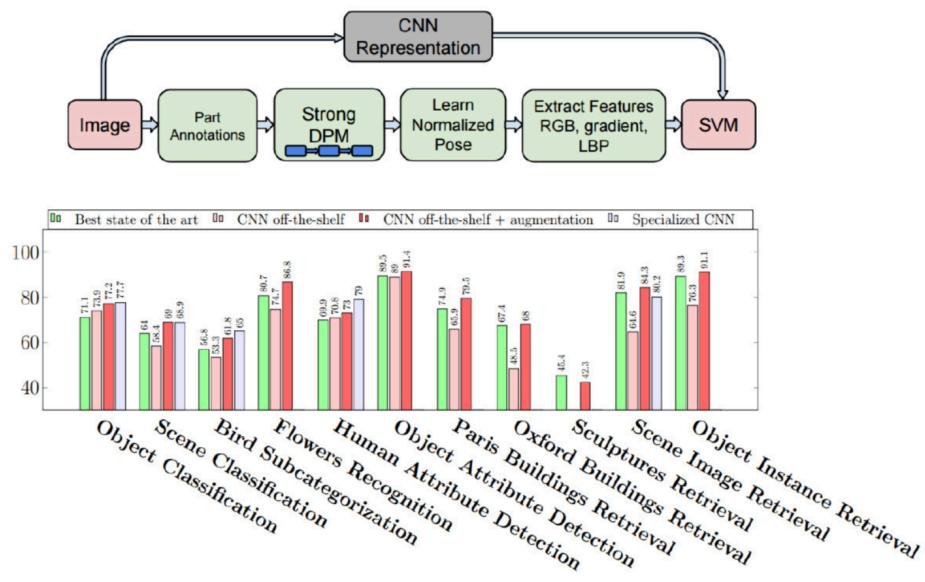
IMAGENET

22,000 categories

15,000,000 images

- ■ 2009 年，AlexNet 应用于 ImageNet，基本开启了深度学习的革命；在 2012 年，AlexNet 首次超越了传统的人工方法
- 仍然是今天预训练所用的通用数据集之一

- 共有 22k 个分类，15000k 张图像
- ImageNet 预训练
 - 预训练，或称为迁移学习 (Transfer Learning) [Razavian, Azizpour, Sullivan, Carlsson: CNN Features Off-the-Shelf: An Astounding Baseline for Recognition. CVPR Workshops, 2014.](#)



- 尽管深度网络有大量的参数，但它们通过预训练可以捕获通用的特征，能够在不同的任务中泛化
- 通常，我们会使用大规模的数据集（如 ImageNet）对 CNN 进行预训练，从而学习对图像的通用表示
- 然后，对于新任务，我们需要进行微调 (Finetune) 或者重训练
 - 对于新任务，仅需对模型的最后几层或者全部层进行微调
 - 微调需要的数据量很小，但可以实现当前最先进的性能 (SOTA, State-of-the-Art)
- 令人惊讶的是，这种迁移学习的方式甚至比专用的网络在特定任务上有着更好的效果，这种方法已经成为现代深度学习中广泛采用的标准流程

10.1.2 Challenges for Image Classification

- 图像的类型太广泛



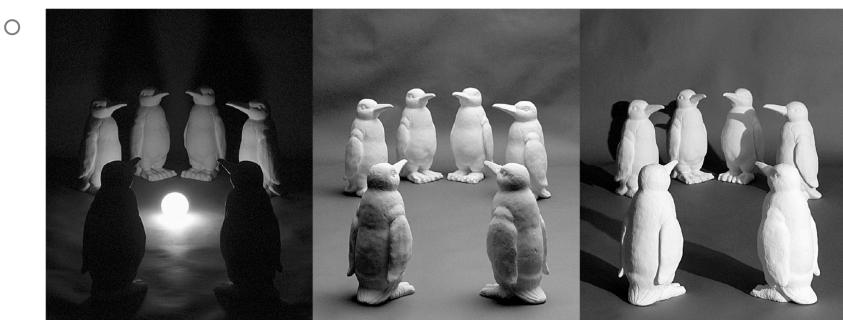
- 一类物体的外形可能有很大的差异



- 同一类物体，视角不同



- 光线强度、光源位置的变化



- 背景不同



- 同一类物体可能会变形



- 遮挡情况



10.1.3 Simple Models

- Nearest Neighbor



- 给定两个图像 I_1 和 I_2 , 定义它们之间的距离为 $\$$

$$d(\lVert \textbf{I}_1 \rVert_1, \lVert \textbf{I}_2 \rVert_1) = \sum_{p=1}^P \mid \mid I_1(p) - I_2(p) \mid \mid$$

— 对于给定的图像 \mathbf{I} ，找到其最近的邻居 $\mathbf{I}^* = \arg \min_{\mathbf{I}' \in D} d(\mathbf{I}, \mathbf{I}')$

- 返回最近的样本 \mathbf{I}^* 作为预测的类别
- 但是实际上，这是一个很糟糕的分类器——一方面，要计算所有像素的距离，这是一个开销很大的过程；其次，图像间的距离并没有包含很多的特征信息，因此它其实并不能作为一个很好的依据，从图中可以看出几乎有一半的样例都识别错误了
- 我们用一个很小的实验证明这一点
 - !

```
[[lec_10_recognition.pdf#page=23&rect=40,59,413,226|lec_10_recognition, p.17|400]]
```

- 这是两个棋盘，将第一个棋盘的黑色和白色的格子位置反转来得到第二个棋盘——也即似乎只是将其平移了一个单位；但是我们却得到了可能得到的最大的距离
 - 它们都是棋盘，但是对于 **Nearest Neighbor** 算法来说，它们之间的距离却非常大

- Bag-of-Words (BoW)

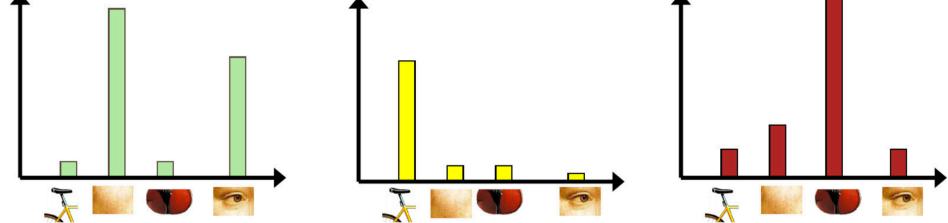


- ■ BoW 将图像分解为若干区域（例如，眼睛、嘴巴、耳朵等），每个局部区域提取特征，通过统计这些特征的频率生成特征直方图

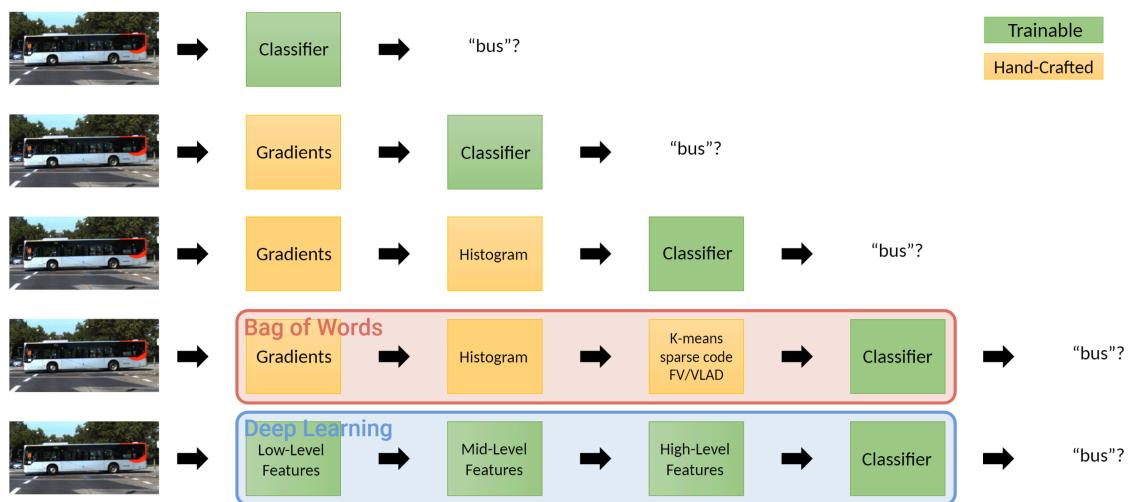
- 将这些局部特征想象装进一个袋子中，而不考虑它们的空间位置关系，通过对这些局部特征的统计进行图像表示



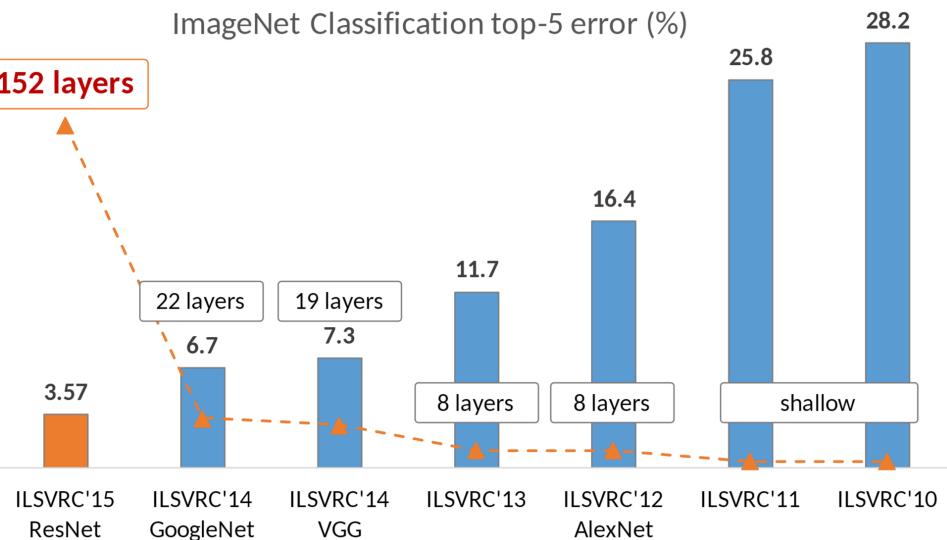
- ○ bag-of-word 的思想最早来自于 NLP 中，用于文档表示，将文档中的单词抽取出来，形成一个词袋 (bag)，忽略单词的顺序，仅关注词频
- 具体过程



- 1. 提取特征 (比如使用 SIFT 检测器)
- 2. 学习这些“可视词汇” (例如对 SIFT 的特征向量使用 k-means 聚类，生成视觉单词的字典)
- 3. 利用视觉词汇的字典，将这些可视词汇量化 (例如使用 Nearest Neighbors)
- 4. 利用直方图来表示每个可视词汇的词频
- 5. 然后训练分类器 (例如使用 k-NN、SVM、随机森林、神经网络等)
- Representation Learning (表征学习)



- ■ BoW 的问题：太多人工设计的部分，学习的占比太少
- 解决方案：学习整个模型（从图像输入到决策），采用 end-to-end 的结构
- ImageNet 大规模视觉识别挑战赛 (ILSVRC)

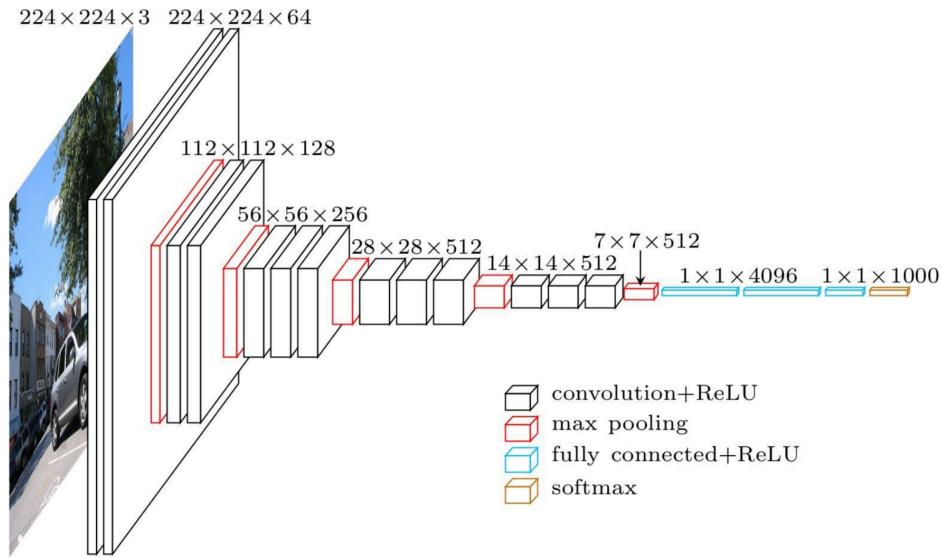


- ■ 评价指标是前 5 个类别是否包含正确类别，数值越低，模型的性能越好
 - 从 2012 年使用 AlexNet，引入 8 层 CNN，错误率骤降，标志着深度学习在 CV 领域的突破
 - 直到 15 年，使用 152 层的深度残差网络 ResNet，错误率降至 3.56%，已经接近人类水平
- 这展示了在 5 年来，近乎不可能的跨越式的进步

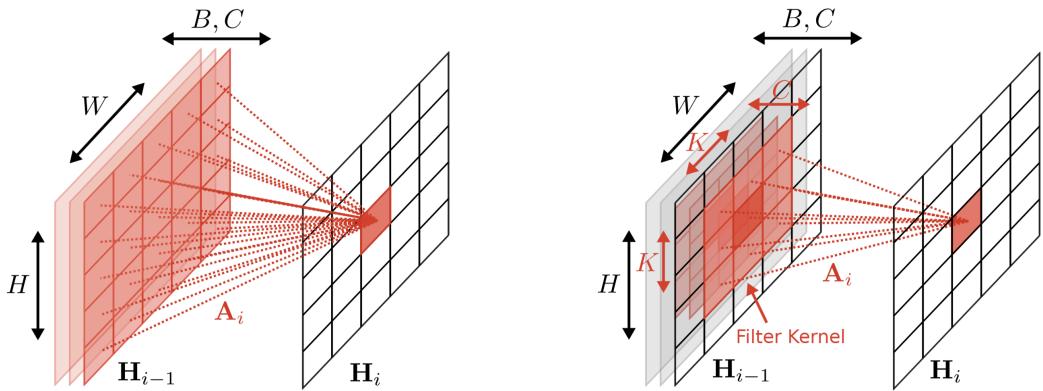
10.1.4 Convolutional Neural Networks

至此，我们可以看到，深度学习对于图像分类任务十分重要，而2012年提出的CNN无疑是目前一切成就的基础，那么我们先从CV角度的CNN讲起

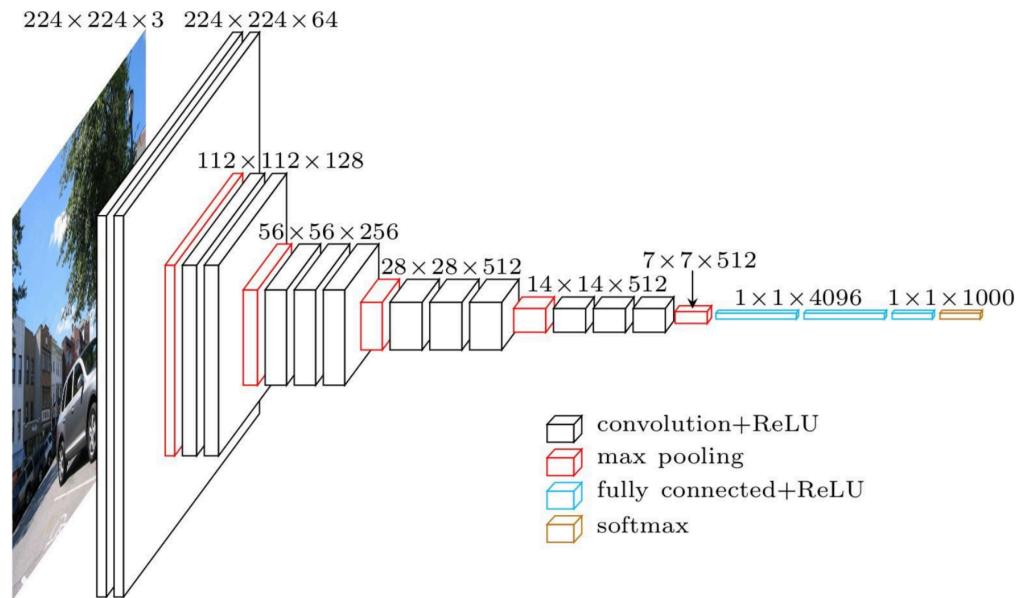
- VGG (Visual Geometry Group, 2014) 网络



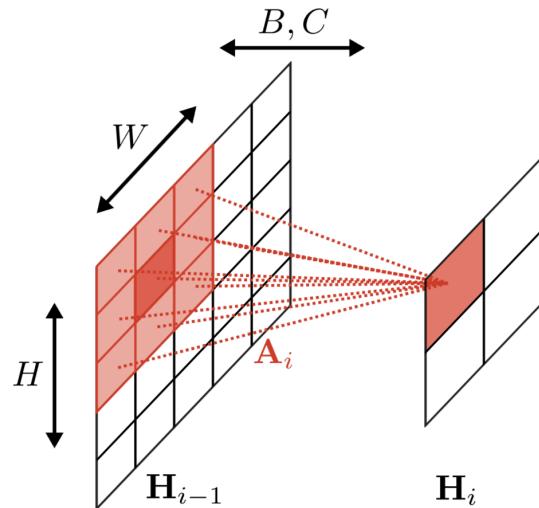
- ■ 对于 CNN 网络中，一般有 3 种类型的层
 - 卷积层 (Convolution layers)
 - 提取特征，卷积后采用 ReLU 激活函数
 - 多个卷积层叠加，通道数逐渐增加
 $64 \rightarrow 128 \rightarrow 256 \rightarrow 512$
 - 下采样层 (Downsampling layers)
 - 使用最大池化 (Max Pooling) 层对特征图尺寸进行缩减
 - 池化窗口大小为 2×2 ，步长为 2
 - 全连接层 (Fully connected layers)
 - 最后三个全连接层 $4096 \rightarrow 4096 \rightarrow 1000$ ，输出的是 1000 个类别的预测结果
 - Convolutional Layers



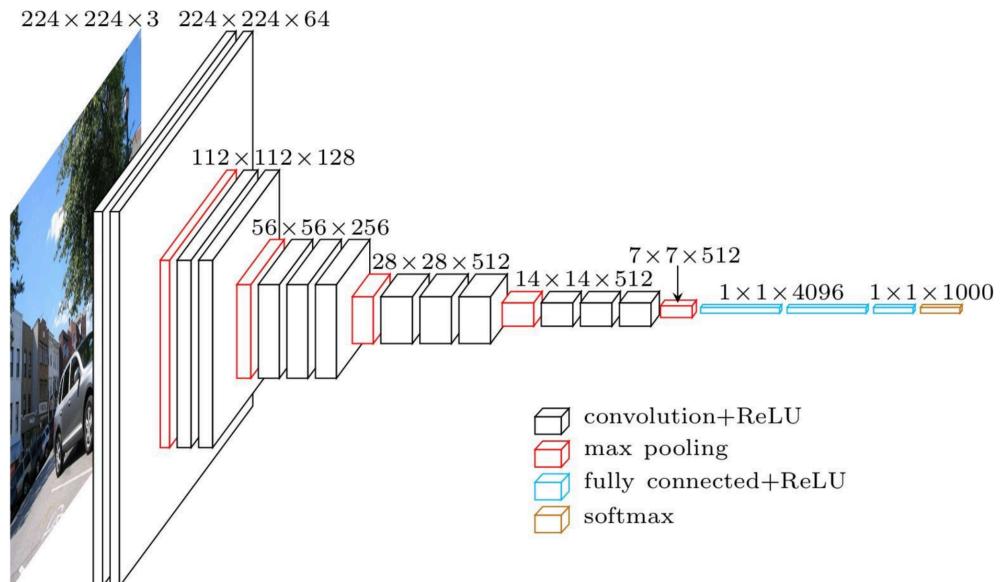
- - 相比起左侧的全连接层，卷积层只是将一个卷积核 (Filter Kernel) 大小的块连接到指定像素，这里涉及到两个重要的概念
 - 权重共享 (Weight Sharing)
 - 在一个卷积层中，同一个卷积核的权重在不同位置是固定的——也即，同一个卷积核在输入数据上滑动时，提取的是相同的特征
 - 而且由于卷积核滑动提取的特点，我们可以捕捉到局部细节的特征
 - 另一方面，比起FC层，卷积层大大减少了参数量
 - 平移等变性 (Translation Equivariance)
 - 可能是 CNN 最重要的一个性质，它指的是输入的平移会导致输出的特征图以相同的方式平移——举个例子来说，就是我们假如我们训练好一个模型可以识别一张猫在正中心的图片，输出猫的位置；那么当我们把这个猫向右平移一下位置，那么对应的特征也会向右平移，输出的猫的位置也会对应的向右平移
 - 公式描述：假设输入特征图为 H ，卷积操作用 $f_\theta(\cdot)$ 表示， θ 是卷积核的参数，平移操作用 T 表示，则： $T[f_\theta(H)] = f_\theta[T(H)]$
 - Downsampling



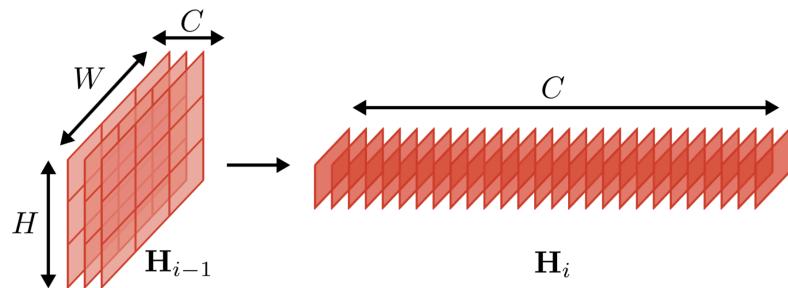
- - 下采样操作会降低特征图的空间分辨率，常见的下采样操作包括池化 (Pooling) 和步幅 (Stride)
 - 下采样操作可以增大每个神经元的感受野 (receptive field)，使其能够捕捉更大范围的上下文信息
- Pooling



- - 池化通常使用 2×2 的窗口，步长为 2，将特征图的宽、高减半，通道数不变（池化操作单独的应用于每一个通道）
 - 池化没有特定的参数（典型的池化操作包括最大池化 (Max Pooling)、平均池化 (Average Pooling) 等）
- Fully Connected Layers



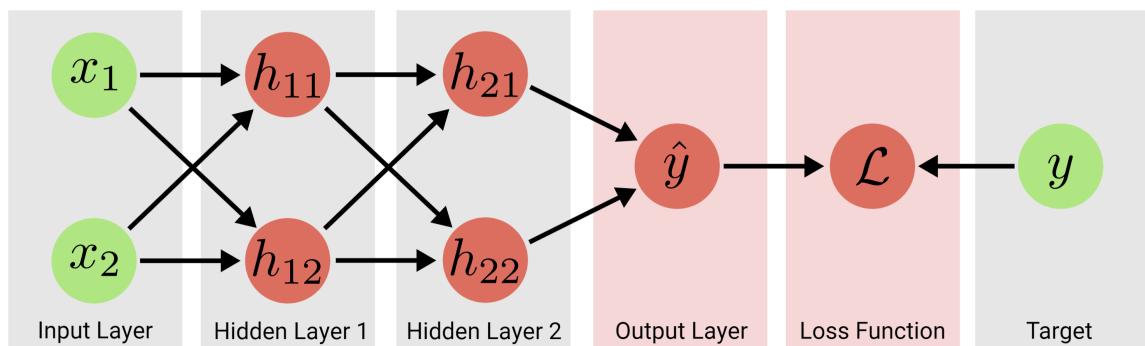
- FC 层是 VGG 结构中最消耗内存的部分，因为需要存储的参数数量非常庞大
 - 全连接层能够结合所有输入特征，实现全局语义理解



- 在 VGG 网络结构中，最后一步并不是池化，而是 reshape 操作

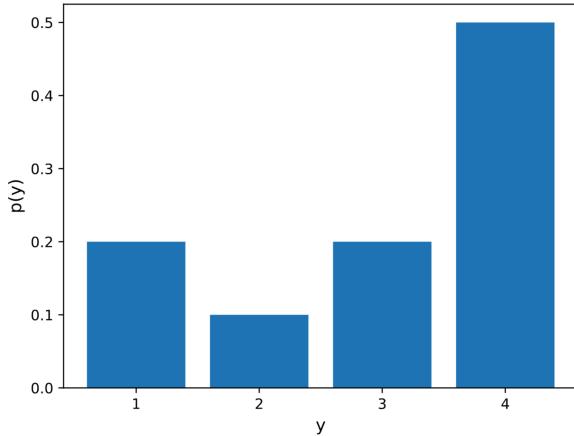
感受野 (receptive field): 是指神经网络中某一层的神经元在输入空间上能够“看到”或“感知到”的区域范围。简单来说，感受野描述了一个神经元对原始输入数据的响应范围

- 输出层和损失函数



- 前面介绍的 3 种层属于隐藏层的部分，最后会连接到输出层，接收前面网络结构的输出，计算出模型最终的输出结果

- 损失函数用于比较预测结果和真实结果的差距
- 在 Image Classification 任务中，我们使用 softmax 层作为输出层，使用 cross entropy (交叉熵) 作为损失函数
- 接下来，我们看一下为什么在图像分类任务中使用这样的输出层和损失函数
 - Categorical Distribution (分类分布)



- 分类分布表示某个类别 y 的概率分布

$$\mu_c = p(y = c)$$

- 其中 μ_c 表示类别 c 的概率

- 另一种表示方式——one-hot 编码 \$\$

$$p(\text{y}) = \prod_{c=1}^C \mu_c^{y_c}$$

= y : 称为 one-hot 向量 $y \in \{0, 1\}^C$ ，其中只有 1 个位

- Softmax

- 我们如何确保 $f_w(x)^{(c)}$ 输出的结果是一个合法的分类分布呢？

- 我们必须保证

$$f_w(x)^{(c)} \in [0, 1] \text{, 并且} \\ \sum_{c=1}^C f_w^{(c)} = 1$$

- 而 softmax 函数则可以保证上面两个式子同时成立，下面是 softmax 函数的一般形式 \$\$

$$\text{softmax}(\mathbf{x}) = \left(\frac{\exp(x_1)}{\sum_{k=1}^C \exp(x_k)}, \dots, \frac{\exp(x_C)}{\sum_{k=1}^C \exp(x_k)} \right)$$

- 令 \mathbf{s}_c 作为模型最后一层输出的分数向量，则模型预测类别 c 的得分可以由

- Log Softmax

- 如果我们在上面的式子基础上左右两侧取对数，则得到下式 $\log f(\mathbf{w}^T \mathbf{x})$

$$\log f(\mathbf{w}^T \mathbf{x}) = s_c - \log \sum_{k=1}^C \exp(s_k)$$

- 第1项 s_c – 鼓励正确类别的得分 s_c 增加 – 第2项 $\log \sum_{k=1}^C \exp(s_k)$ – 惩罚

$$\text{softmax}(\mathbf{x}) = \frac{\exp(\mathbf{x}_c)}{\sum_{k=1}^C \exp(\mathbf{x}_k)}$$

- 这样可以避免指数函数计算中出现过大的值，确保数值稳定

- 接下来，我们给出一个实例来说明上面的这些内容
 - 这里假设有 4 个类别，4 个训练样本——可以看出，CE 损失函数更倾向于惩罚模型错误预测概率较高的样本（这里的大象样本）

Input \mathbf{x}	Label \mathbf{y}	Predicted scores \mathbf{s}	$\text{softmax}(\mathbf{s})$	CE Loss
	(1, 0, 0, 0)^\top	(+3, +1, -1, -1)^\top	(0.85, 0.12, 0.02, 0.02)^\top	0.16
	(0, 1, 0, 0)^\top	(+3, +3, +1, +0)^\top	(0.46, 0.46, 0.06, 0.02)^\top	0.78
	(0, 0, 1, 0)^\top	(+1, +1, +1, +1)^\top	(0.25, 0.25, 0.25, 0.25)^\top	1.38
	(0, 0, 0, 1)^\top	(+3, +2, +3, -1)^\top	(0.42, 0.16, 0.42, 0.01)^\top	4.87

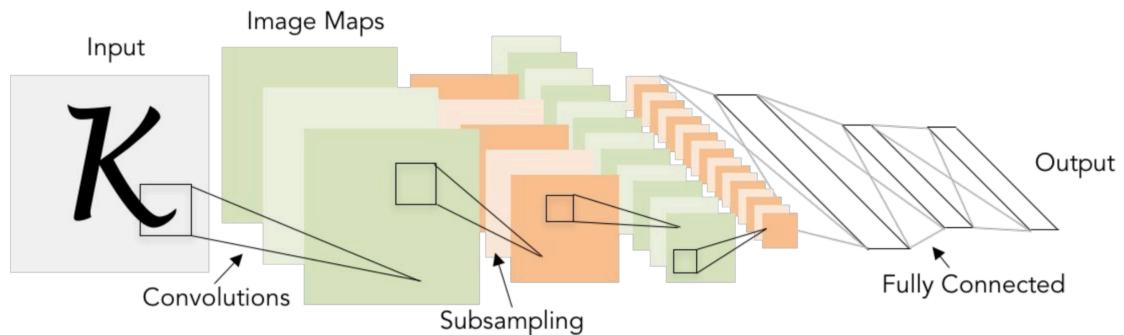
- 对于单个的训练样本 $(\mathbf{x}, \mathbf{y}) \in \mathcal{X}$ ，CE 损失计算如下 $\log f(\mathbf{w}^T \mathbf{x})$

$$\text{CE Loss} = -\sum_{c=1}^C y_c \log f(\mathbf{w}^T \mathbf{x})$$

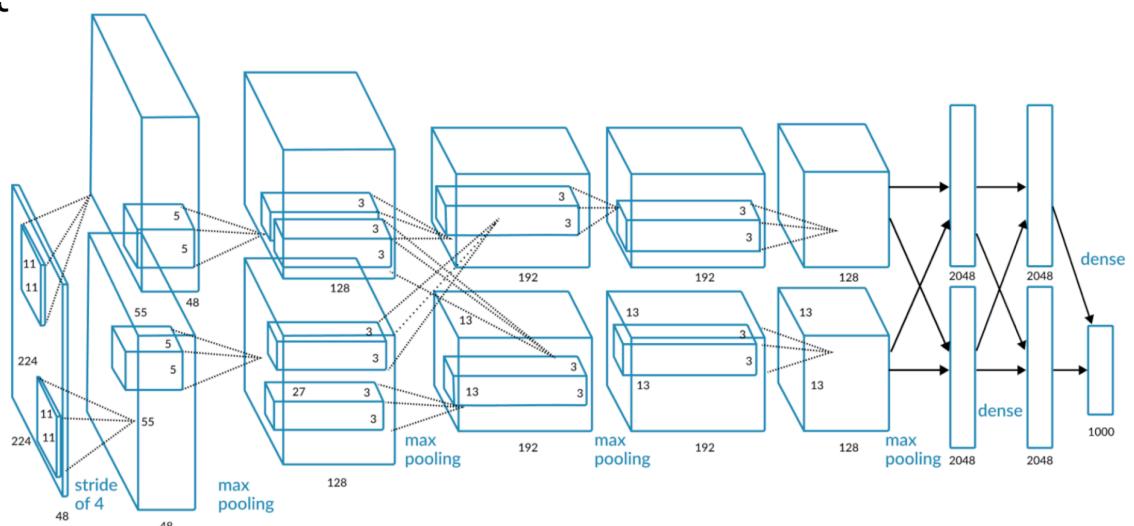
10.1.5 Network Architectures

接下来展示一些典型的神经网络的结构

- 1998: LeNet-5 [LeCun et al. Gradient-based learning applied to document recognition. IEEE, 1998.](#)

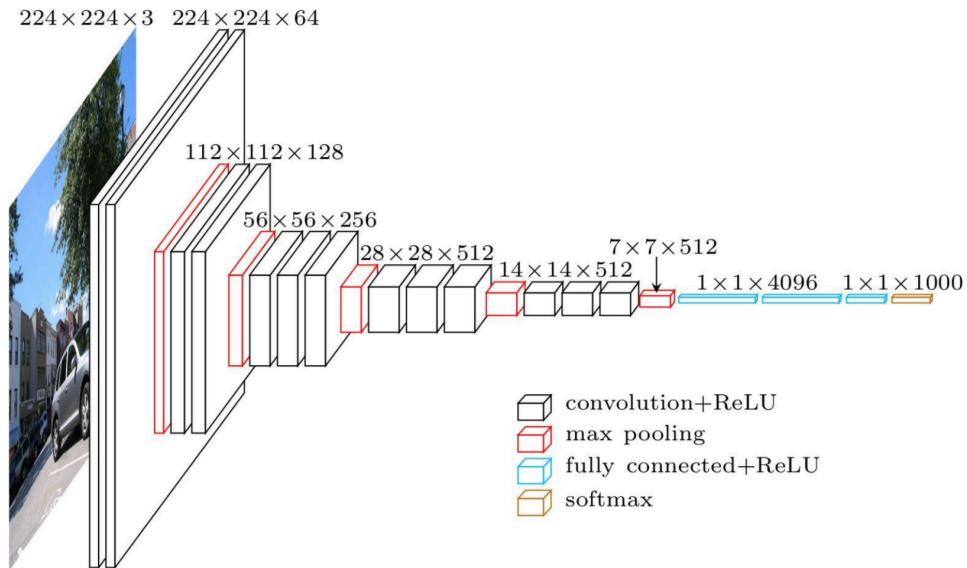


- ■ 在 1998 年，第一个成功的 CNN 网络
- 2 个卷积层 (5×5)、2 个池化层 (2×2)、2 个 FC 层，是一个很小的神经网络
- 在 MNIST 数据集上取得了 SOTA 的成果——不过在 CV 领域中，人们更喜欢关注于现实一些的问题
- 2012: AlexNet [Krizhevsky, Sutskever and Hinton: ImageNet Classification with Deep Convolutional Neural Networks. NIPS, 2012.](#)

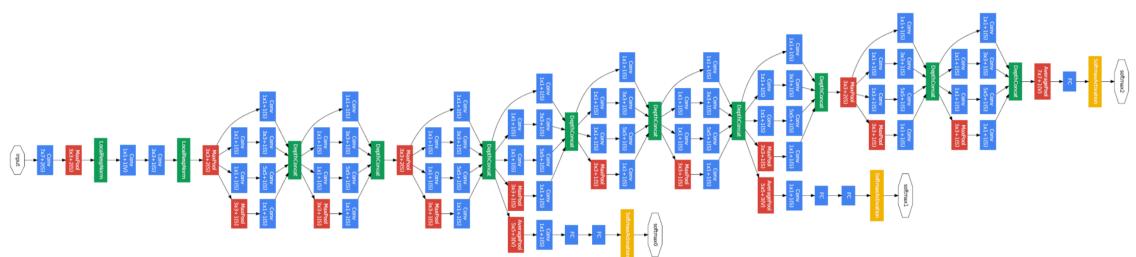


- ■ 8 层结构，包括 5 个卷积层和 3 个 FC 层（在 2 块 GTX 580 上训练，同时也象征着 GPGPU 并行化的开始）；
 - 使用 ReLU 激活函数（相比 Sigmoid 或 Tanh，ReLU 加快了训练速度并减少梯度消失问题）；

- 在 FC 层中使用 Dropout，减少过拟合，使用数据扩充 (Data augmentation) (如图像翻转、平移、旋转) 扩充训练数据，提高泛化能力
- 特征通道数逐渐增加，空间分辨率逐层降低
- 在 2012 年的 ImageNet 比赛中大获成功，证明了 CNN 在图像分类任务中的强大性能
- 推动了深度学习的革命风潮
- 2015: VGG [Simonyan and Zisserman: Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR, 2015.](#)

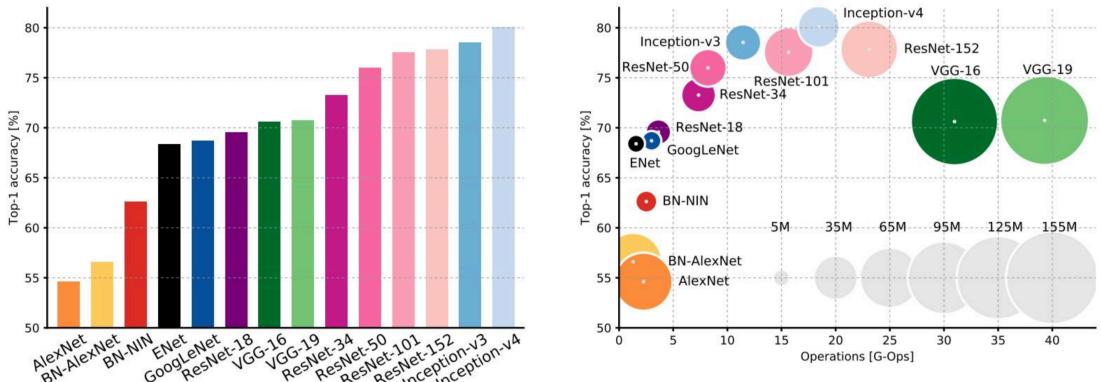


- ■ 全部采用 3×3 的卷积核
- 3 个 3×3 的卷积层，与单层 7×7 的单层卷积层，拥有同样的感受野，但是可以减少参数量、引入更多的非线性特性
- VGG 有 2 个主要的变体：VGG-16 (13 卷积+3FC) 、VGG-19 (16 卷积+3FC) ，证明网络越深，表达能力越强
- 2015: Inception / GoogLeNet [Szegedy et al.: Going deeper with convolutions. CVPR, 2015.](#)



- ■ 模块化设计的网络——Inception 模块，通过并行分支引入了多种卷积核和池化操作，整合了多尺度特征
 - 每个模块包含
 - 不同大小卷积核 (如 1×1 、 3×3 、 5×5)

- 最大池化层
 - 1×1 卷积用于降维，减少计算量
 - 网络深度为 22 层，是当时最深的网络之一
 - 创新：
 - 中间分类头
 - 在中间层引入分类器，辅助训练，缓解梯度消失问题
 - 全局平均池化
 - 取代 FC 层，大幅减少参数量（比 VGG-16 的参数数量少了 27 倍）
 - 多尺度特征提取
 - Inception 模块整合了多尺度信息，适应不同大小的目标
 - 2016: ResNet [He, Zhang, Ren and Sun: Deep Residual Learning for Image Recognition. CVPR, 2016.](#)
- The diagram illustrates the ResNet architecture. On the left, a sequence of layers is shown: '7x7 conv, 64/2' followed by a series of '3x3 conv, 64' layers. A 'pool-1/2' layer is indicated above the first two '3x3 conv' layers. A residual connection is shown as a skip path from the input to the output of the '3x3 conv, 128' layer. On the right, a detailed view of a residual block shows the input x being processed by a 'weight layer' and 'relu' function to produce $\mathcal{F}(x)$. This is then added to the input x via an 'identity' connection (indicated by a circle with a plus sign) before passing through another 'weight layer' and 'relu' function to produce the final output $\mathcal{F}(x) + x$.
- 随着网络深度的增加，梯度消失或梯度爆炸的问题使得深层网络的训练极为困难
 - ResNet 引入一种方案：残差连接 (Residual Connection)，通过跳跃连接 (skip connection) 缓解了深层网络的训练难题
 - 非常简单标准的网络结构，使用 3×3 的卷积核，使用步幅 (Stride) 代替池化进行下采样
 - ResNet 提供了多个深度变体 (18、34、50、101、152)，即使网络深度大幅增加，残差连接并未显著增加计算复杂度
 - 残差连接的思想被广泛应用于后续模型 (如 DenseNet、Transformer)
 - Accuracy VS Complexity



- 左图：模型的 Top-1 准确率

- AlexNet 和 BN-AlexNet：准确率最低（约55%），表明较浅的网络难以捕捉复杂的特征
- GoogLeNet 和 ResNet 系列：随着深度增加（如 ResNet-50、ResNet-101、ResNet-152），准确率显著提升，达到75%-80%
- VGG 系列：VGG-16 和 VGG-19 的准确率中等（约71%-73%），但因参数较多，效率相对较低
- Inception 系列：Inception-v3 和 Inception-v4 的准确率最高，达到了接近80%

- 右图：准确率 VS 复杂度

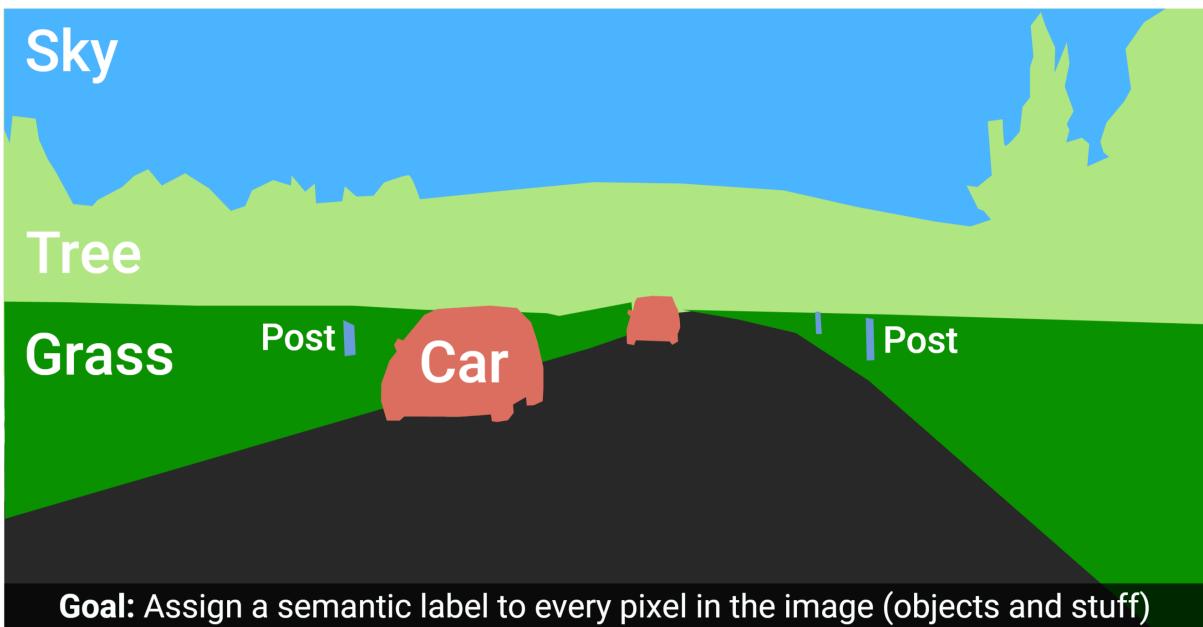
- 横轴为计算复杂度、纵轴为 Top-1 准确率，气泡大小表示参数量
 - AlexNet：
 - 操作数小 (<5 G-Ops)，但准确率较低 (55%)
 - 参数量较大，但性能并不优越
 - VGG 系列：
 - 准确率较高，但计算复杂度和参数量都很大（如 VGG-19 的参数量达155M）
 - 在效率上表现较差
 - GoogLeNet 和 ResNet 系列：
 - 通过设计更高效的架构（如 Inception 模块、残差连接），在计算复杂度和准确率之间取得了良好平衡
 - ResNet 和 Inception-v3/v4 具有较高准确率，同时计算复杂度和参数量控制得当
 - Inception-v4：
 - 在高准确率（约80%）的同时，计算复杂度和参数量得到很好控制

- 达到了最高的准确率 (80%+)，同时保持了较低的计算复杂度

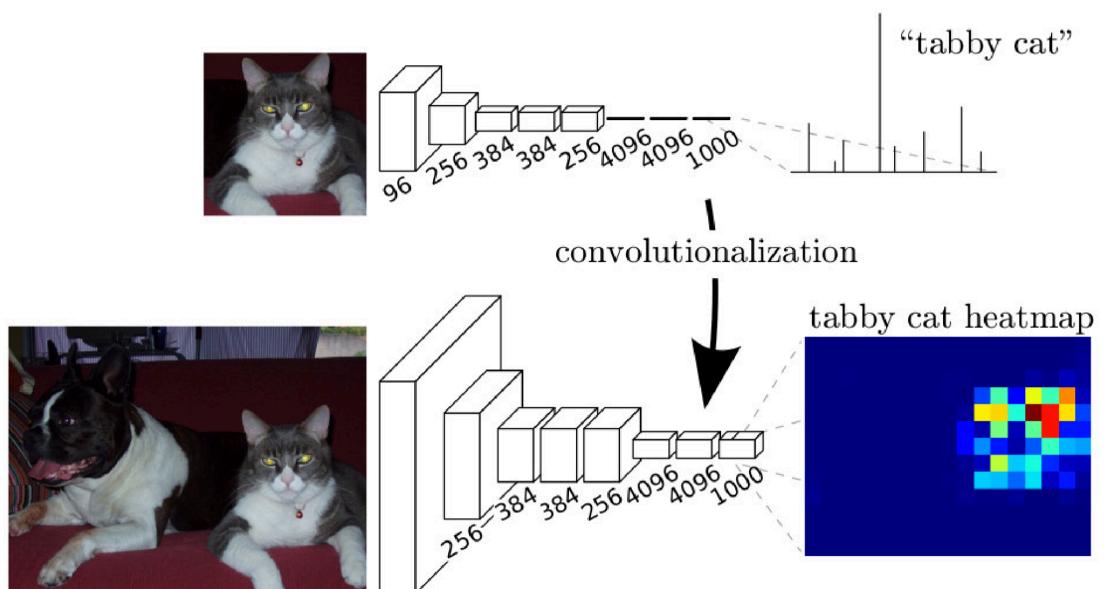
10.2 Semantic Segmentation

接下来我们讨论关于语义分割部分的问题；在深度学习出现之前，也有一些人尝试用传统方法解决这类问题，但是它们的效果远不及深度学习方法——因此，我们直接从深度学习方法开始讨论

- 我们首先再回顾一下语义分割任务的目标



- 我们希望给图像中每一个像素分配一个语义标签
- Fully Convolutional Networks [Long, Shelhamer and Darrell: Fully convolutional networks for semantic segmentation. CVPR, 2015.](#)



- 在 2015 年，来自伯克利大学的一个小组意识到，既然我们目前的分类网络发展的这么强大，为什么我们不将其应用在语义分割任务中？
 - 通过以卷积方式应用分类网络来获取不同类别的热图，在原文中将其称为 convolutionalization 操作，例如图中给出了 tabby cat 标签对应的热图；可以看到，颜色最偏向红色的部分的像素是最有可能为 tabby cat 的部分
- 然而，这引申出一个问题——在之前的网络中，我们都会使用池化等方式下采样，从而提高感受野，然而会导致输出图像的分辨率降低，在这种问题中，我们希望输出的概率分布热图是和输入图像等分辨率的；
 - 但是如果我们不使用池化来提高感受野的话，那么可能需要几百层的卷积才能够达到同样的效果——而几百层的卷积的训练难度是不可想象的；

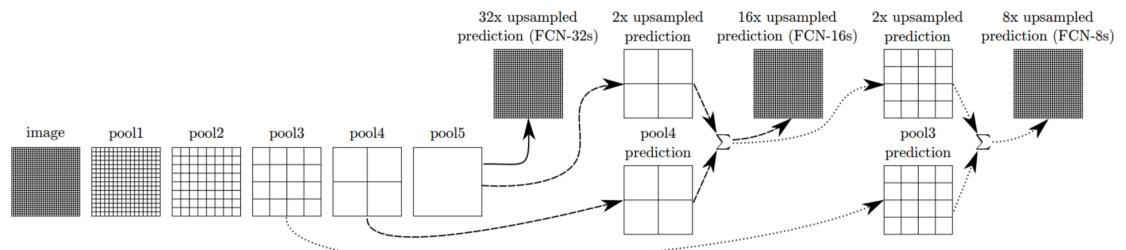
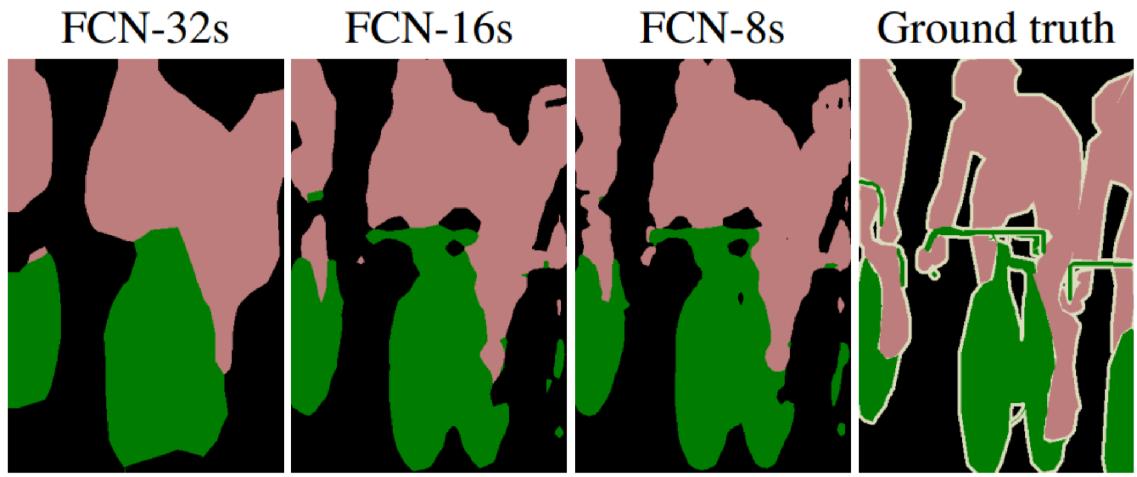
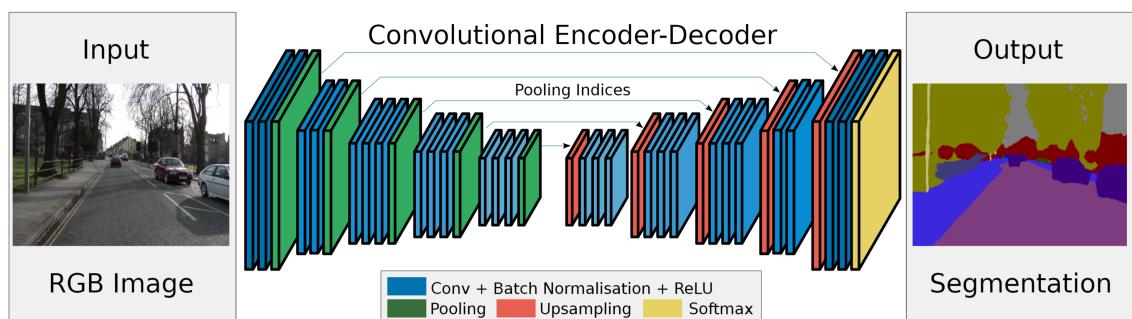


Figure 3. Our DAG nets learn to combine coarse, high layer information with fine, low layer information. Layers are shown as grids that reveal relative spatial coarseness. Only pooling and prediction layers are shown; intermediate convolution layers (including our converted fully connected layers) are omitted. Solid line (FCN-32s): Our single-stream net, described in Section 4.1, upsamples stride 32 predictions back to pixels in a single step. Dashed line (FCN-16s): Combining predictions from both the final layer and the pool4 layer, at stride 16, lets our net predict finer details, while retaining high-level semantic information. Dotted line (FCN-8s): Additional predictions from pool3, at stride 8, provide further precision.

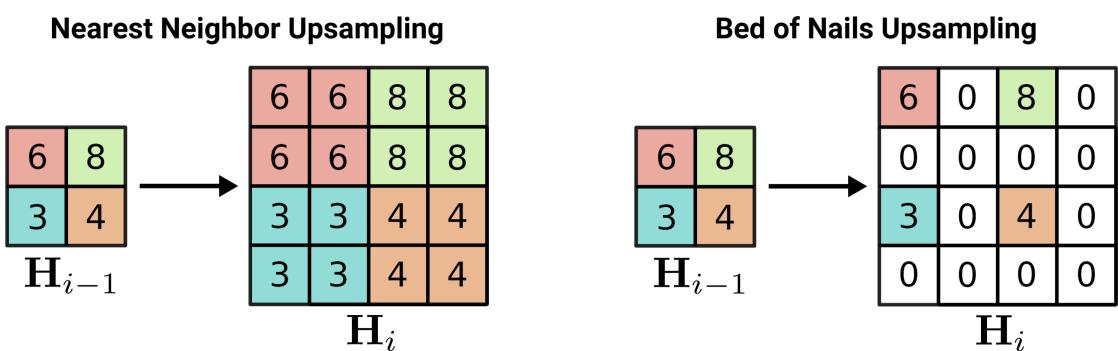
- 该篇文献的作者提出了一种想法——学习上采样 (upsampling) 操作，结合高层（深层网络）和低层（浅层网络）的信息。其中，高层捕捉全局语义信息，但分辨率较低；低层具有更高的分辨率，但语义信息较弱
 - FCN 通过一种分层结构，把不同层的特征融合起来，既保证语义准确性，又保留细节分辨率



- ■ 图中展示了不同版本的FCN的结构及其效果，从左到右分别为pool5+上采样32倍、pool5+pool4+上采样16倍、pool5+pool4+pool3+上采样8倍
 - 可以看到效果确实是逐渐变好的
- 不过，这是一个早期的版本，在2015年被提出，效果仍然是比较粗糙的
- SegNet

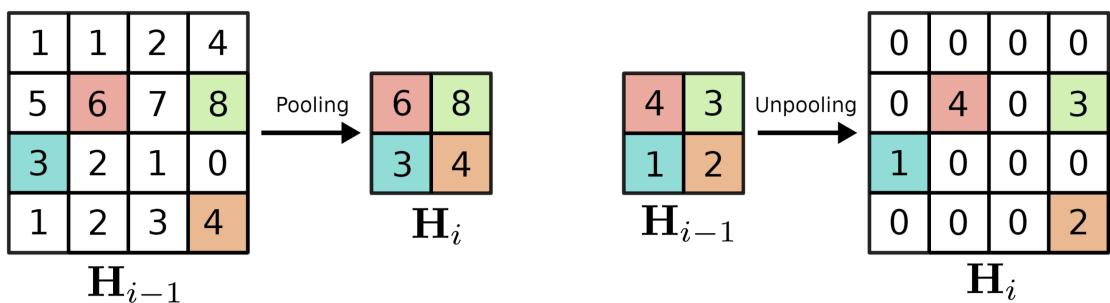


- ■ SegNet是另一个采用了下/上采样思想的模型
- 采用了编码器-解码器架构。其中编码器部分对输入图像进行卷积、批归一化、ReLU激活和池化操作，在提取特征的同时减少分辨率；解码器部分使用与编码器对称的架构，将下采样操作换成了上采样操作，逐级恢复分辨率
- 上采样操作

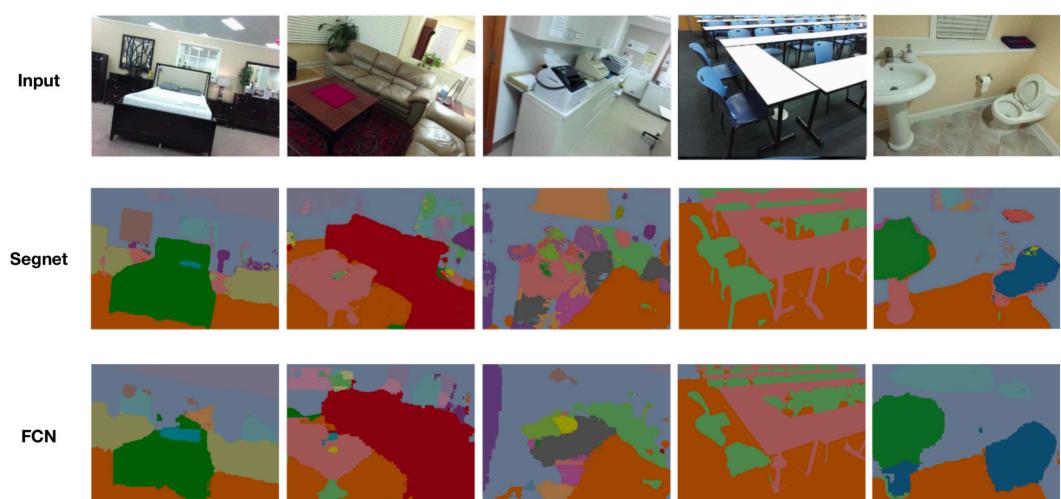


- 对于上采样而言，我们常规可以采用的方法包括
 - Nearest Neighbor (最邻近插值)

- 通过最近邻插值，将低分辨率特征图中的像素值复制到高分辨率特征图中，扩展为更大的特征图
- **Bilinear (双线性插值)**
 - 使用双线性邻近值插值
- **Bed of nails (钉床插值)**
 - 通过稀疏插值的方法，仅在扩展特征图的特定位置插入原始像素值，其余位置填充为零
- 但是在这篇文章中，它们实际采用的是一种称为最大反向池化 (Max Unpooling) 的操作
- Max Unpooling



- 最大池化操作将核大小的像素中的最大值取出——而最大反向池化操作则是记住了对应池化操作的最大值的位置，将第 $i-1$ 层的所有最大值映射回它们原来所在的位置，其余部分补零
- 最大反向池化操作，通过位置信息的还原，确保上采样后的特征图与原始结构保持一致
- 从原理上不难看出，最大反向池化操作必须和池化操作一一对应
- SegNet 的结果展示

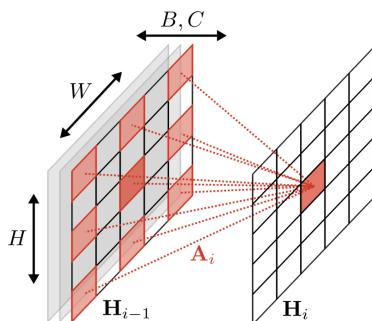


- ○ 可以看到，SegNet 的结果确实优于 FCN 的结果

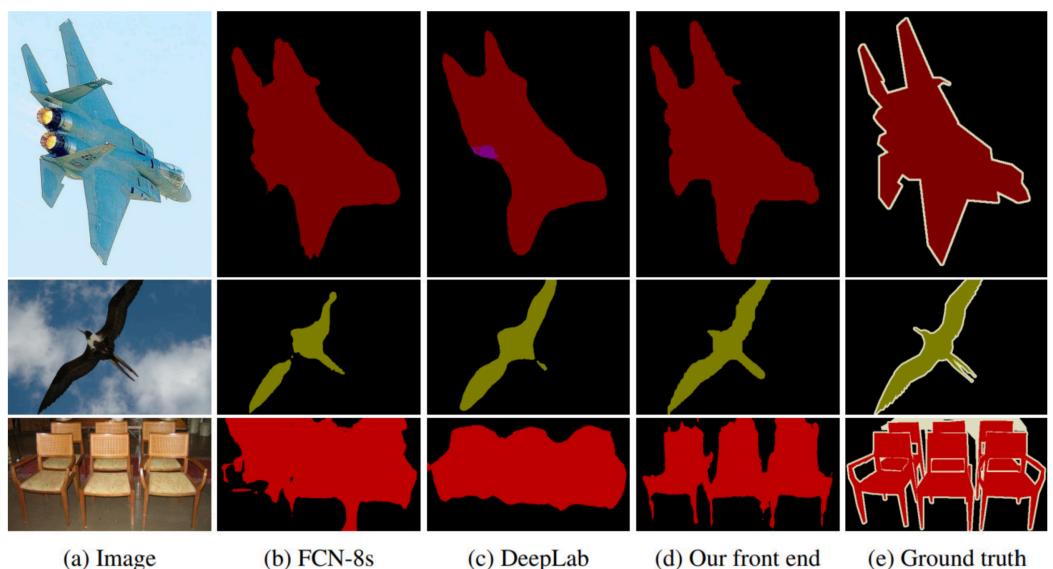
另一种可以替代下/上采样操作来快速扩大感受野同时保持分辨率的方法，称为空洞卷积 (Dilated Convolution)

- Dilated Convolution (空洞卷积) [Yu and Koltun: Multi-Scale Context Aggregation by Dilated Convolutions. ICLR, 2016](#)

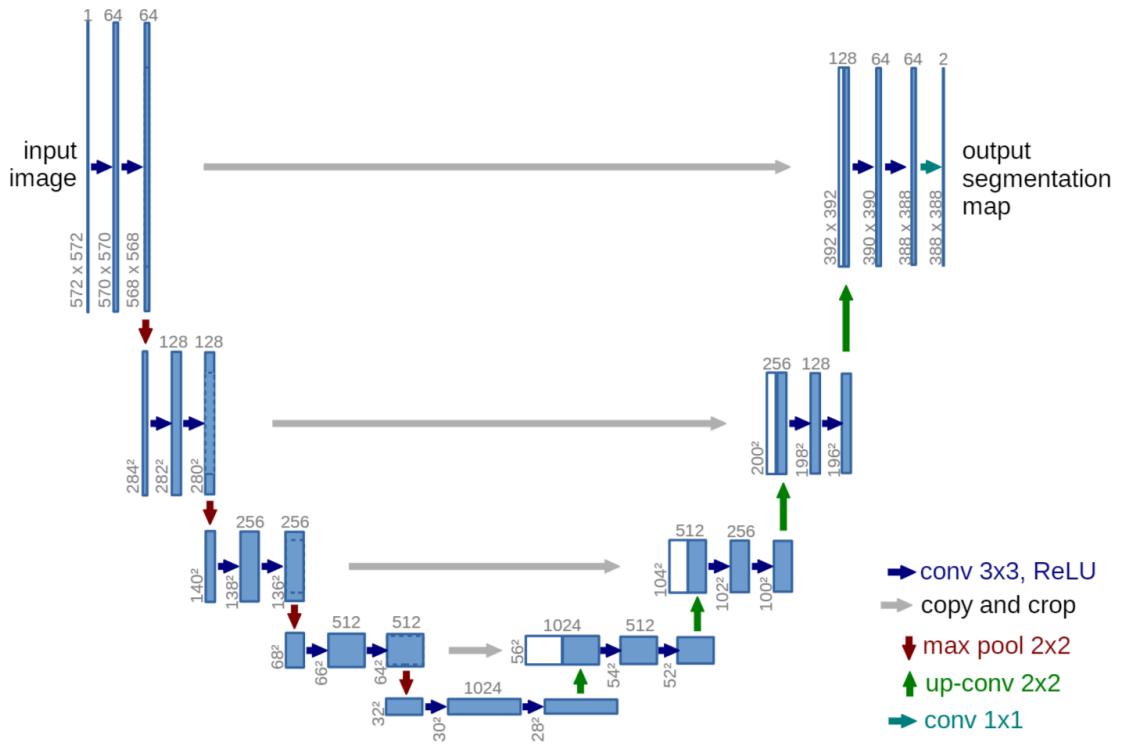
- 其核心思想是通过插入空洞 (dilation) 来扩展卷积核的感受野，而不需要增加参数量或减少特征图的分辨率；一般会结合一些其它的 backbone 使用



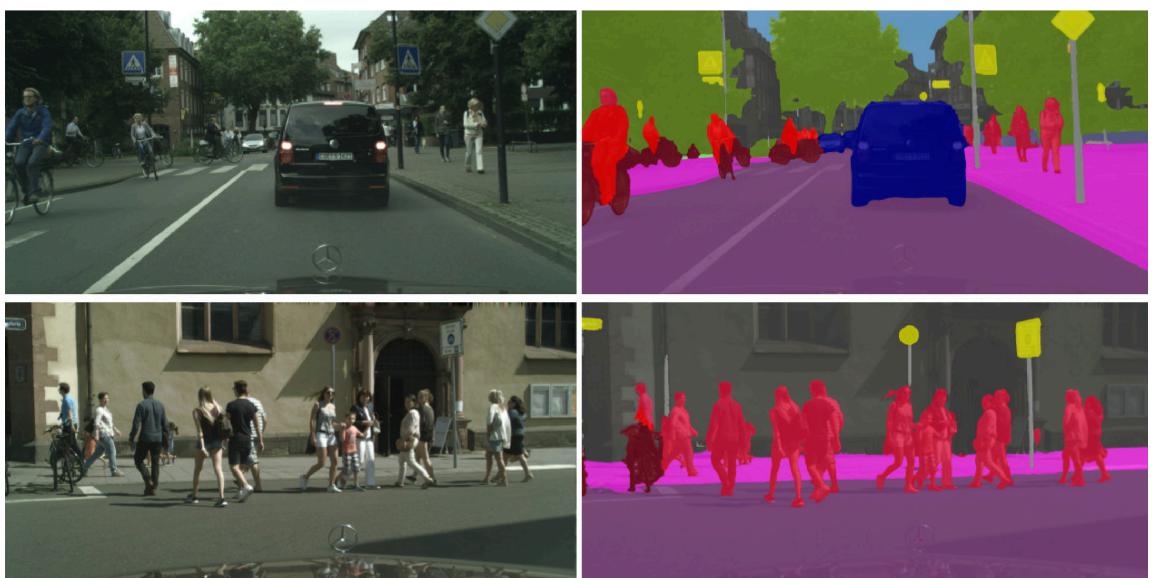
- - Dilation 因子 d (这里 $d = 2$)，使得感受野变得更加“分布式”，从而捕获更大的上下文信息
 - 当 $d = 1$ 时等同于原始的卷积方式
 - 这种方法可以在不增加参数量或降低空间分辨率的前提下，增加感受野——然而，代价是可能会引入一些伪影等问题
- 空洞卷积的结果展示如下



- U-Networks [Ronneberger, Fischer and Brox: U-Net: Convolutional Networks for Biomedical Image Segmentation. MICCAI, 2015.](#)

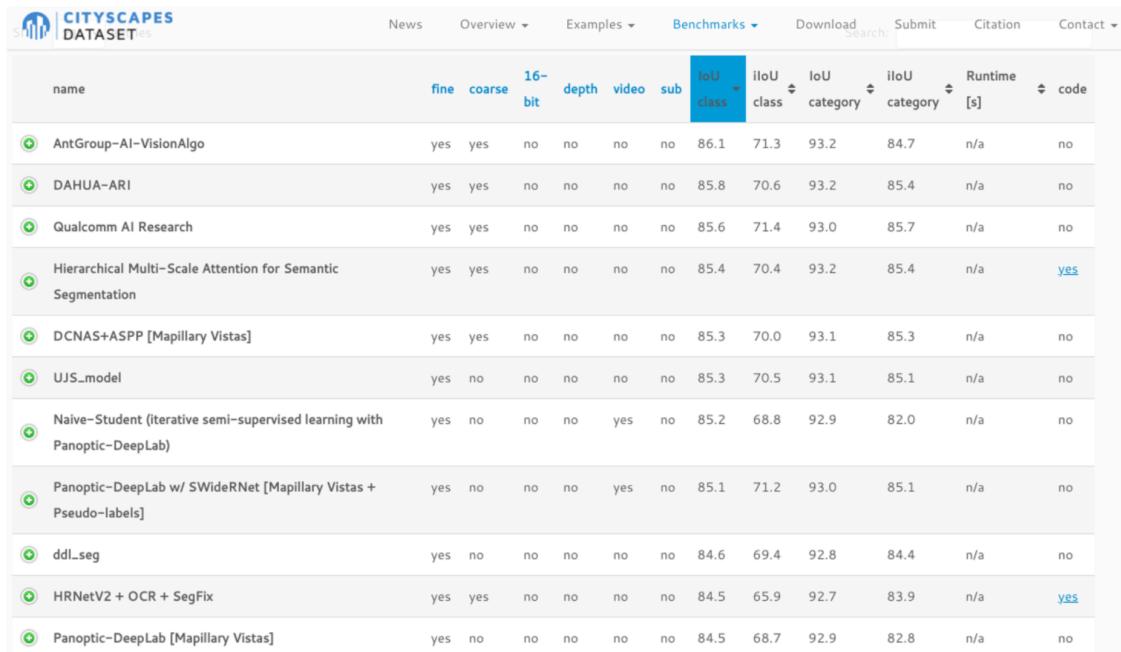


- 另一项非常相似的工作称为 U-Nets，它也采用 max-pooling、up-convolutions、skip-connections
 - 左侧通过一系列卷积和最大池化进行下采样
 - 然后通过最大反向池化，逐渐恢复特征图的分辨率
 - 每级同分辨率的图像进行跳跃连接，融合局部和全局特征
 - U-Net 现在已经是很多图像输出任务的标准选择（例如分割任务、深度预测等）
- Feature Space Optimization (特征空间优化) [Kundu, Vineet and Koltun: Feature Space Optimization for Semantic Video Segmentation. CVPR, 2016.](#)



- 这是另一篇工作，它针对于语义分割的连续性问题；例如视频中，帧与帧之间同一个物体的语义标签变化应该具有一致性——需要一种方法确保语义分割结果在时间上的平滑性和连续性
 - 这篇工作提出一种名为特征空间优化的方法，可以显著提升视频语义分割任务的质量，是视频理解中的关键技术——广泛用于自动驾驶、视频监控和虚拟现实等领域
- Cityscapes Leaderboard [Cordts et al.: The Cityscapes Dataset for Semantic Urban Scene Understanding. CVPR, 2016.](#)

- Cityscapes 数据集排行榜



The screenshot shows a table from the Cityscapes Leaderboard website. The table compares various semantic segmentation models across different evaluation metrics. The columns include: name, fine, coarse, 16-bit, depth, video, sub, IoU class, IoU category, IoU category, Runtime [s], and code. The rows list models such as AntGroup-AI-VisionAlgo, DAHUA-ARI, Qualcomm AI Research, Hierarchical Multi-Scale Attention for Semantic Segmentation, DCONAS+ASPP [Mapillary Vistas], UJS_model, Naive-Student (iterative semi-supervised learning with Panoptic-DeepLab), Panoptic-DeepLab w/ SWideRNet [Mapillary Vistas + Pseudo-labels], ddl_seg, HRNetV2 + OCR + SegFix, and Panoptic-DeepLab [Mapillary Vistas]. The table highlights some models with green icons and blue text.

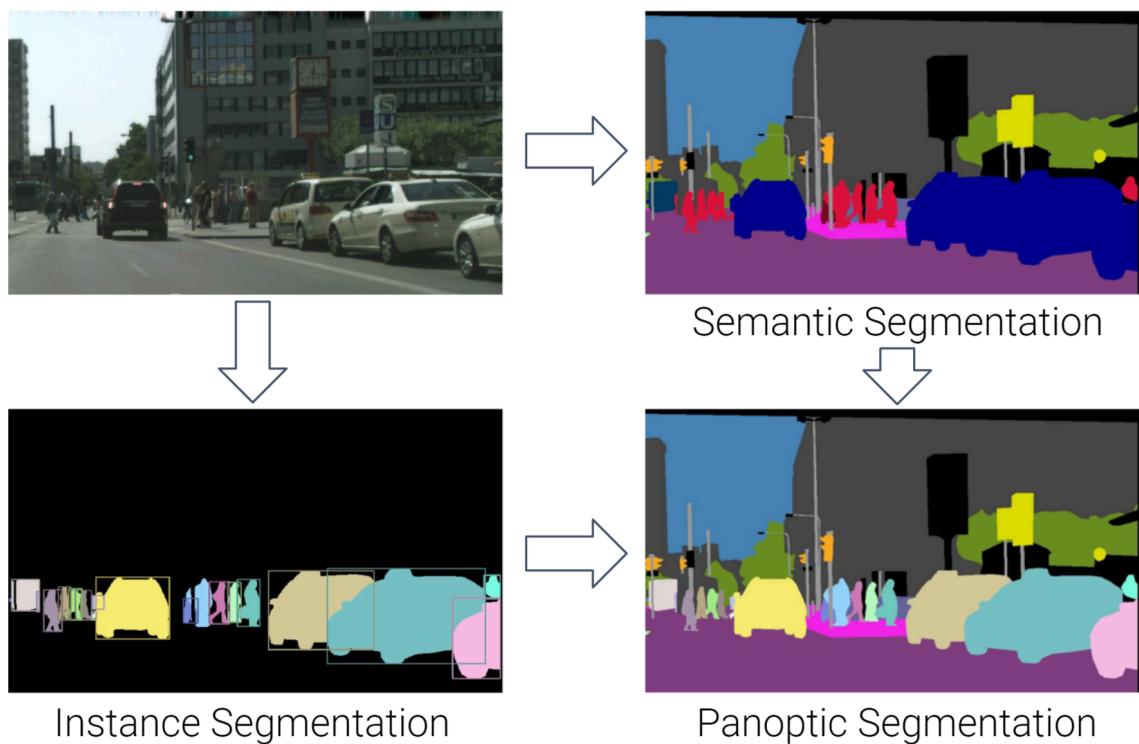
name	fine	coarse	16-bit	depth	video	sub	IoU class	IoU category	IoU category	Runtime [s]	code
AntGroup-AI-VisionAlgo	yes	yes	no	no	no	no	86.1	71.3	93.2	84.7	n/a
DAHUA-ARI	yes	yes	no	no	no	no	85.8	70.6	93.2	85.4	n/a
Qualcomm AI Research	yes	yes	no	no	no	no	85.6	71.4	93.0	85.7	n/a
Hierarchical Multi-Scale Attention for Semantic Segmentation	yes	yes	no	no	no	no	85.4	70.4	93.2	85.4	n/a
DCONAS+ASPP [Mapillary Vistas]	yes	yes	no	no	no	no	85.3	70.0	93.1	85.3	n/a
UJS_model	yes	no	no	no	no	no	85.3	70.5	93.1	85.1	n/a
Naive-Student (iterative semi-supervised learning with Panoptic-DeepLab)	yes	no	no	no	yes	no	85.2	68.8	92.9	82.0	n/a
Panoptic-DeepLab w/ SWideRNet [Mapillary Vistas + Pseudo-labels]	yes	no	no	no	yes	no	85.1	71.2	93.0	85.1	n/a
ddl_seg	yes	no	no	no	no	no	84.6	69.4	92.8	84.4	n/a
HRNetV2 + OCR + SegFix	yes	yes	no	no	no	no	84.5	65.9	92.7	83.9	n/a
Panoptic-DeepLab [Mapillary Vistas]	yes	no	no	no	no	no	84.5	68.7	92.9	82.8	n/a

www.cityscapes-dataset.com

- Cityscapes是一个专注于城市交通场景的高质量数据集，广泛用于语义分割、实例分割等任务；数据集包含精细标注的城市道路图像，特别适合自动驾驶领域的研究
 - 评估指标
 - Per-pixel accuracy (逐像素准确率)
 - 计算模型预测的像素类别与实际类别的匹配比例，描述了模型在整个图像上正确预测的像素比例
 - Per-class accuracy (逐类别准确率)
 - 针对每个类别分别计算预测准确率，然后取平均值，用于衡量模型对所有类别的平均表现
 - Hierarchical Multi-Scale Attention [Tao, Sapra and Catanzaro: Hierarchical Multi-Scale Attention for Semantic Segmentation. Arxiv, 2020.](#)



- ■ 这篇工作提出一种称为 Hierarchical Multi-Scale Attention (分层多尺度注意力) 的方法，一种用于语义分割的前沿方法
 - 模型从不同尺度的输入图像中提取特征，从而兼顾全局语义信息和局部细节信息；
 - 引入注意力机制，通过关注输入图像中不同区域的重要性，增强模型对关键部分的分割能力
- Panoptic Segmentation [Kirillov, He, Girshick, Rother and Dollár: Panoptic Segmentation. CVPR, 2019.](#)

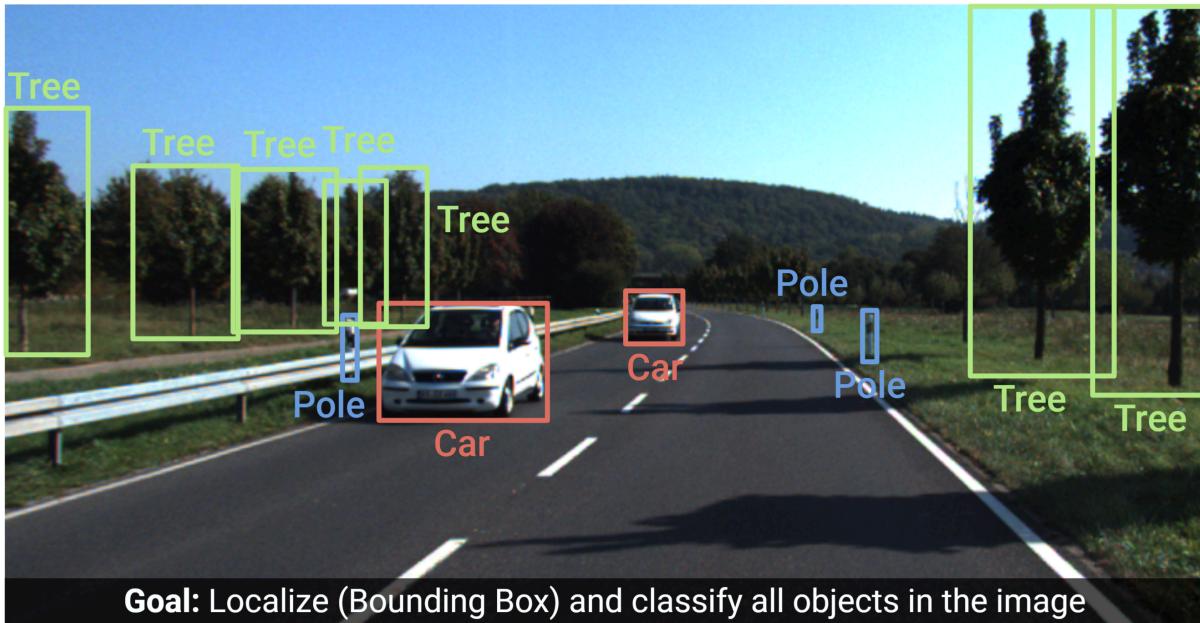


- ■ Panoptic Segmentation (全景分割) 是一种结合语义分割 (Semantic segmentation) 和实例分割 (Instance segmentation) 的统一框架
- 输入图像通过两个分支进行处理
 - 语义分割分支：预测静态区域，如天空、建筑
 - 实例分割分支：预测动态物体，如车辆、行人

- 全景分割，为每个像素分配语义类别和实例 ID，适用于需要对场景进行全局理解的任务，例如自动驾驶、机器人视觉和场景重建

10.3 Object Detection and Segmentation

- 首先我们再回顾一下 Object Detection 任务的目标



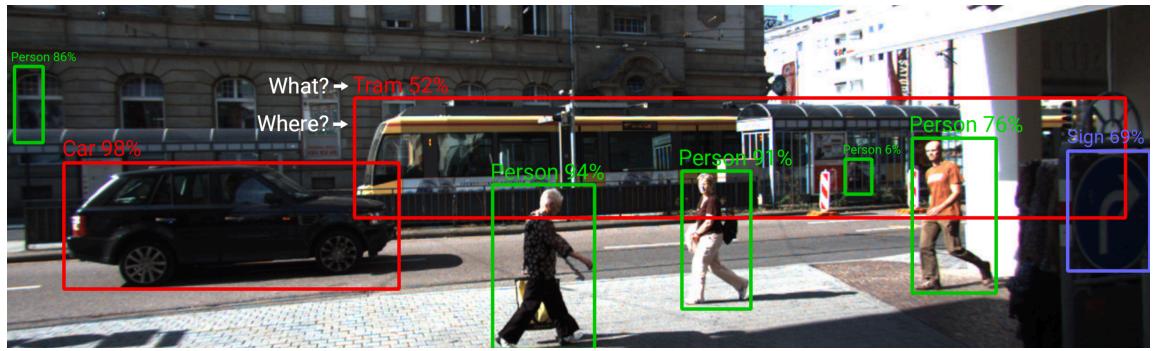
- 检测、框选所有对象物体，并且进行分类
- 这项工作的动机很多——几乎所有的包含视觉的 AI 任务都需要以此为基础，例如下图



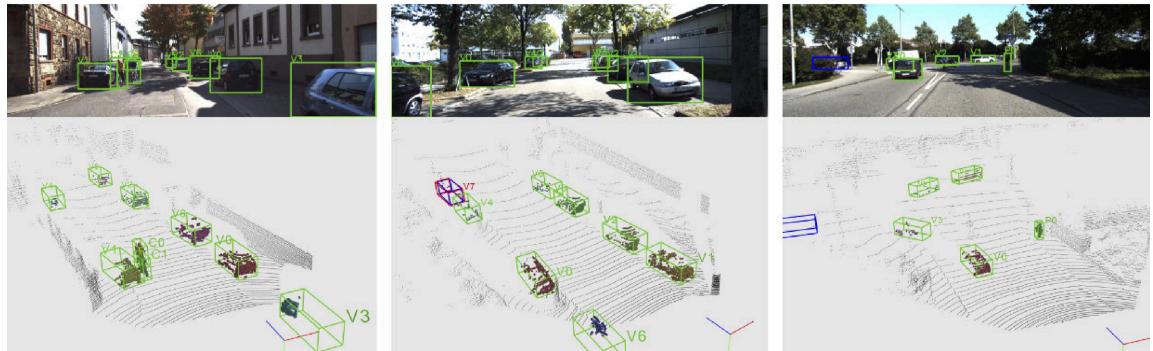
- 这是来自于 Mobileye 公司的一张图，来自于自动驾驶中的一项功能——检测车辆前方的其它车辆，并计算距离

接下来，我们考虑一下，该任务应该如何定义输入和输出

- 问题构建
 - 2D 情况



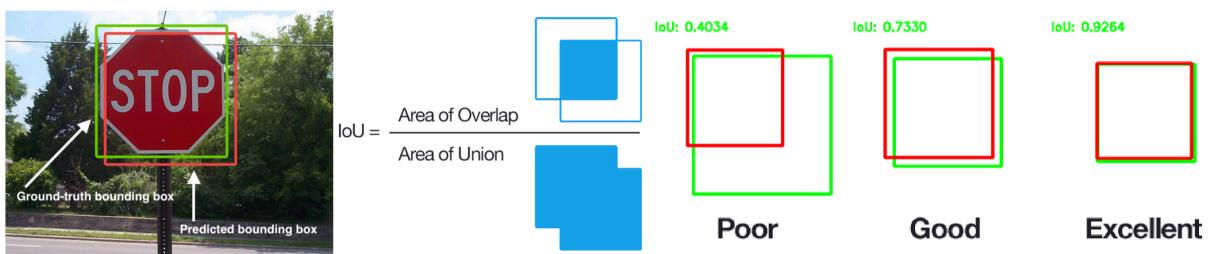
- 输入: RGB 图像, 或者激光扫描结果
- 输出: 一组 2D 边界框, 附带分类标签 (label) 和分类置信度 (confidence)
- 3D 情况



- 输入: RGB 图像, 或者激光扫描结果
- 输出: 一组 3D 边界框, 附带分类标签 (label) 和分类置信度 (confidence)
- 注意, 对于对象的数量和体积大小, 属于先验未知信息

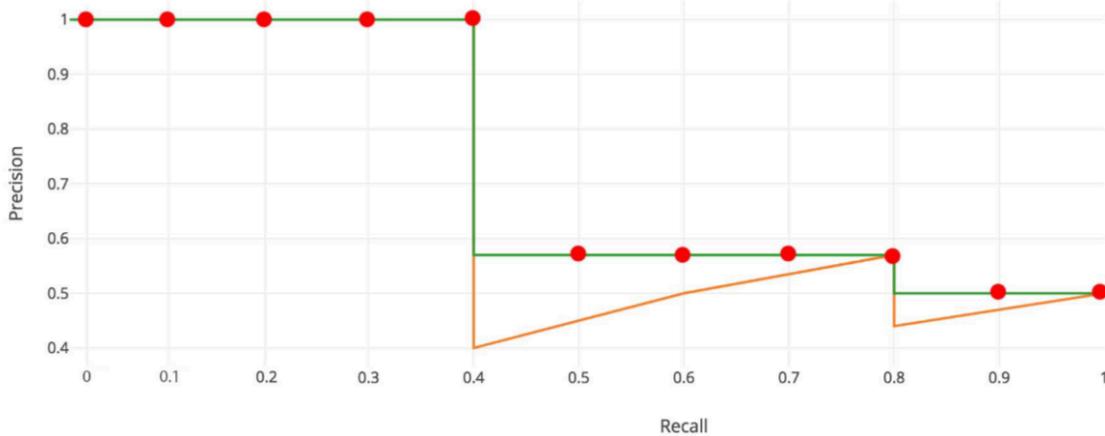
我们先从如何衡量性能表现入手

- Performance Evaluation



- 当然, 我们可以使用 Accuracy 作为一个衡量指标; 但是就可能会出现一种问题——如图中所示, 这个框的大小和类别可能非常贴近, 但是和它应该在的位置相差很远, 那么也不能算是一个很好的预测结果, 因此, 我们还会使用 IoU 作为衡量指标
- IoU (Intersection-over-Union) (图中红框为预测, 绿框为 gt)

- 一般来说，想要 IoU 完全重合是几乎不可能的，能够 ≥ 0.9 就已经是非常好的 detector 了；如果 ≥ 0.7 ，我们也认为是一个很好的 detector；如果它 ≤ 0.5 时，我们就认为这是一个不合格的结果
- 这些都是针对单个物体的——那么，对于多个物体的时候，我们应该如何选择一个公平的衡量指标呢？
-



- Average Precision Metric:

1. 对于不同的物体，在不同的阈值上运行 detector，返回对应每种物体下的检测结果
2. 我们将每一个检测结果分配到一个最接近的 gt 对象（分类目标）上，比如利用匈牙利方法在二分图中求解此问题
3. 计算 TP、FP、FN（混淆矩阵）
4. 计算 Average Precision (AP)

True Positives (TP): 正确检测到的数量 ($\text{IoU} \geq 0.5$)

False Negatives (FN): 没有检测到的数量 ($\text{IoU} < 0.5$)

False Positives (FP): 错误检测的数量

从而计算出

$$\text{Precision (P)} = \frac{TP}{TP+FP}$$

$$\text{Recall (R)} = \frac{TP}{TP+FN}$$

$$AP = \frac{1}{N} \sum_{R \in \{0, \dots, N\}} \max_{R' \geq R} P(R')$$

下面我们先从最简单的一种检测算法——滑动窗口法开始说起