华东师范大学

East China Normal University, ECNU

SEEK TRUTH. FOSTER ORIGINALITY AND
LIVE UP TO THE NAME OF TEACHER

第十五章. 智能识别技术

01 智能识别的概念基础

02 识别技术发展中的主要问题

03 现有的智能识别的典型算法

04 识别算法实例分析

01

智能识别的概念基础

目标检测
（Object Detection）

◆ **目标识别/检测**（Object Detection）是找出图像中所有感兴趣的目标（物体），确定它们的**类别**和**位置**，是计算机视觉领域的核心问题之一。

◆ 由于各类物体有不同的外观、形状和姿态，加上成像时光照、遮挡等因素的干扰，目标检测一直是计算机视觉领域最具有挑战性的问题。

目标检测
（Object Detection）

计算机视觉中关于图像识别有四大类任务：

（1）分类-Classification：解决“是什么？”的问题，即给定一张图片或一段视频判断里面包含什么类别的目标。

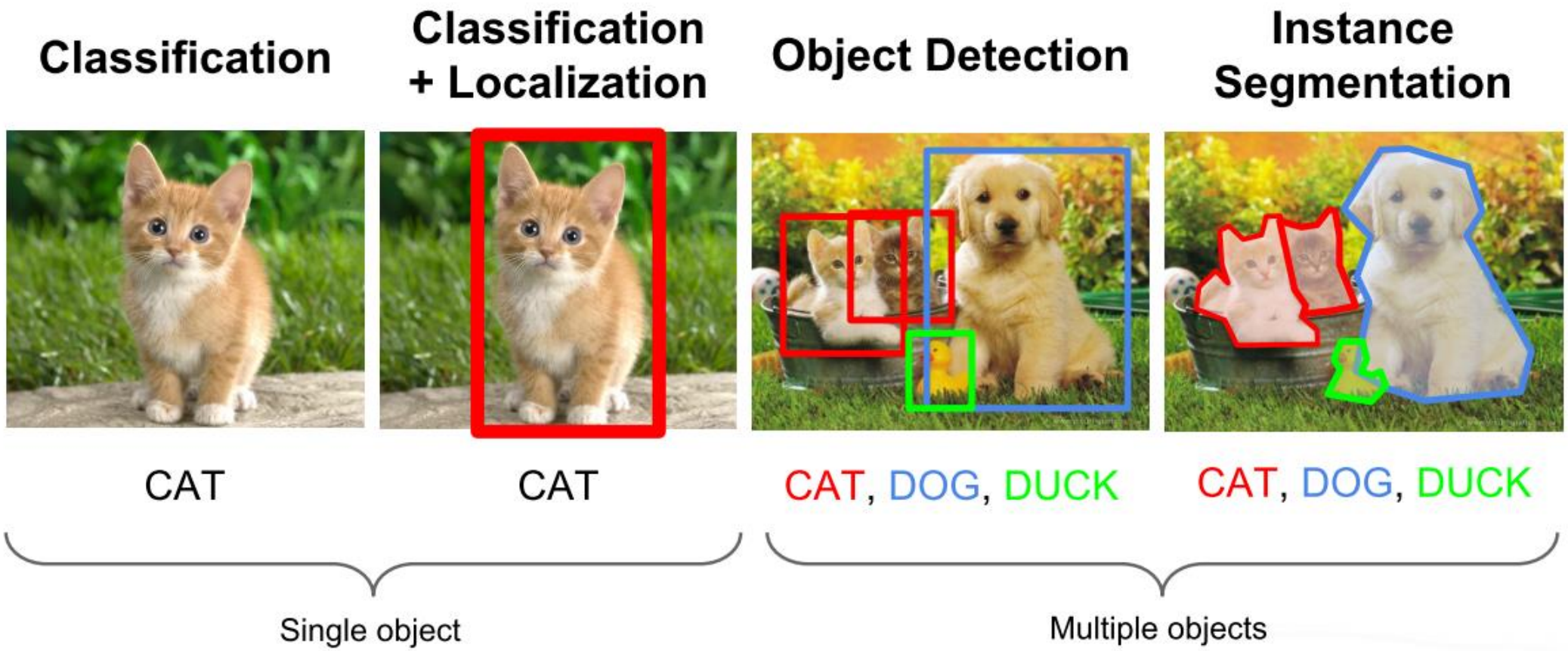
（2）定位-Location：解决“在哪里？”的问题，即定位出这个目标的的位置。

（3）检测-Detection：解决“在哪里？是什么？”的问题，即定位出这个目标的位置并且知道目标物是什么。

（4）分割-Segmentation：分为实例的分割（Instance-level）和场景分割（Scene-level），解决“每一个像素属于哪个目标物或场景”的问题。

所以，目标检测是一个分类、回归问题的叠加。

目标检测
（Object Detection）



目标检测的核心问题：

（1）**分类问题**：即图片（或某个区域）中的图像属于哪个类别。

（2）**定位问题**：目标可能出现在图像的任何位置。

（3）**大小问题**：目标有各种不同的大小。

（4）**形状问题**：目标可能有各种不同的形状。

目标识别应用：

视频监控

智能汽车

场景搜索

基于内容的图像\视频检索

医学图像分析

02

识别技术发展中的主要问题

如何鲁棒识别？

光照的影响

物体姿态的影响

背景混淆

遮挡

类内差异

视点的影响

类 内 差 异 （ intra-class variability）

Many face of Madonna

类 间 相 似 性 （ inter-class similarity）

www.marykateandashley.com

news.bbc.co.uk/hi/english/in_depth/americas/2000/us_elections

双胞胎

父子

计算量大

一幅图像中像素个数多，目前每秒约产生300G像素的图像/视频数据。

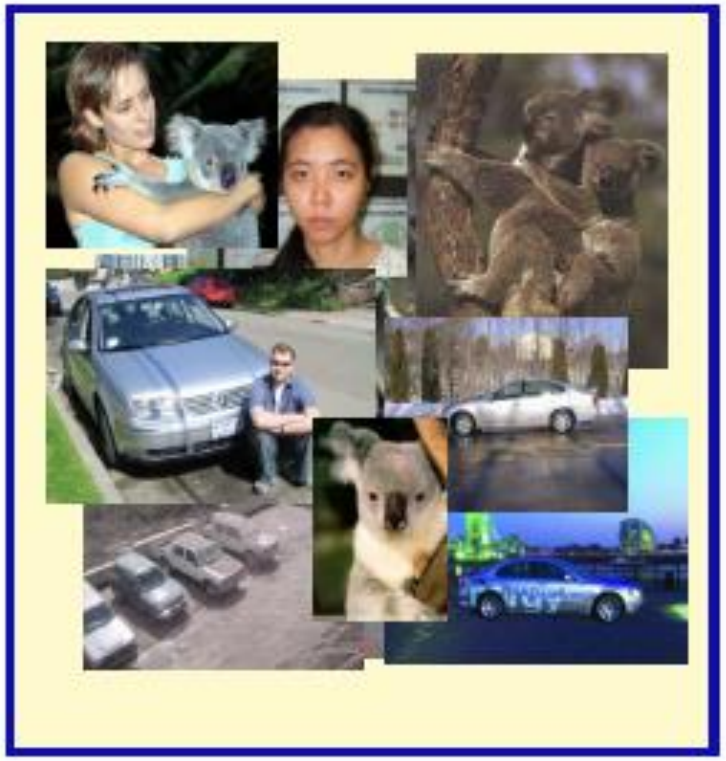
- Google图片搜索中已有几十亿幅图像
- 全球数字照相机一年产生180亿张以上的图片（2004年）
- 全球一年销售约3亿部照相手机（2005）

人的物体识别能力是强大的

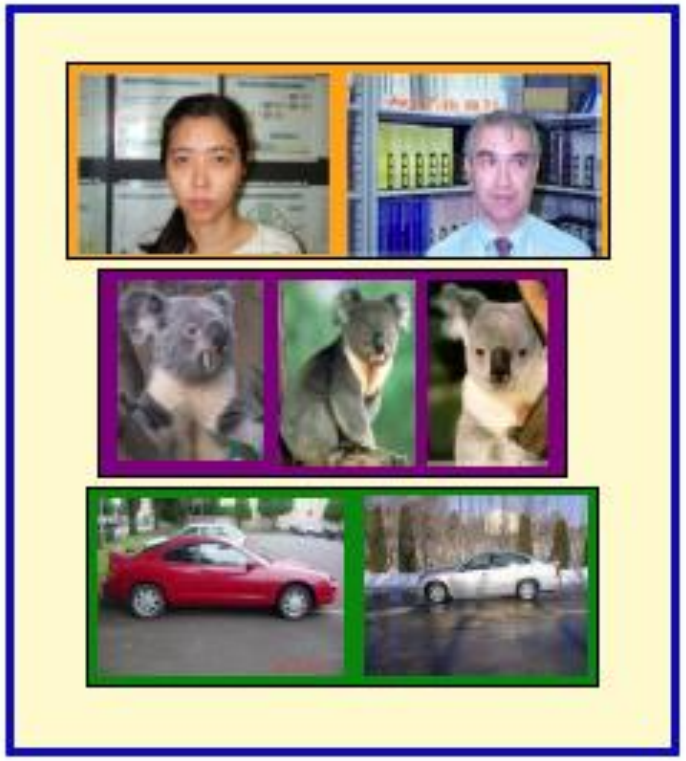
- 灵长类动物约使用大脑皮层的一半来处理视觉信息 [Felleman and van Essen 1991]
- 可以识别3,000-30,000种物体
- 物体姿态可允许30度以上的自由度。

如何在小样本条件下学习

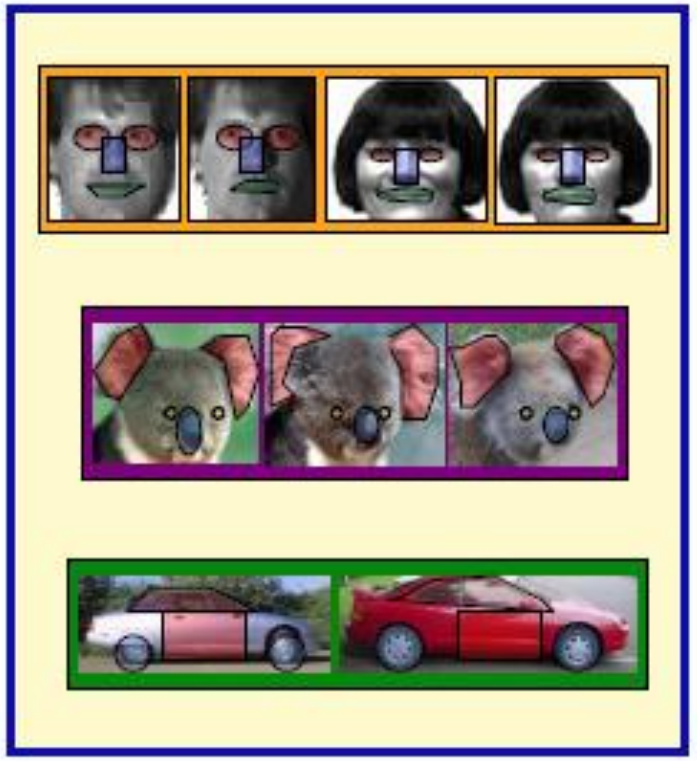
低 （人为监督学习的复杂程度） 高



无标注，多物体



图像整体标注，有背景混淆



物体标注（分割到物体甚至部件）

03

现有的智能识别的典型算法

目标检测算法分类

基于深度学习的目标检测算法主要分为两类：Two stage和One stage

◆ **Tow Stage：**先进行区域生成，该区域称之为region proposal（简称RP，一个有可能包含待检物体的预选框），再通过卷积神经网络进行样本分类。

➢ **任务流程：**

特征提取 --> 生成RP --> 分类/定位回归。

➢ **常见tow stage目标检测算法有：**

R-CNN、SPP-Net、Fast R-CNN、Faster R-CNN和R-FCN等。

目标检测算法分类

基于深度学习的目标检测算法主要分为两类：Two stage和One stage

◆ **One Stage：** 不用RP，直接在网络中提取特征来预测物体分类和位置。

➢ **任务流程：**

特征提取—> 分类/定位回归。

➢ **常见的one stage目标检测算法有：**

OverFeat、YOLOv1、YOLOv2、YOLOv3、SSD和RetinaNet等

候选区域产生方法：

1）区域建议方法

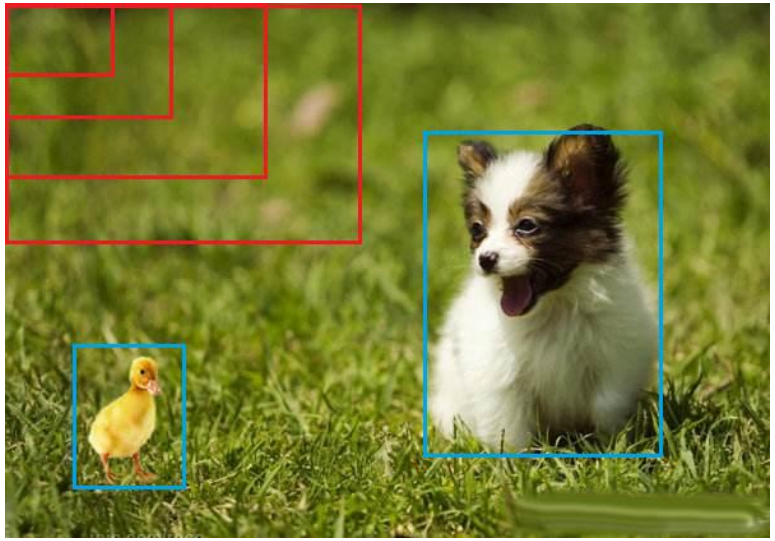
- 滑动窗口法
- 超像素分组法：选择性搜索(selective search)，CPMC，MCG等

2）区域建议网络RPN

候选区域产生方式

滑动窗口：

- 首先，采用**不同窗口大小**的滑窗，对输入图像进行从左往右、从上到下的滑动。
- 对当前窗口执行**分类**(分类器是事先训练好的)。如果当前窗口得到较高的分类概率，则认为检测到了物体。
- 不同尺度窗口检测到同个目标的多个标记，会存在较高的重叠部分，最后采用**非极大值抑制**(Non-Maximum Suppression, NMS)方法进行筛选。最终，经过NMS筛选后获得检测物体。



候选区域产生方式

选择性搜索：

选择搜索（selective search，简称SS）由Koen E.A于2011年提出。

主要思想：图像中物体可能存在的区域应该是有某些相似性或者连续性区域的。因此，选择搜索采用**子区域合并**的方法进行提取bounding boxes。

候选区域产生方式

选择性搜索：

- 首先，对输入图像进行分割算法产生初始的分割区域。
- 其次，根据区域之间的相似性，进行区域合并，不断的进行区域迭代合并。

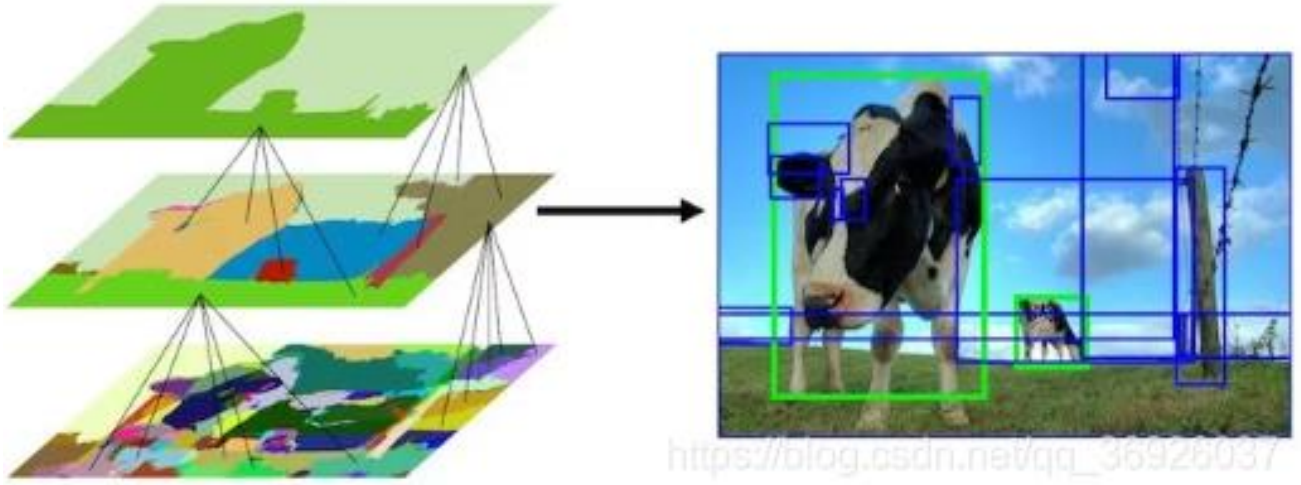
  每次迭代过程中对合并的子区域做bounding boxes(外切矩形)，这些子区域**外切矩形**就是通常所说的**候选框**。

候选区域产生方式

选择性搜索流程:

- Step1：首先，将所有分割区域的外框加到候选区域列表中；
- Step2：基于相似度（如颜色、纹理、大小和形状交叠）合并一些区域；
- Step3：将合并后的分割区域作为一个整体，跳到步骤1。

通过不停的迭代，候选区域列表中的区域越来越大。最后得到目标检测的候选区域。



候选区域产生方式

选择性搜索优点：

- 计算效率优于滑动窗法
- 由于采用子区域合并策略，所以可以包含各种大小的疑似物体框
- 合并区域相似的指标多样性，提高了检测物体的概率

数据表示

经过标记后的样本数据如下所示：

数据表示

预测输出可以表示为：

$$y=\begin{bmatrix}p_c\\b_x\\b_y\\b_w\\b_h\\C_1\\C_2\\C_3\end{bmatrix},y_{true}=\begin{bmatrix}1\\40\\45\\80\\60\\0\\1\\0\end{bmatrix},y_{pred}=\begin{bmatrix}0.88\\41\\46\\82\\59\\0.01\\0.95\\0.04\end{bmatrix}$$

其中，

p_c：为预测结果的置信概率；

b_x, b_y, b_w, b_h：为边框坐标；

C_1, C_2, C_3：为属于某个类别的概率。

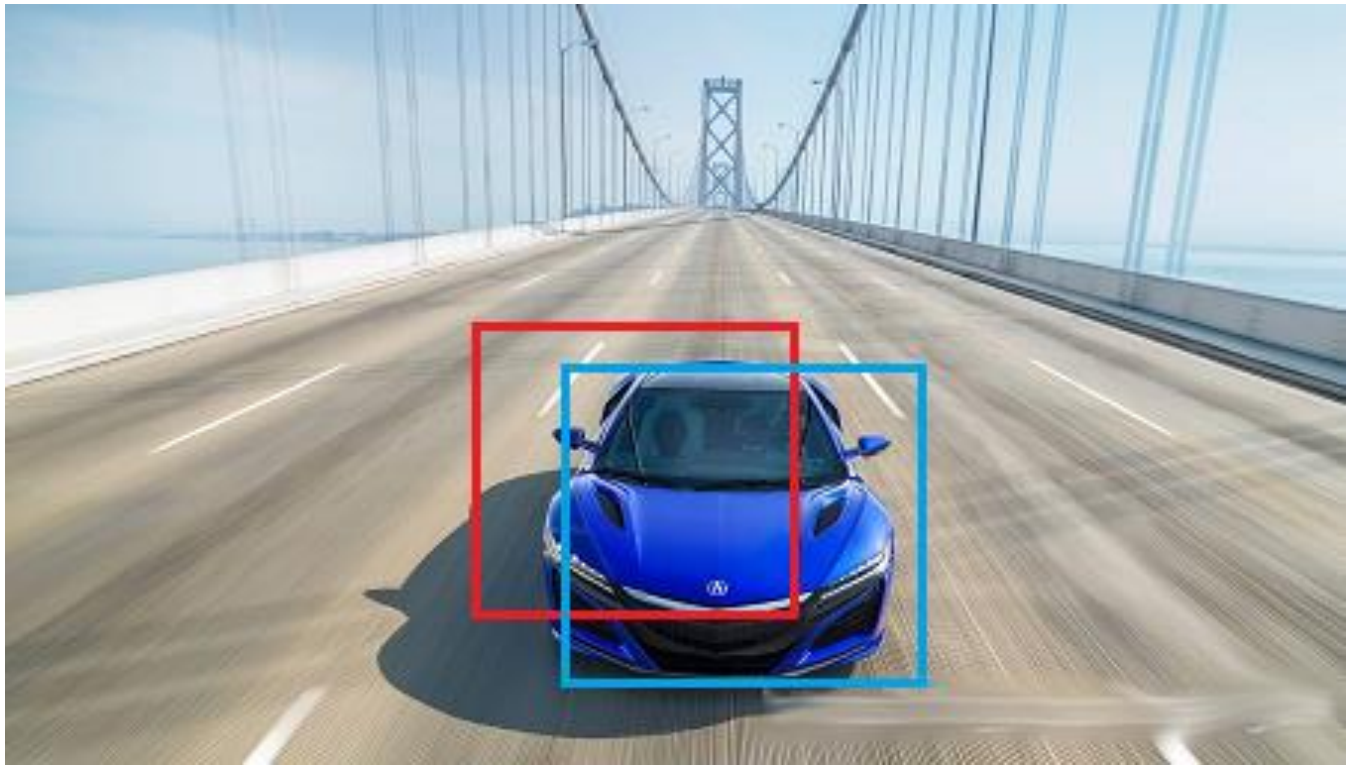
通过预测结果、实际结果，构建损失函数。

损失函数包含了**分类**、**回归**两部分组成。

效果评估

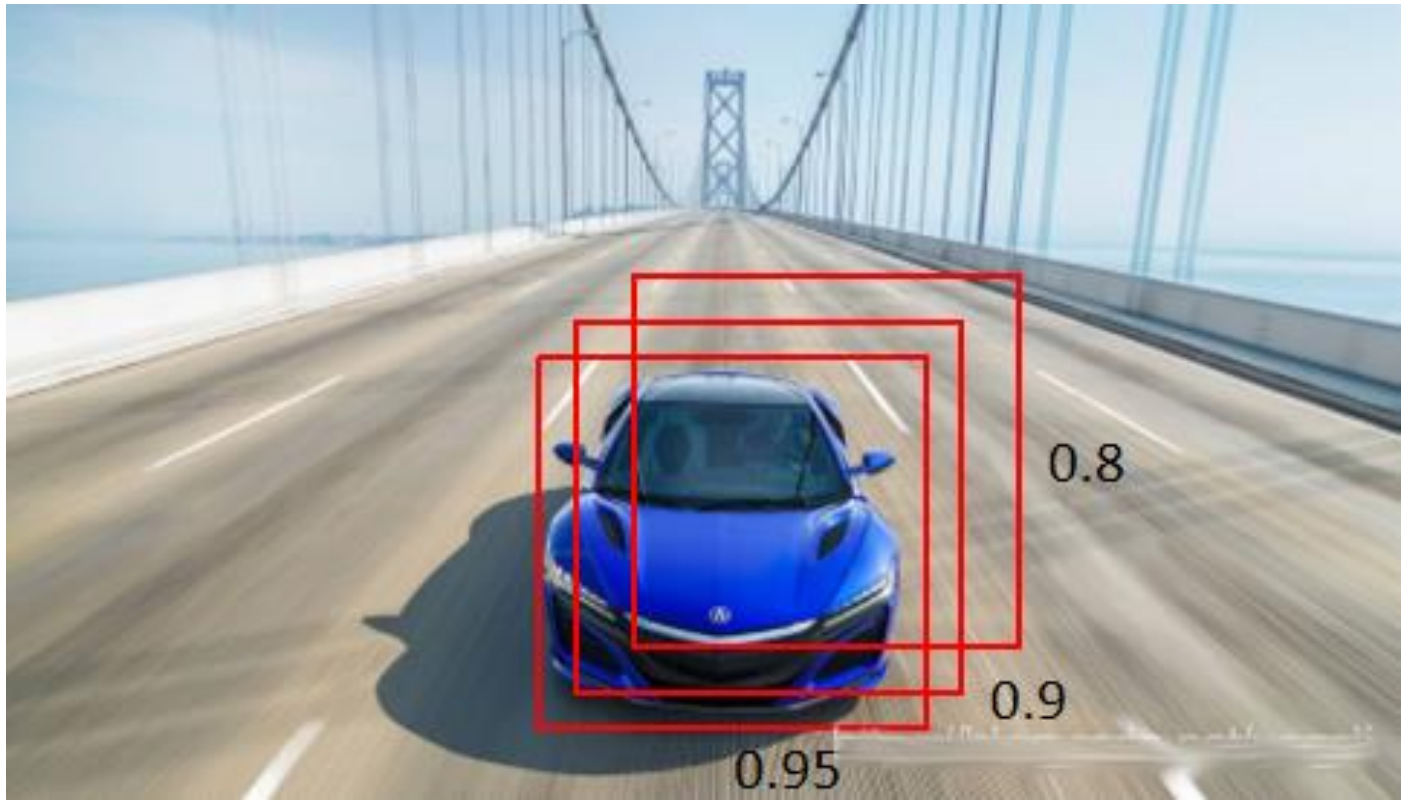
使用IoU（Intersection over Union，交并比）来判断模型的优劣。

交并比：指预测边框、实际边框交集和并集的比率，一般约定0.5为一个可以接收的值。



非极大值抑制

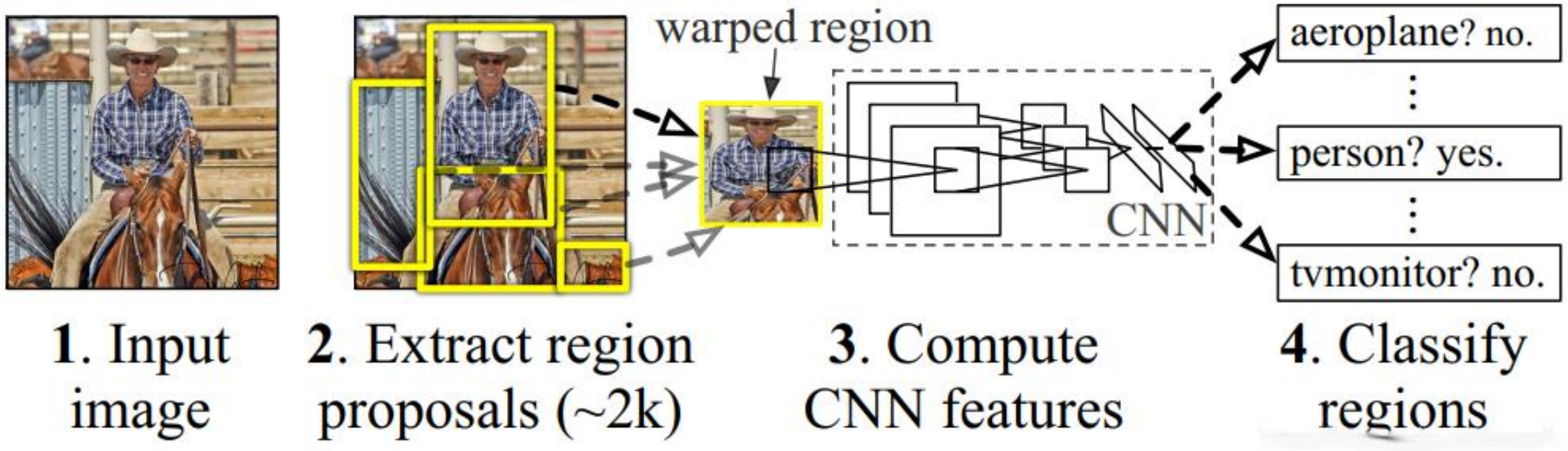
预测结果中，可能多个预测结果间存在重叠部分，需要保留交并比最大的、去掉非最大的预测结果，即，**非极大值抑制**（Non-Maximum Suppression: NMS）。



R-CNN

- R-CNN(全称Regions with CNN features)，是R-CNN系列的第一代算法。
- R-CNN pipeline中的第二步和第四步属于传统的“计算机视觉”技术。

使用selective search提取region proposals，使用SVM实现分类。



R-CNN

R-CNN流程：

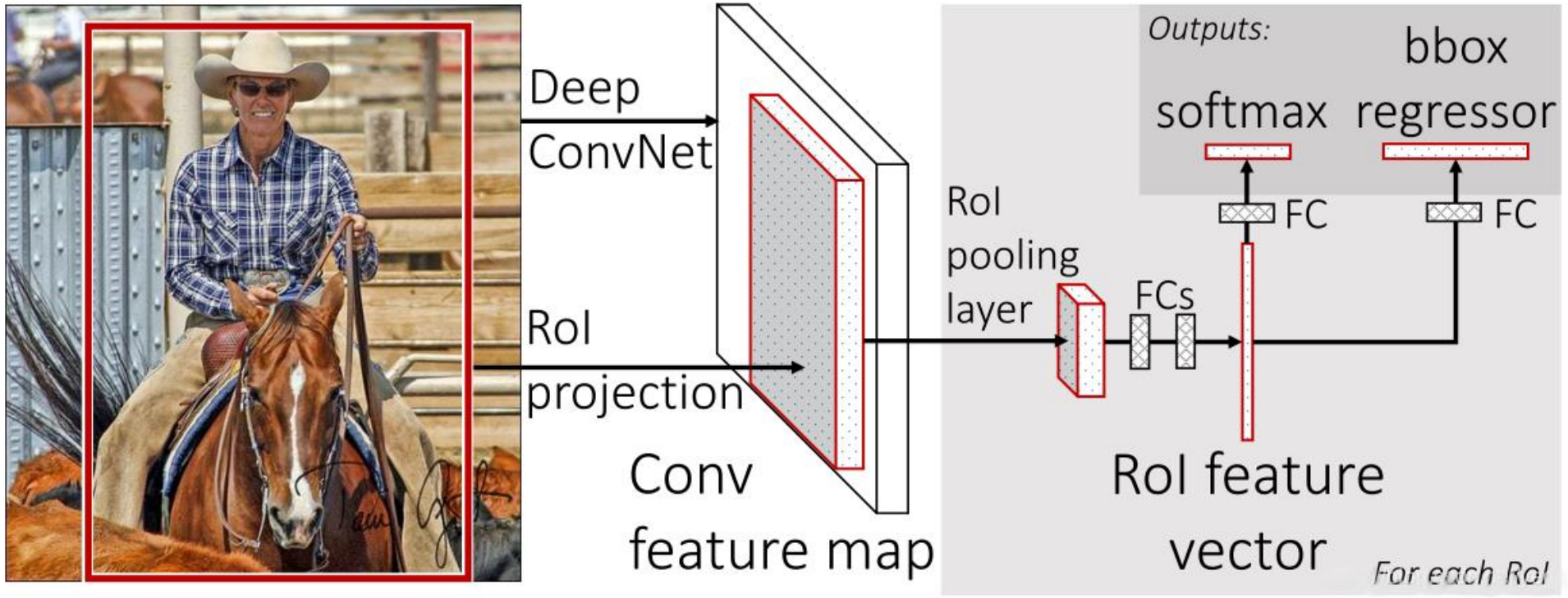
- **预训练模型：**选择一个预训练（pre-trained）神经网络（如AlexNet、VGG）。
- **重新训练全连接层。**使用需要检测的目标重新训练（re-train）最后全连接层。
- **提取 proposals并计算CNN 特征：**利用选择性搜索（Selective Search）算法提取所有proposals（大约2000幅），并调整（resize/warp）为固定大小，以满足CNN输入要求（因为全连接层的限制），然后将feature map 保存到本地磁盘。
- **训练SVM：**利用feature map 训练SVM来对目标和背景进行分类。
- **边界框回归**（Bounding boxes Regression）：训练将输出一些校正因子的线性回归分类器。

R-CNN缺点：

- 重复计算，每个region proposal，都需要经过一个AlexNet特征提取，为所有的RoI（region of interest）提取特征大约47秒。
- selective search方法生成region proposal，对一帧图像，需要花费2秒
- 三个模块（**提取**、**分类**、**回归**）是分别训练的，并且在训练时候，对于存储空间消耗较大。

Fast R-CNN

- Fast R-CNN是基于R-CNN和SPPnets进行的改进。
- SPPnets创新点在于：**只进行一次图像特征提取**（而不是每个候选区域计算一次），然后**将候选区域特征图映射到整张图片特征图中**。



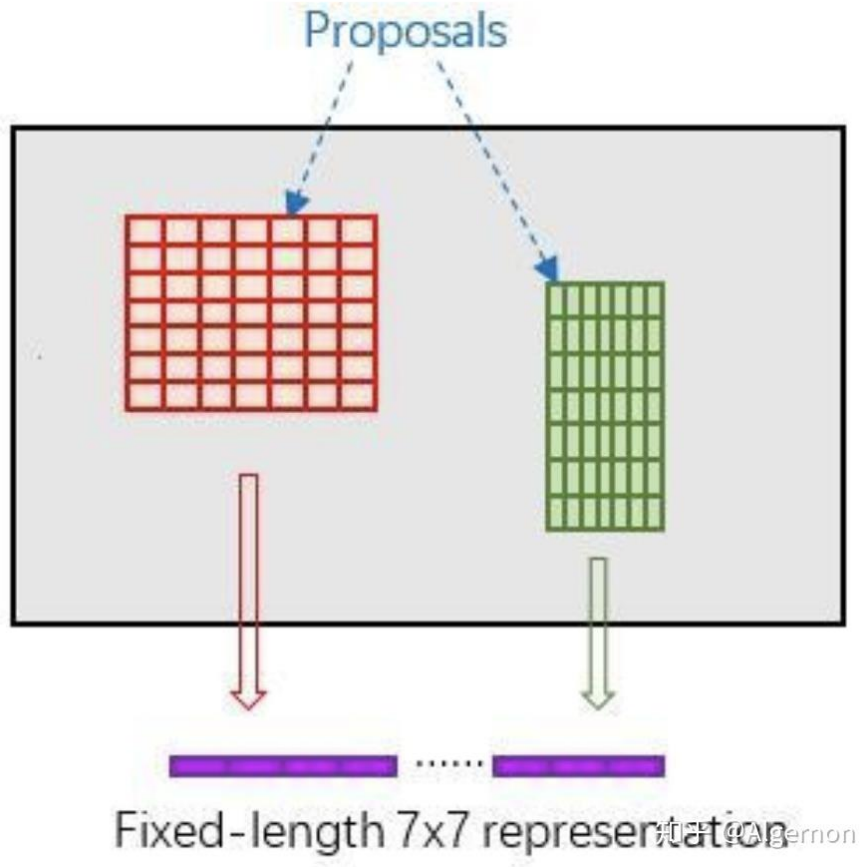
Fast R-CNN

Fast R-CNN流程：

- 使用selective search生成region proposal，大约2000个左右区域候选框。
- (joint training)缩放图片的scale得到图像金字塔，FP得到conv5的特征金字塔。
- (joint training)对于每个scale的每个ROI，求取映射关系，在conv5中剪裁出对应的patch。并用一个单层的SSP layer来统一到一样的尺度（对于AlexNet是6*6）
- (joint training) 经过两个全连接得到的特征，又分别共享到两个新的全连接，连接上两个优化目标：1）分类，使用softmax；2）bbox regression，使用平滑的L1-loss。
- 测试时需要加上NMS处理：利用窗口得分分别对每一类物体进行非极大值抑制提出重叠建议框，最终得到每个类别中回归修正后的得分最高的窗口

RoI pooling:

- 输入： h × w
- 输出： H × W
- 每个子窗口： h/H × w/W,
- 子窗口内： **max-pooling**
- 每个通道独立进行pooling;



RPN在特征图中会产生尺寸不一致的Proposal区域，ROI pooling对任意大小的Proposal，都pooling成了7*7的尺寸。

Fast R-CNN

改进：

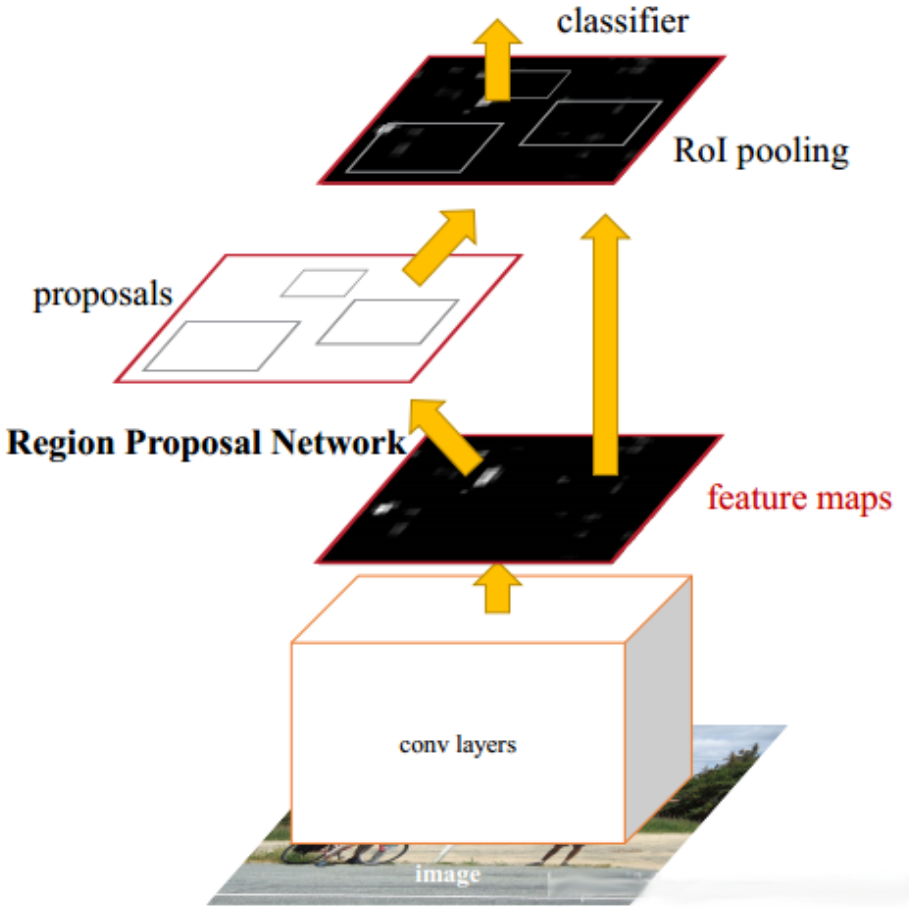
- 和RCNN相比，训练时间：84小时-->9.5小时，测试时间:47秒--> 0.32秒。在VGG16上，训练/测试速度是RCNN的9倍/213倍。训练/测试速度是SPP-net的3倍。
- 加入RoI Pooling，采用一个神经网络对全图提取特征。
- 在网络中加入了多任务函数边框回归，实现了端到端的训练。

缺点：

- 依旧采用selective search提取region proposal（耗时2~3秒，特征提取耗时0.32秒）
- 无法满足实时应用，没有真正实现端到端训练测试。
- 利用了GPU，但是region proposal方法是在CPU上实现的。

Faster RCNN

Ross B.Girshick在2016年提出了新的Faster RCNN，在结构上将**特征抽取**、**region proposal提取**， **bbox regression**，**分类**都整合到了一个网络中，使得综合性能有较大提高，在检测速度方面尤为明显。



Faster RCNN

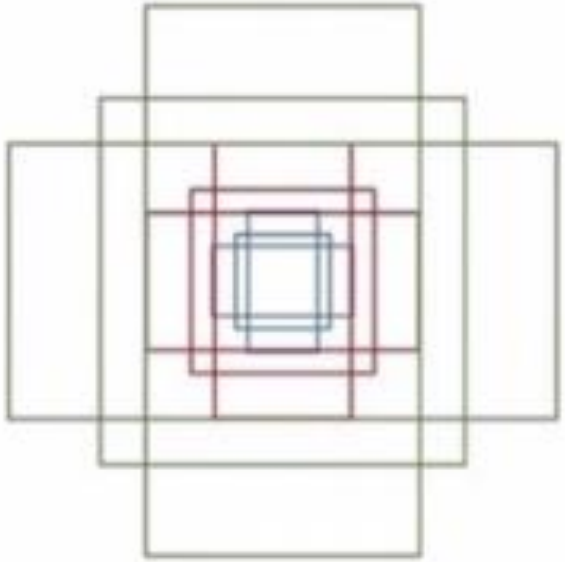
Faster R-CNN流程：

- **Conv Layers**：Faster RCNN首先使用一组基础的卷积/激活/池化层提取图像的特征，形成一个特征图，用于后续的RPN层和全连接层。
- **Region Proposal Networks（RPN）**：RPN网络用于生成候选区域，该层通过softmax判断锚点（anchors）属于前景还是背景，再利用bounding box regression（包围边框回归）获得精确的候选区域。
- **RoI Pooling**：收集输入的特征图和候选区域，综合这些信息，提取候选区特征图（proposal feature maps），送入后续全连接层判定目标的类别。
- **Classification**：利用取候选区特征图计算所属类别，并再次使用边框回归算法获得边框最终的精确位置。

Faster RCNN

Anchors（锚点）：指由一组矩阵，每个矩阵对应不同的检测尺度大小。如下矩阵：

```
[[-84.   -40.   99.   55.]
 [-176.  -88.   191.  103.]
 [-360.  -184.  375.  199.]
 [-56.   -56.   71.   71.]
 [-120.  -120.  135.  135.]
 [-248.  -248.  263.  263.]
 [-36.   -80.   51.   95.]
 [-80.   -168.  95.   183.]
 [-168.  -344.  183.  359.]]
```

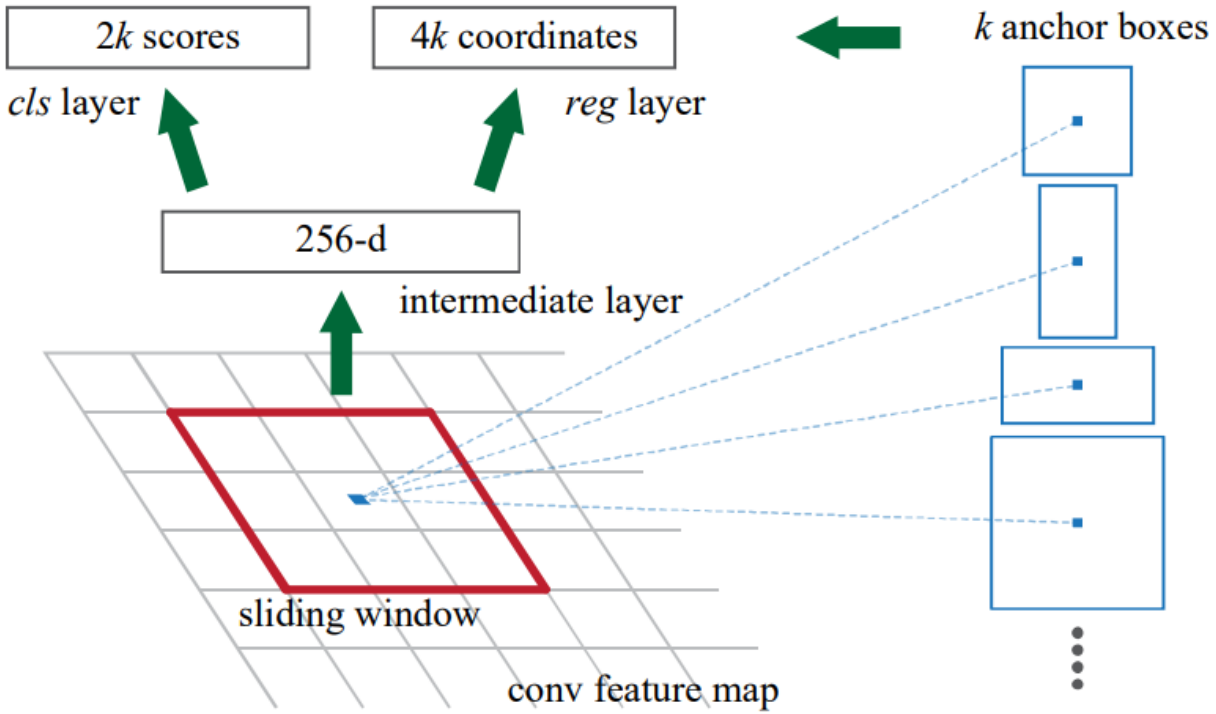


- 其中，每行4个值（x_1, y_1, x_2, y_2），对应矩形框左上角、右下角相对于中心点的偏移量。9个矩形共有三种形状，即1:1, 1:2, 2:1，即进行多尺度检测。
- 例如，一张800*600的影像，经过VGG下采样16倍大小，即50*38，每个点设置9个anchor，则总数为：

ceil(800 / 16) * ceil(600 / 16) * 9 = 50 * 38 * 9 = 17100

Region Proposal Networks：

sliding window，是一个kernal size为3*3的卷积层。通过两个1*1的卷积层，输出两个特征图，即图中的2k scores、4k coordinates。k：anchor框个数



Bounding box regression：

物体识别完成后，通过一种方式对外围框进行调整，使得和目标物体更加接近。

损失函数 = 分类损失函数 + 回归损失函数

$$L(\{p_i,\{t_i\}) = \frac{1}{N_{cls}}\sum L_{cls}(p_i,p_i^*) + \lambda\frac{1}{N_{reg}}\sum p_i^* L_{reg}(t_i,t_i^*)$$

i是mini-batch中一个anchor的索引；p_i是anchor i为目标的预测概率(1/0)；

t_i是预测的包围盒的4个参数化坐标向量；

N_{cls}是Mini-batch的大小；N_{reg}为anchor位置的数量（大约2400）；$\lambda = 10$

Faster RCNN

分类损失函数：

$$L_{cls}(p_i, p_i^*) = -\log[p_i^* p_i + (1 - p_i^*)(1 - p_i)]$$

位置损失函数：

$$L_{reg}(t_i, t_i^*) = R(t_i - t_i^*)$$

$$R = smooth_{L1}(x) = \begin{cases} 0.5x^2 & if|x| < 1 \\ |x| - 0.5 & otherwise \end{cases}$$

Faster RCNN

改进：

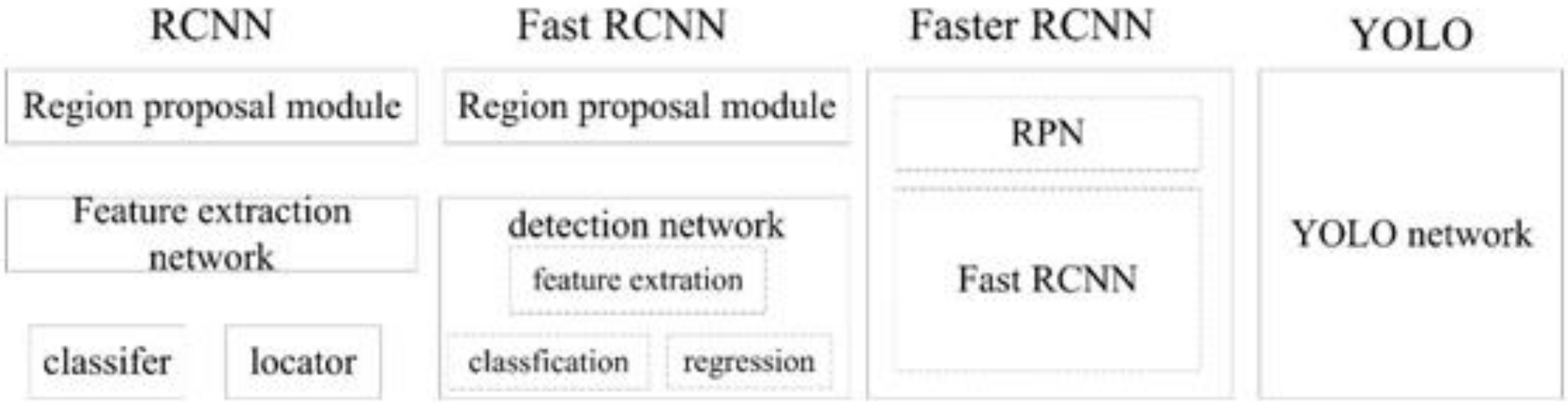
- 在VOC2007测试集测试mAP达到73.2%，目标检测速度可达5帧/秒
- 提出Region Proposal Network(RPN)，取代selective search，生成待检测区域，时间从2秒缩减到了10毫秒
- 真正实现了一个完全的End-To-End的CNN目标检测模型
- 共享RPN与Fast RCNN的特征

缺点：

- 还是**无法**达到**实时**检测目标
- 获取region proposal， 再对每个proposal分类计算量还是较大

YOLO: You Only Look Once

基本思想： 生成RoI+目标检测两阶段（two-stage）算法用一套网络的一阶段（one-stage）算法替代，直接在输出层回归bounding box的位置和所属类别。

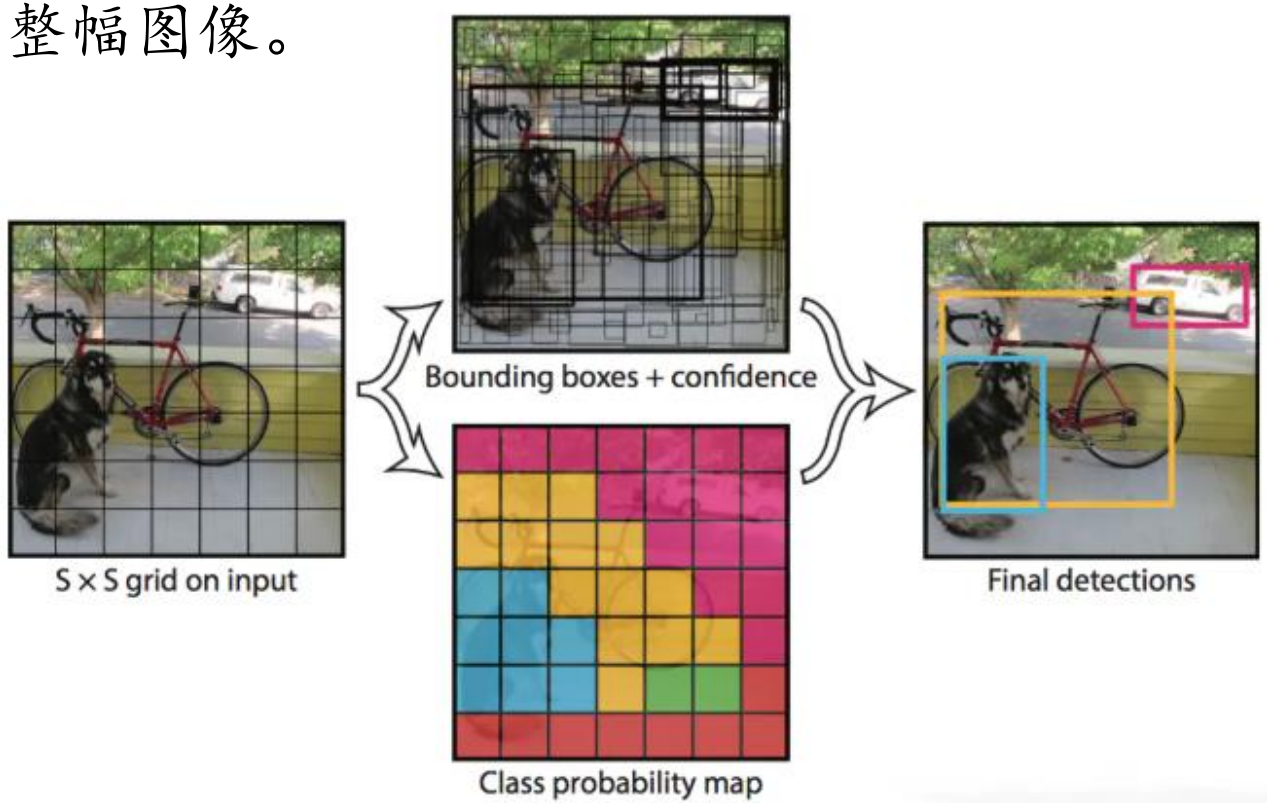


YOLOv1

实际上，YOLO并没有真正去掉候选区，而是采用了**预定义候选区**的方法，即：

- 将图片划分为7*7个网格，
- 每个网格允许预测出2个边框，总共49*2个bounding box，

即，98个候选区域，很粗略地覆盖了整幅图像。



YOLOv1

- 每个网格单元预测2个边界框和置信度分数。
- **置信度分数**反映了：预测框包含目标的可靠程度，以及预测框的准确程度。置信度定义为：

$$\Pr(Object) * IOU_{pred}^{truth}$$

- 如果该单元格中不存在目标，则置信度分数应为零。
- 否则，我们希望置信度分数等于预测框与真实值之间重叠部分的交并比（IOU）。

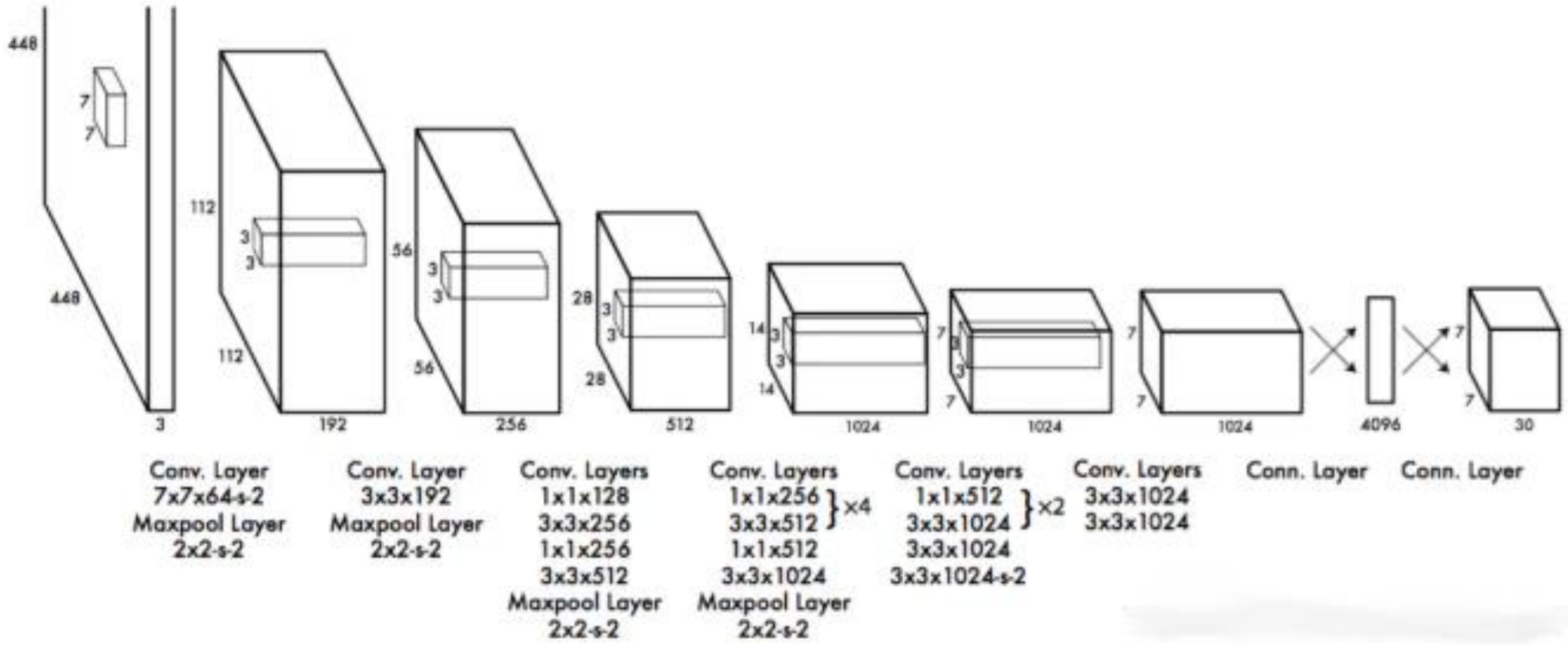
YOLOv1

- 每个边界框包含5个预测：x，y，w，h 和置信度。
  - (x, y)：表示边界框相对于网格单元边界框的中心。
  - w和h：是相对于整张图像预测的；
  - 置信度预测：预测框与实际边界框之间的IOU。

- 每个网格单元还预测C 个条件类别概率$\Pr(Class_i|Object)$。这些概率以包含**目标的网格单元**为条件。

YOLOv1

网络结构：有24个卷积层+2个全连接层。使用1×1 降维层，后面是3×3 卷积层。



训练过程与细节：

（1）**预训练**。前20个卷积层、平均池化层、全连接层大约一周的预训练；

（2）**输入**。输入数据为224*224和448*448大小的图像；

（3）**采用相对坐标**。通过图像宽度和高度来规范边界框的宽度和高度，取值为0~1；边界框x 和y 坐标参数为特定网格单元位置的偏移量，取值为0~1；

（4）**损失函数**

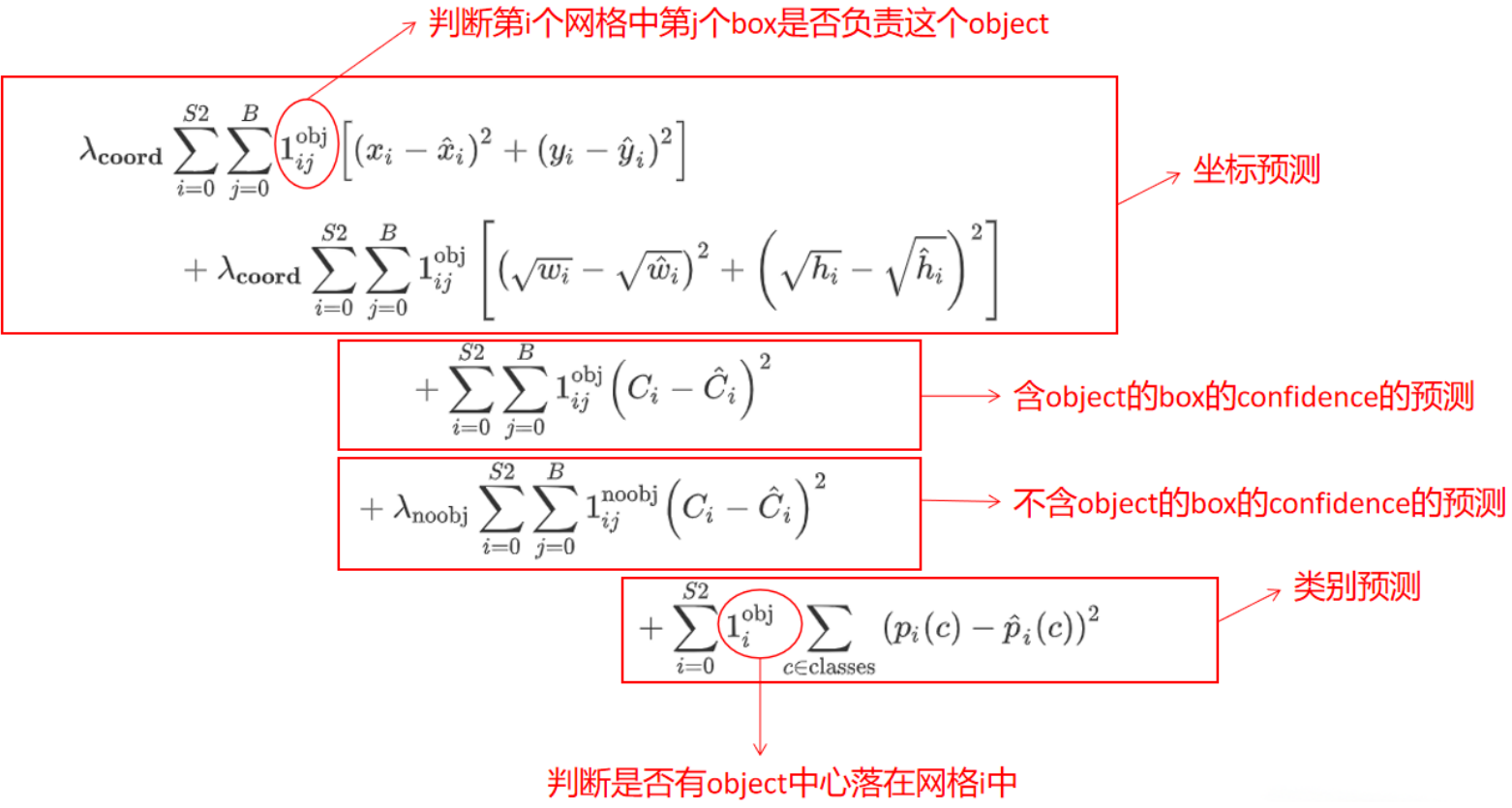
（5）**学习率**。第一个迭代周期，慢慢地将学习率从10^{-3}提高到10^{-2}；然后以10^{-2}的学习率训练75个迭代周期，用10^{-3}的学习率训练30个迭代周期，最后用10^{-4}的学习率训练30个迭代周期。

（6）**避免过拟合策略**。使用dropout和数据增强来避免过拟合。

YOLOv1

训练过程与细节：

（4）损失函数：由**坐标预测**、**是否包含目标**物体置信度、**类别**预测构成；



$$\lambda_{\text{coord}} \sum_{i=0}^{S2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S2} \sum_{j=0}^{B} 1_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S2} 1_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

1_i^{obj}：目标是否出现在网格单元i 中；

1_{ij}^{obj}：网格单元i 中的第j 个边界框预测器是否“负责”该预测；

- 如果目标存在于该网格单元中(条件类别概率)，则损失函数仅惩罚**分类错误**；
- 如果预测器“负责”实际边界框（即该网格单元中具有最高IOU的预测器），则它也惩罚边界框坐标错误。

YOLOv1

优点：

- YOLO检测物体**速度非常快**，其增强版GPU中能跑45fps（frame per second），简化版155fps
- YOLO在训练和测试时都能看到一整张图的信息。因此，能**很好利用上下文信息**，不容易在背景上预测出错误的物体信息。
- YOLO可以学到物体泛化特征

缺点：

- 精度低于其它state-of-the-art的物体检测系统
- **容易产生定位错误**
- 对**小物体**检测效果不好，尤其是密集的小物体，因为一个栅格只能检测2个物体
- 由于**损失函数**的问题，**定位误差**是影响检测效果的主要原因，尤其是大小物体处理上还有待加强

YOLOv2

Ross Girshick吸收fast-RCNN和SSD算法，设计了YOLOv2：

- 在**精度**上利用一系列训练技巧；
- 在**速度**上应用了新的网络模型DarkNet19；
- 在**分类任务**上采用**联合训练方法**，结合wordtree等方法，使YOLOv2的检测种类扩充到了上千种，可以检测超过9000个目标类别，所以也称**YOLO9000**。
- YOLOv2模型可以以不同的尺寸运行，从而在速度和准确性之间提供了一个简单的折衷。

YOLOv2

YOLOv2对YOLOv1采取了很多改进措施，以提高模型mAP，如下图所示：

	YOLO							YOLOv2
batch norm?		✓	✓	✓	✓	✓	✓	✓	✓
hi-res classifier?			✓	✓	✓	✓	✓	✓	✓
convolutional?				✓	✓	✓	✓	✓	✓
anchor boxes?				✓	✓				
new network?					✓	✓	✓	✓	✓
dimension priors?						✓	✓	✓	✓
location prediction?						✓	✓	✓	✓
passthrough?							✓	✓	✓
multi-scale?								✓	✓
hi-res detector?									✓
VOC2007 mAP	63.4	65.8	69.5	69.2	69.6	74.4	75.4	76.8	**78.6**

YOLOv2

(1)Batch Normalization（批量正则化）

- YOLOv2中在每个卷积层后加Batch Normalization(BN)层，去掉dropout。
- BN层可以起到一定的正则化效果，能提升模型收敛速度，防止模型过拟合。
- YOLOv2通过使用BN层使得mAP提高了2%。

YOLOv2

(2)High Resolution Classifier（高分辨率分类器）

◆ **YOLO网络：**在预训练的时候采用的是224*224的输入，然后在detection的时候采用448*448的输入。这会导致从分类模型切换到检测模型的时候，模型还要适应图像分辨率的改变。

◆ **YOLOv2**将预训练分成两步：

- 首先，用224*224的输入从头开始训练网络，大概160个epoch；
- 然后，再输入448*448，训练10个epoch。(这两步均在ImageNet数据集上操作。)
- 最后，再在检测的数据集上fine-tuning，也就是detection的时候用448*448的图像作为输入就可以顺利过渡了。

YOLOv2

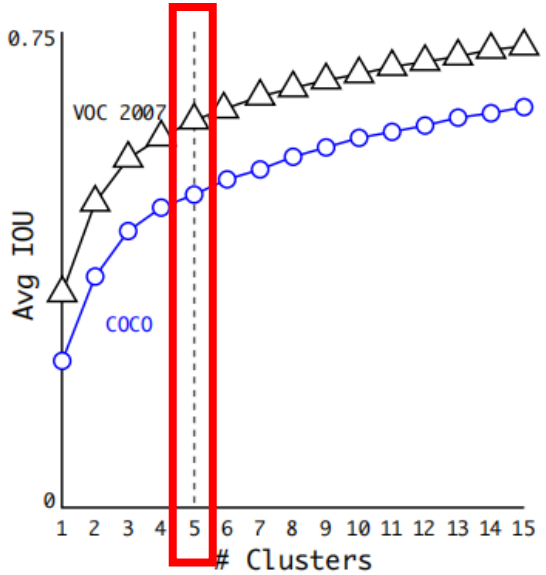
(3)Convolutional With Anchor Boxes（带Anchor Boxes的卷积）

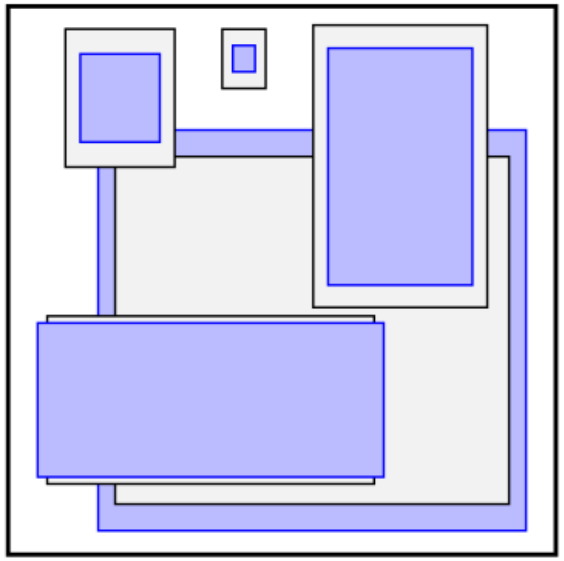
- YOLOv2去掉了YOLOv1中的**全连接层(空间信息丢失多，定位不准)**，使用Anchor Boxes预测边界框，
- 为了得到更高分辨率的特征图，YOLOv2还去掉了一个池化层。
- YOLOv2通过缩减网络，使用416*416的输入，模型下采样的总步长为32，最后得到13*13的特征图，然后对13*13的特征图的每个cell预测5个anchor boxes，对每个anchor box预测边界框的位置信息、置信度和一套分类概率值。
- 使用anchor boxes之后，YOLOv2可以预测13*13*5=845个边界框，模型的召回率由原来的81%提升到88%(**提升7%**)，mAP由原来的69.5%降低到69.2%(**下降0.3%**)。

YOLOv2

（4）Dimension Clusters（维度聚类）

- 在Faster R-CNN和SSD中，先验框都是手动设定的，带有一定的主观性。
- YOLOv2采用k-means聚类算法对训练集中的边界框做了聚类分析，选用boxes之间的IOU值作为聚类指标。





5个聚类中心→5个先验框

扁长框少，瘦高框多，符合行人检测

YOLOv2

（5）New Network（新的网络）

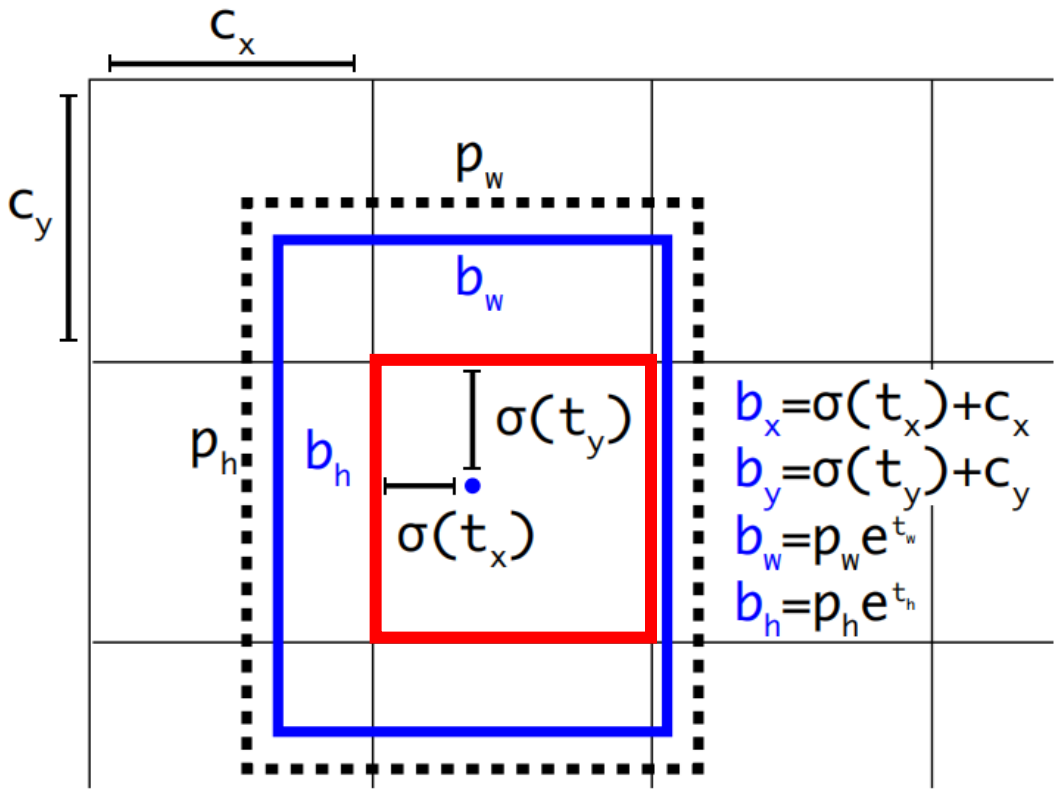
YOLOv2采用Darknet-19，如右图所示：

- 包括19个卷积层和5个max pooling层，主要采用3*3卷积和1*1卷积。
- **1*1卷积**可以压缩特征图通道数以降低模型计算量和参数，
- 每个卷积层后使用**BN层**以加快模型收敛同时防止过拟合。
- 最终采用global avg pool 做预测。

Type	Filters	Size/Stride	Output
Convolutional	32	3 × 3	224 × 224
Maxpool		2 × 2/2	112 × 112
Convolutional	64	3 × 3	112 × 112
Maxpool		2 × 2/2	56 × 56
Convolutional	128	3 × 3	56 × 56
Convolutional	64	1 × 1	56 × 56
Convolutional	128	3 × 3	56 × 56
Maxpool		2 × 2/2	28 × 28
Convolutional	256	3 × 3	28 × 28
Convolutional	128	1 × 1	28 × 28
Convolutional	256	3 × 3	28 × 28
Maxpool		2 × 2/2	14 × 14
Convolutional	512	3 × 3	14 × 14
Convolutional	256	1 × 1	14 × 14
Convolutional	512	3 × 3	14 × 14
Convolutional	256	1 × 1	14 × 14
Convolutional	512	3 × 3	14 × 14
Maxpool		2 × 2/2	7 × 7
Convolutional	1024	3 × 3	7 × 7
Convolutional	512	1 × 1	7 × 7
Convolutional	1024	3 × 3	7 × 7
Convolutional	512	1 × 1	7 × 7
Convolutional	1024	3 × 3	7 × 7
Convolutional	1000	1 × 1	7 × 7
Avgpool		Global	1000
Softmax			

（6）直接定位预测（Direct location Prediction）

- 根据所在网格单元的位置来预测坐标。
- 设一个网格相对于图片**左上角**的**偏移量**是c_x，c_y。
- **先验框**的**宽度和高度**分别是p_w 和p_h，
- 则预测的边界框相对于特征图的中心坐标(b_x, b_y) 和宽高 (b_w, b_h)的计算公式如右图所示。



t_w, t_h是尺度缩放，分别经过sigmoid，输出0-1之间的偏移量；σ为sigmoid函数；t_x, t_y 是预测的坐标偏移值（中心点坐标），与c_x, c_y 相加后得到bounding box中心点的位置。

YOLOv2

（7）细粒度特征（Fine-Grained Features）

YOLOv2借鉴SSD使用多尺度的特征图做检测，提出pass through层将高分辨率的特征图与低分辨率的特征图联系在一起，从而实现多尺度检测。

（8）多尺度训练（Multi-Scale Training）

- YOLOv2采用多尺度输入的方式训练，在训练过程中每隔10个batches，重新随机选择输入图片的尺寸。
- 采用Multi-Scale Training, 可以适应不同大小的图片输入，当采用低分辨率的图片输入时，mAP值略有下降，但速度更快，当采用高分辨率的图片输入时，能得到较高mAP值，但速度有所下降。

YOLOv2

实验结果：

Detection Frameworks	Train	mAP	FPS
Fast R-CNN [5]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[15]	2007+2012	73.2	7
Faster R-CNN ResNet[6]	2007+2012	76.4	5
YOLO [14]	2007+2012	63.4	45
SSD300 [11]	2007+2012	74.3	46
SSD500 [11]	2007+2012	76.8	19
YOLOv2 288 × 288	2007+2012	69.0	91
YOLOv2 352 × 352	2007+2012	73.7	81
YOLOv2 416 × 416	2007+2012	76.8	67
YOLOv2 480 × 480	2007+2012	77.8	59
YOLOv2 544 × 544	2007+2012	**78.6**	40

YOLOv2

训练过程:

(1)先在ImageNet分类数据集上训练Darknet-19,此时模型输入为224*224，共训练160轮

(2)将网络输入调整为448*448，继续在ImageNet分类数据集上训练细调模型，共10轮，此时分类模型top-1准确率为76.5%，而top-5准确度为93.3%

(3)修改Darknet-19分类模型为检测模型，并在检测数据集上继续细调网络

YOLOv2

优点：

- YOLOv2使用了一个**新的分类器**作为特征提取部分。较多使用了3*3卷积核；每次池化后把通道数翻倍；网络使用了**全局平均池化**，把1*1卷积核置于3*3卷积核之间，用来压缩特征；也用了batch normalization稳定模型训练
- 最终得出的基础模型就是Darknet-19，包含19个卷积层，5个最大池化层，运算次数55.8亿次，top-1图片分类准确率72.9%，top-5准确率91.2%
- YOLOv2比VGG16更快，精度略低于VGG16

缺点：

- YOLOv2检测准确率不够，比SSD稍差
- 不擅长检测小物体
- 对近距离物体准确率较低

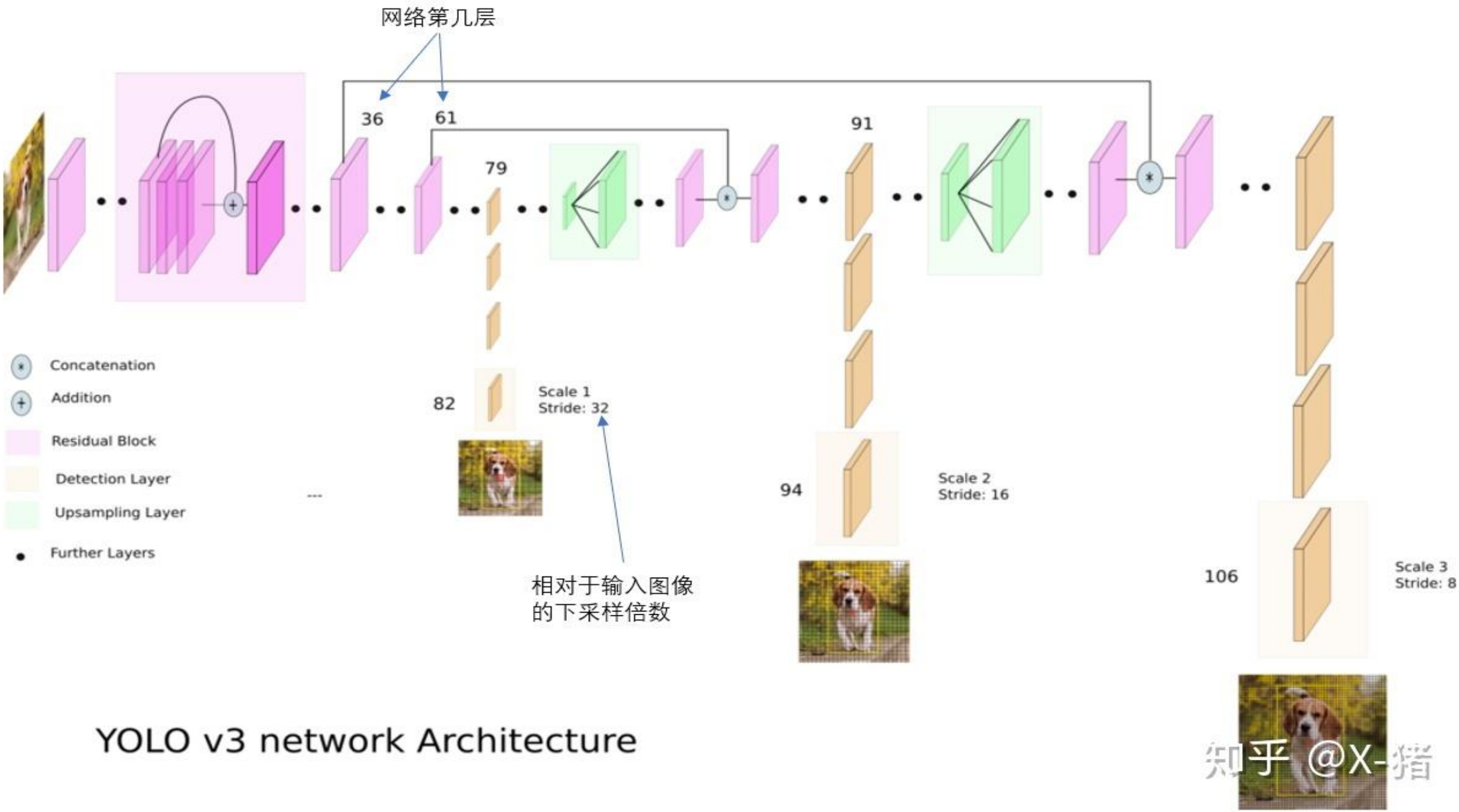
YOLOv3

YOLOv3总结了自己在YOLOv2的基础上做的一些尝试性改进，两个主要的亮点：

- 一个是使用**残差模型**，进一步加深了网络结构；
- 另一个是使用FPN架构实现**多尺度检测**。

改进：

- 新网络结构：DarkNet-53；
- 用逻辑回归替代softmax作为分类器；
- 融合FPN（特征金字塔网络），实现多尺度检测。



YOLO v3 network Architecture

YOLOv3

多尺度预测：

YOLOv3在基本特征提取器上添加几个卷积层，其中最后一个卷积层预测了一个三维张量——**边界框**，**目标**和**类别预测**。

- 在COCO实验中，为每个尺度预测3个框，所以对于4个边界框偏移量，1个目标预测和80个类别预测，张量的大小为N×N×[3 *（4 + 1 + 80）]。
- 接下来，从前面的2个层中取得特征图，并将其上**采样2倍**。然后，再添加几个卷积层来处理这个组合的特征图，并最终**预测**出一个类似的张量，其尺寸是之前的两倍。
- 最后，再次使用相同的设计来预测最终尺寸的边界框。

因此，第三个尺寸的预测将既能从所有先前的计算，又能从网络前面层的细粒度特征中获益。

YOLOv3

网络结构：

YOLOv3在之前Darknet-19的基础上引入了**残差块**，并进一步**加深了网络**，改进后的网络有53个卷积层，取名为Darknet-53，

网络结构如右图所示（以256*256的输入为例）

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Table 1. **Darknet-53.**

YOLOv3

效果：兼顾速度与准确率。

在COCO数据集上，mAP指标与SSD模型相当，但速度提高了3倍；mAP指标比RetinaNet模型差些，但速度要高3.8倍。

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [5]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [8]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [6]	Inception-ResNet-v2 [21]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [20]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [15]	DarkNet-19 [15]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [11, 3]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [3]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [9]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [9]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

04

识别算法实例分析

Faster-Rcnn

Faster-RCNN实践地址：

https://github.com/rbgirshick/py-faster-rcnn

作业15

Assignment15

- 查阅文献，完成本章内容的《计算机视觉课程报告》

THANKS