
YOLO

Joseph Redmon

— You Only Look Once

统一，实时的目标检测框架

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	155
YOLO	2007+2012	63.4	45

Less Than Real-Time

Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

单阶段经典模型

无需提取候选区域
无复杂上下游处理工作

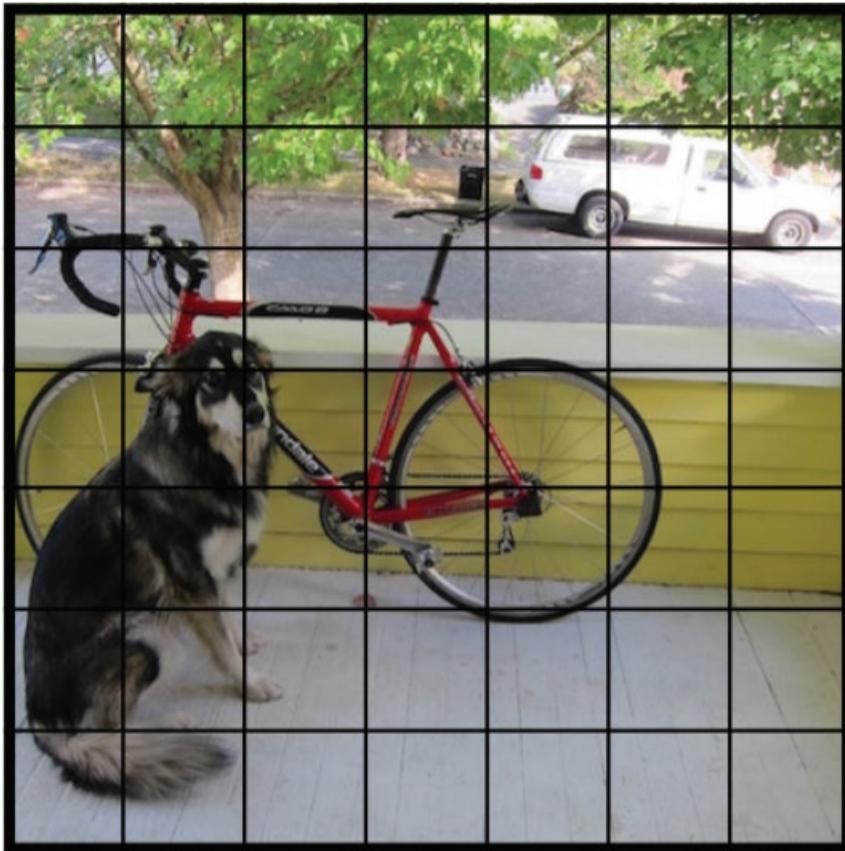
- 次前向推断得到bbox定位及分类结果
端到端训练优化



1. Resize image. 缩放图片
2. Run convolutional network. 卷积网络
3. Non-max suppression. 后处理
Conf过滤 → 非极大值抑制 NMS



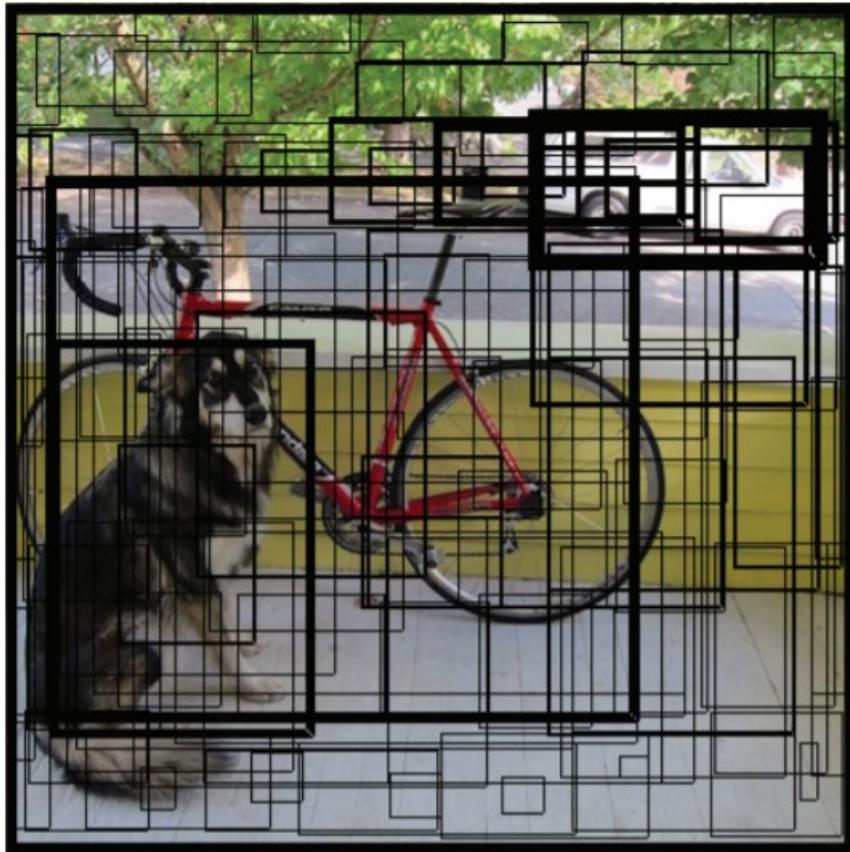
预测阶段



$S \times S$ 个grid cell

$$7 \times 7 = 49$$

预测阶段



B个bounding box

2

(x,y,h,w,c)

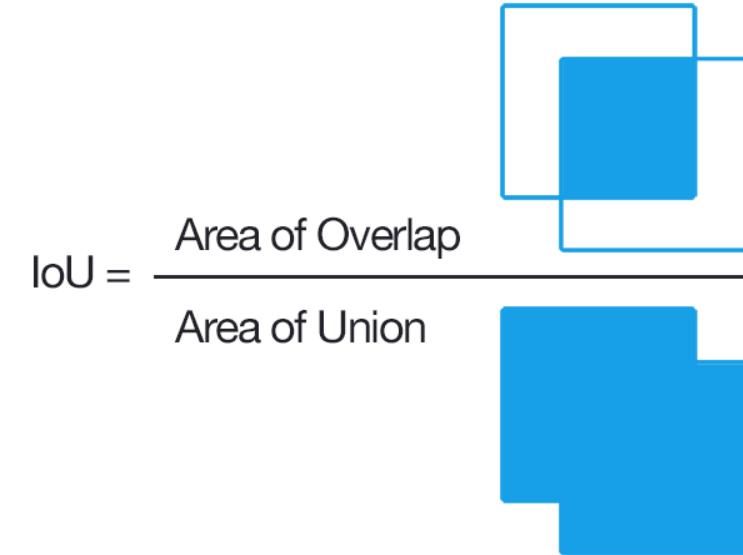
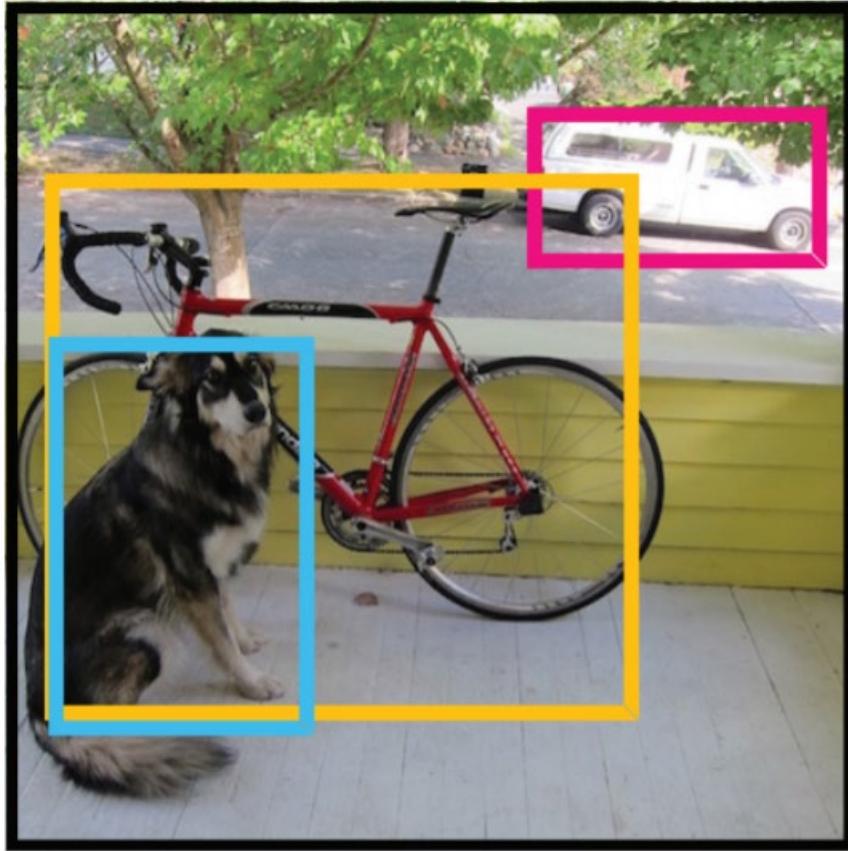
预测阶段



一组条件类别概率

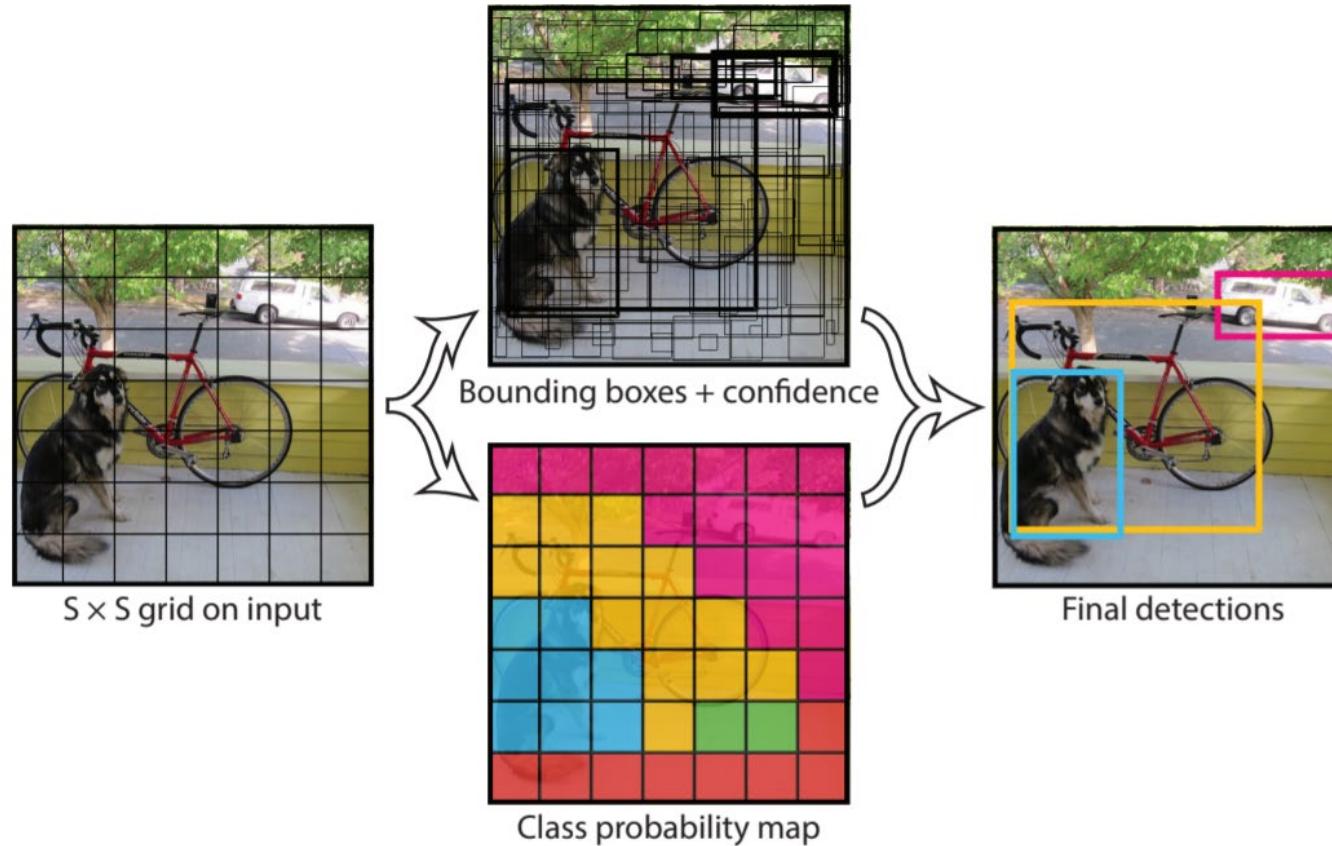
每个cell至少预测出一个类别

预测阶段



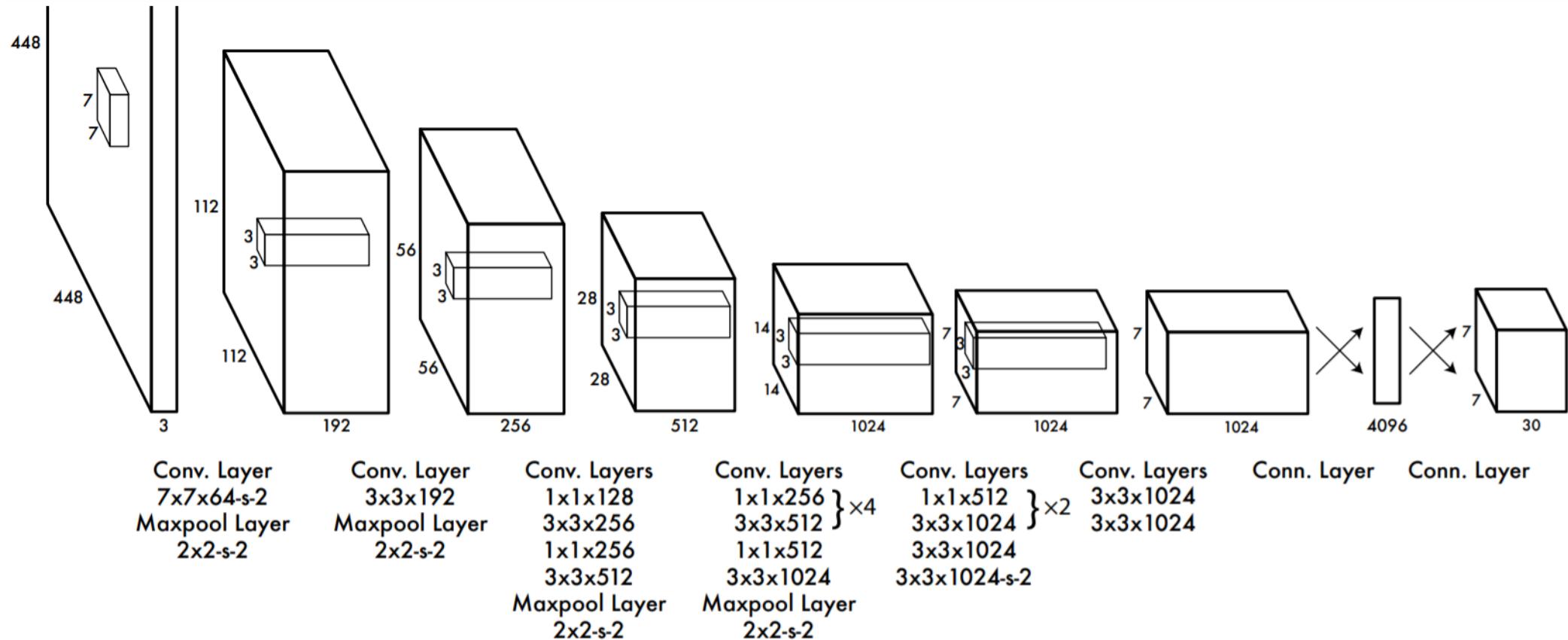
NMS 非极大值抑制

预测阶段



$$7 \times 7 \times (2 \times 5 + 20) = 7 \times 7 \times 30 \text{ tensor} = 1470 \text{ outputs}$$

预测阶段



24层卷积层提取图像特征，2层全连接层回归得到 $7 \times 7 \times 30$ 的tensor

训练阶段

目标损失函数——将目标检测问题当作回归问题

负责检测物体的bbox
中心点定位误差

负责检测物体的bbox
宽高定位误差
求根号能使小框对误差更敏感

负责检测物体的bbox
Confidence 误差

不负责检测物体的bbox
Confidence 误差

负责检测物体的grid cell
分类误差

$$\begin{aligned}
 & \lambda_{\text{coord}}^5 \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
 & + \lambda_{\text{coord}}^5 \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right] \\
 & + \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
 & + \lambda_{\text{noobj}}^{0.5} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
 & + \sum_{i=0}^{S^2} \mathbb{1}_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \quad (3)
 \end{aligned}$$

坐标回归误差

预测值 标签值

计算这个bbox与 ground truth 的 IOU

$C = \Pr(\text{object}) \times \text{IoU}^{\text{pred}}$

从模型正向推断结果为 $S \times S \times (B + S + C)$ 维向量中找到这个bbox的 Confidence Score

性能表现

优点：

1. YOLO检测速度非常快。标准版本的YOLO可以每秒处理45张图像；YOLO的极速版本每秒可以处理150帧图像。这就意味着YOLO可以以小于25毫秒延迟，实时地处理视频。对于欠实时系统，在准确率保证的情况下，YOLO速度快于其他方法。
2. YOLO实时检测的平均精度是其他实时监测系统的两倍。
3. 迁移能力强，能运用到其他的新的领域（比如艺术品目标检测）。

局限：

1. YOLO对相互靠近的物体，以及很小的群体检测效果不好，这是因为一个网格只预测了2个框，并且都只属于同一类。
 2. 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，尤其是大小物体的处理上，还有待加强。（因为对于小的bounding boxes，small error影响更大）
 3. YOLO对不常见的角度的目标泛化性能偏弱。
-

后期展望

1. 深入理解YOLO
 2. 训练YOLO模型
 3. 了解嵌入式机器学习、算法在嵌入式领域的应用
 4. 查阅图像处理相关论文及书籍
-

YOLO9000: Better, Faster, Stronger

Joseph Redmon^{*†}, Ali Farhadi^{*†}

University of Washington^{*}, Allen Institute for AI[†]

<http://pjreddie.com/yolo9000/>

YOLOV1算法的缺点

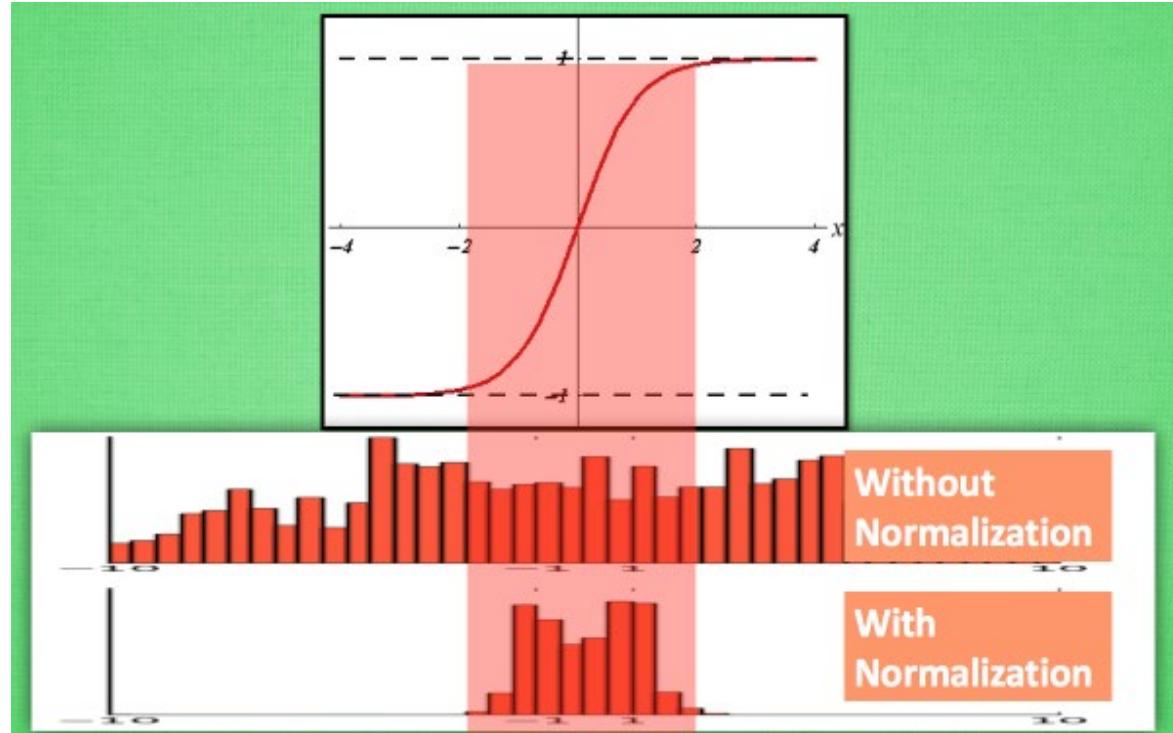
1. 定位性能较差
 2. RECALL比较差
 3. 检测小目标和密集目标性能比较差
 4. 虽然速度快，但是map准确度比较低
-

Better

Batch Normalization
High Resolution Classifier
Anchor
Dimension Cluster
Direct location prediction
Fine-Grained Features.
Multi-Scale Training

Batch Normalization: BN层/批标准化

几乎每一个深度学习的模型都会用到



把神经元的输出减去均值除以标准差，变成以0为均值，标准差为1的分布

如果输出太大或者太小，会陷入激活函数的饱和区，意味着梯度消失，会难以训练。让每一层的值在有效的范围内传递下去，每个区间都有分布的这一种对于神经网络就会更加有价值

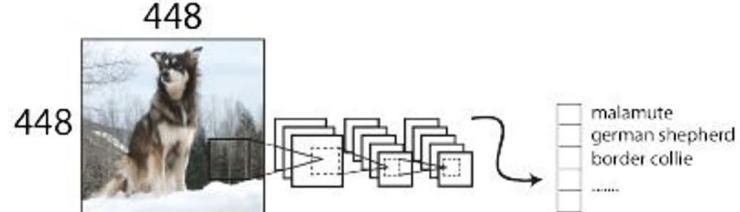
High Resolution Classifier

mAP+4%。

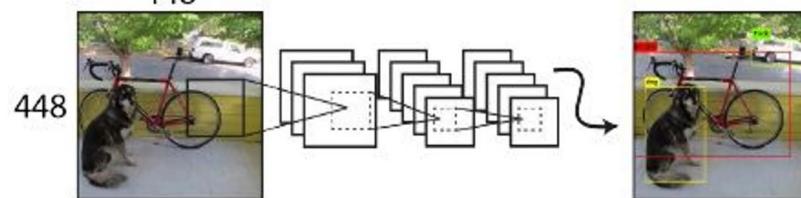
Train on ImageNet



Resize, fine-tune
on ImageNet



Fine-tune on detection



YOLOv1先在ImageNet (224x224) 分类数据集上预训练模型的主体部分，获得较好的分类效果，然后再训练网络的时候将网络的输入从224x224增加为448x448。但是直接切换分辨率，检测模型可能难以快速适应高分辨率。mAP提升了约4%。

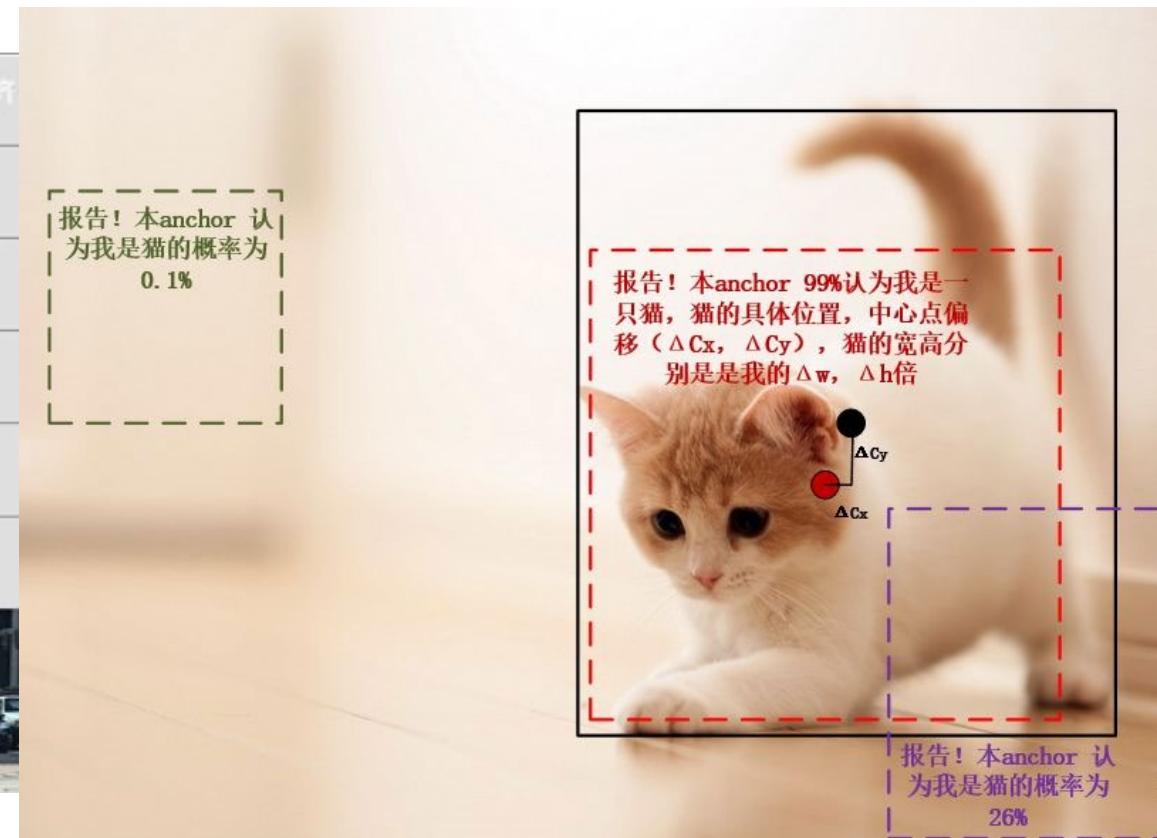
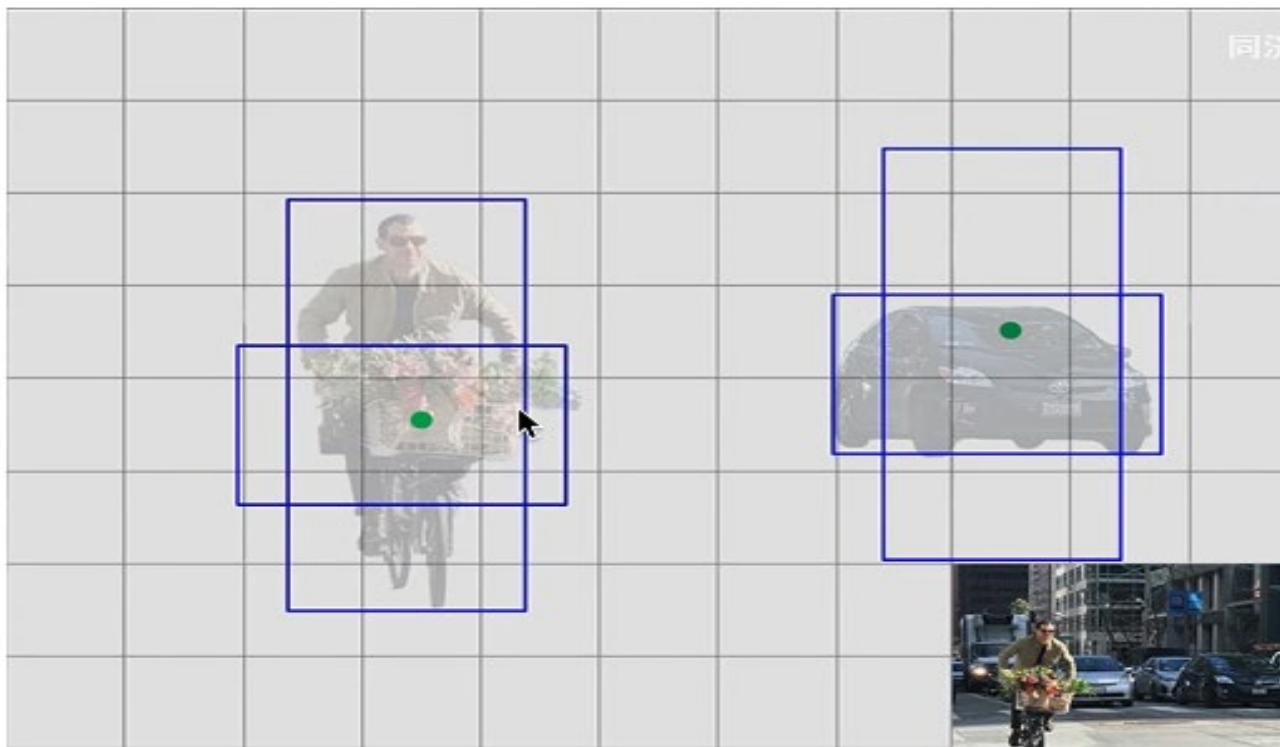
网络不需要从小的分辨率的模型开始去训练，而是直接转型成大分辨率来训练

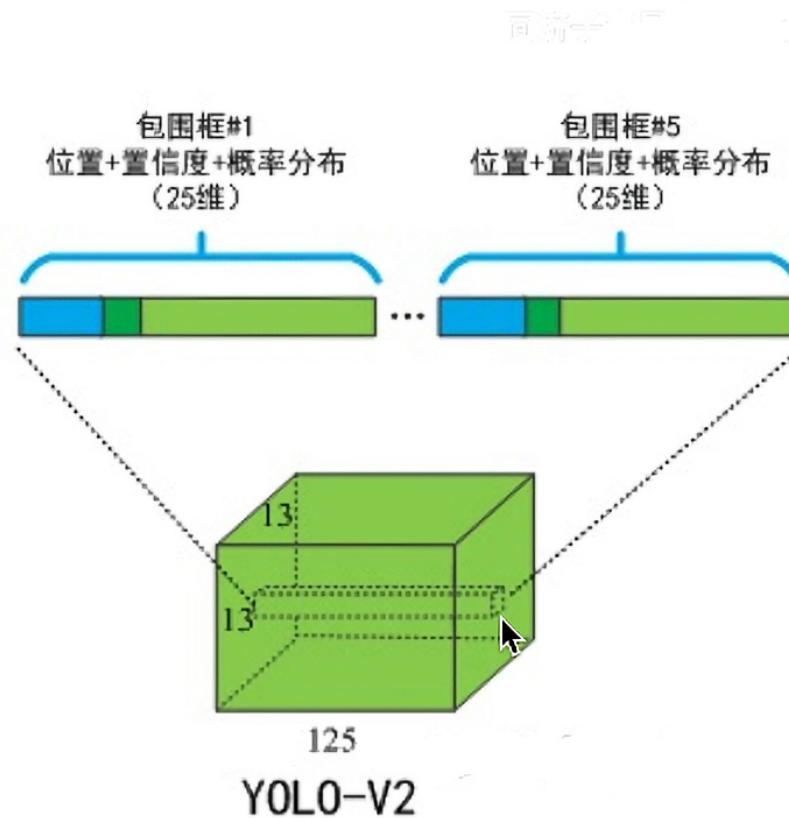
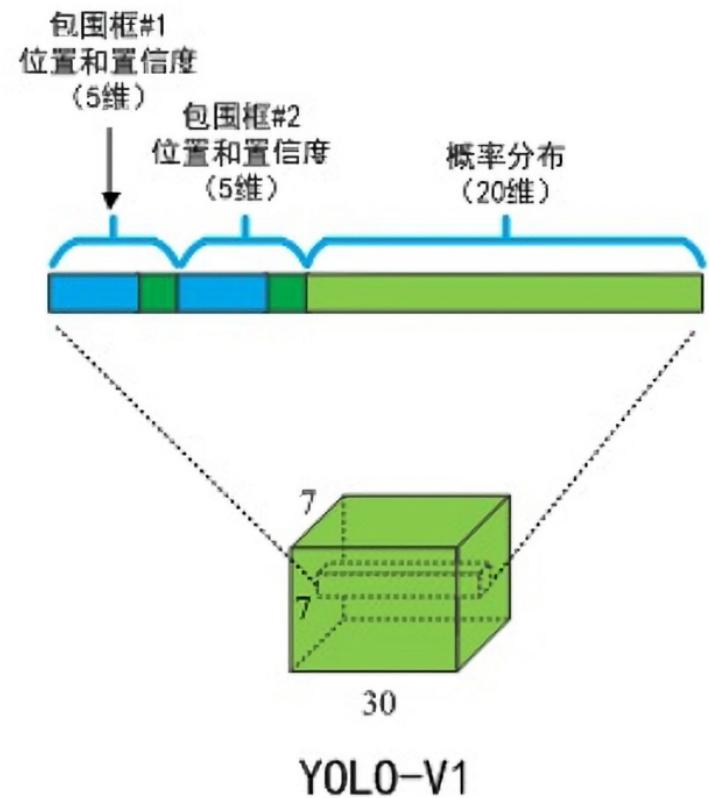
Anchor 锚点/先验参考框（人工设置）

anchor box就是从训练集中真实框（ground truth）中统计或聚类得到的几个不同尺寸的框。避免模型在训练的时候盲目的找，有助于模型快速收敛。

anchor其实就是要预测的对象范围进行约束，并加入了尺寸先验经验，从而实现多尺度学习的目的。

假设每个网格对应k个anchor，也就是模型在训练的时候，它只是会在每一个网格附近找出这k种形状，不会找其他的。预测框只需要预测出相对于anchor的偏移量，13X13个grid cell, 每个cell预设五种大小长宽不同的先验框

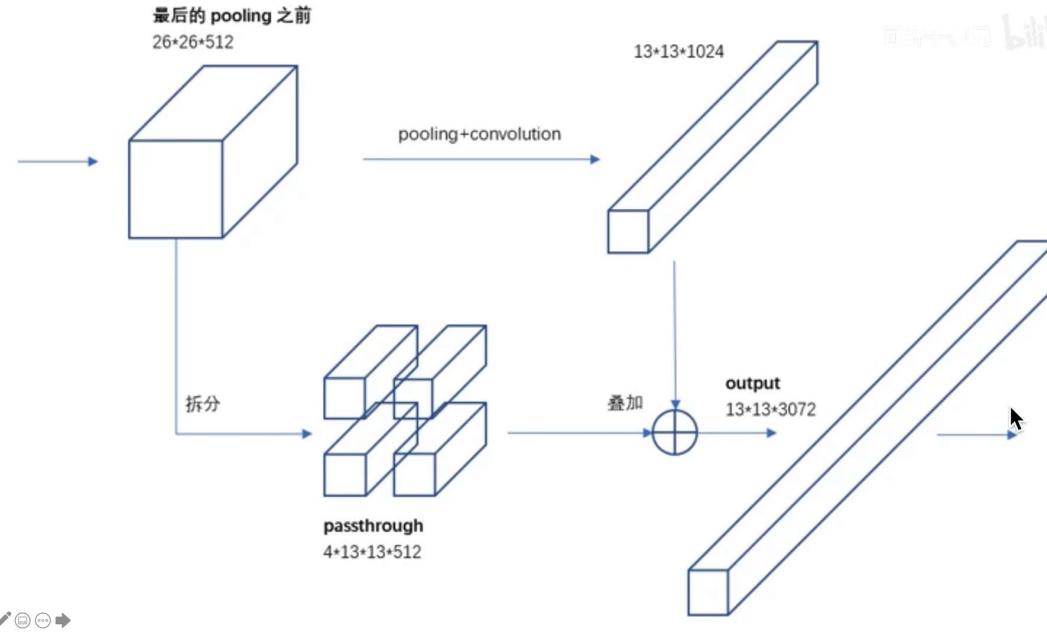




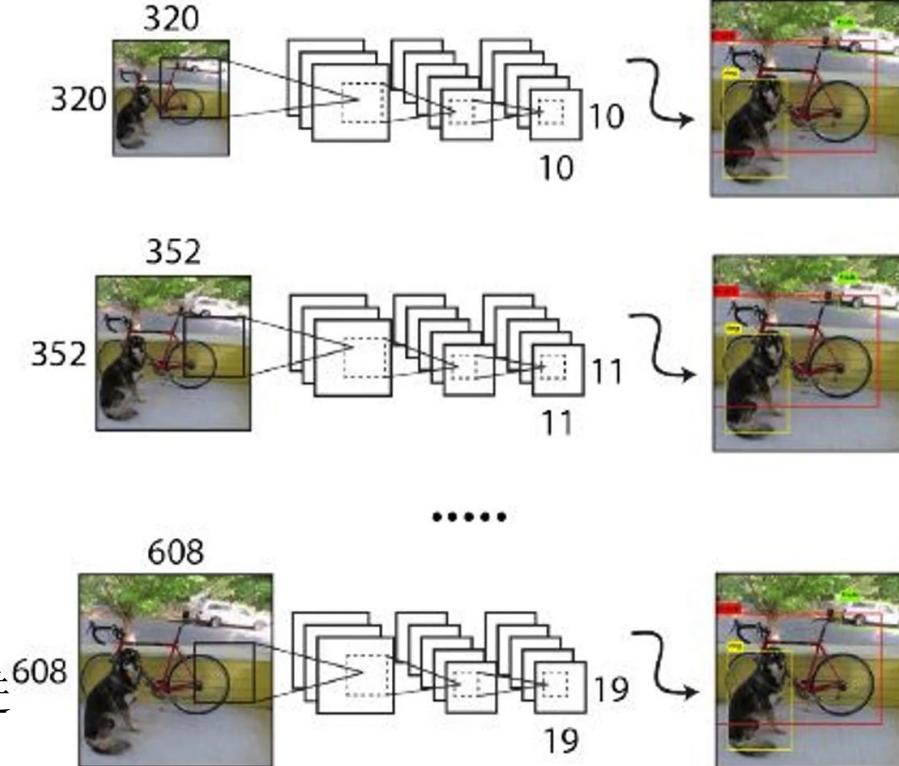
Each grid cell $5 \times (4+1+20) = 125$

Fine-Grained Features 细粒度特征

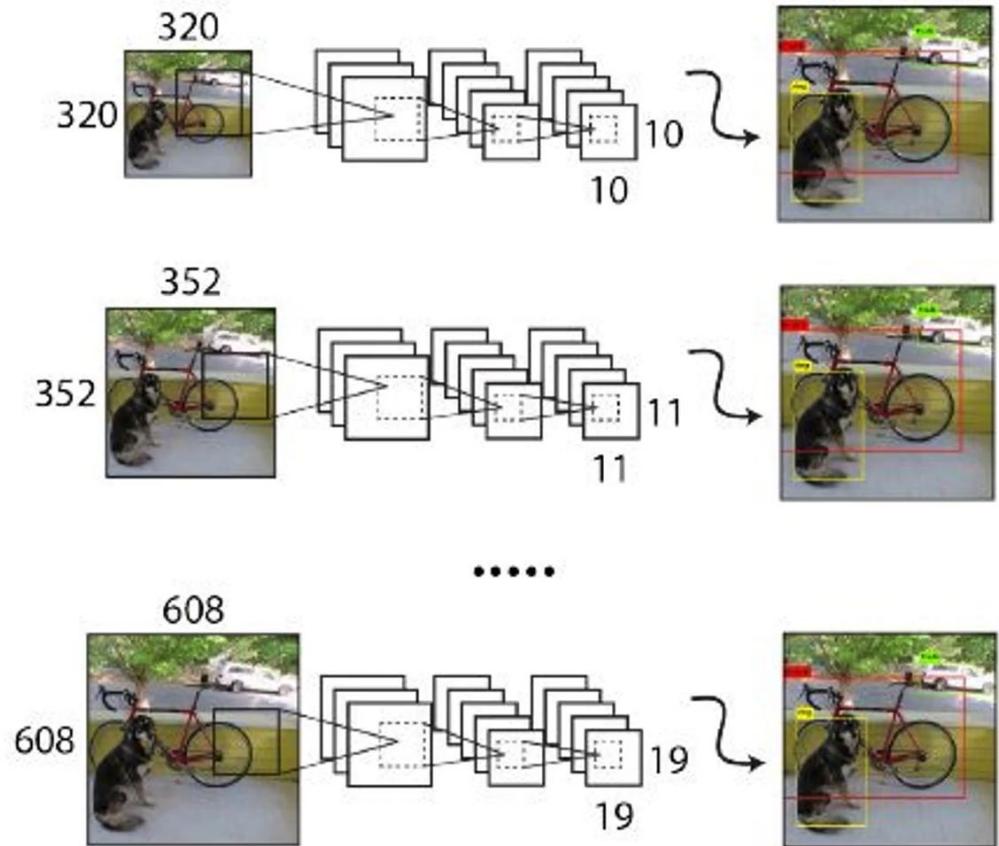
Multi-Scale Training



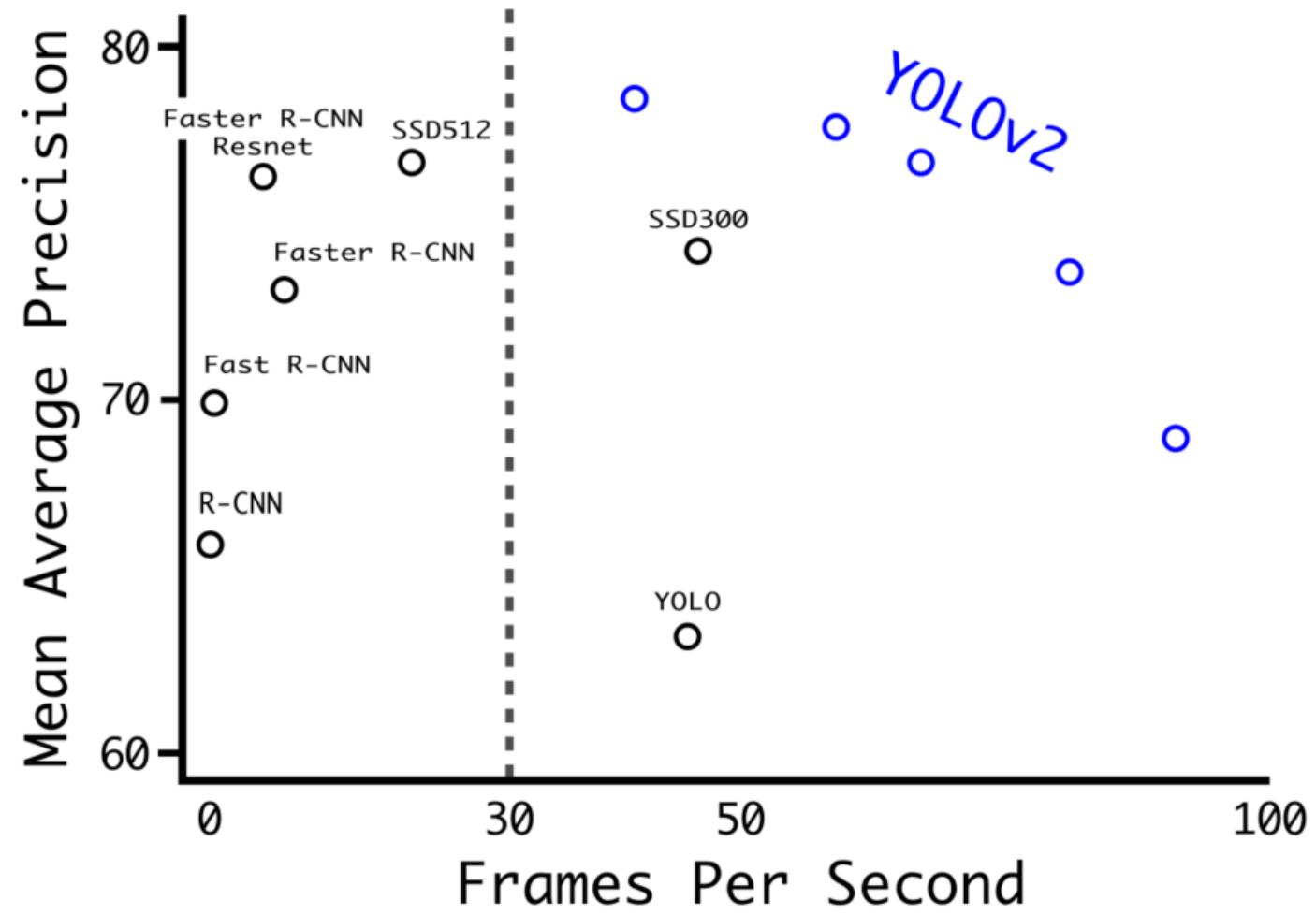
把浅层的网络Feather Map一路拆分成四个部分，另外一路进行下采样卷积池化，后来把之前的四个部分拼接成一个长的向量，与池化后的拼接在一起，就包含了浅层信粒度特征和高层的特征，这样就有利于检测小物体



Multi-Scale Training



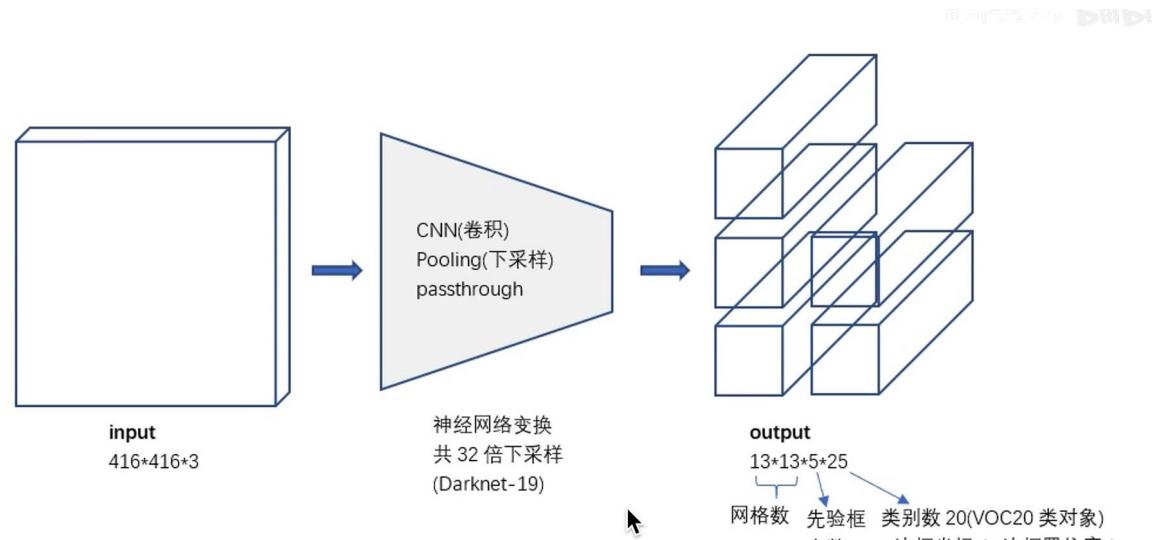
为了让模型更有鲁棒性，作者引入了多尺度训练。就是在训练过程中，每迭代一定的次数，改变模型的输入图片大小。



Faster

Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	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		$2 \times 2/2$	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			

<https://blog.csdn.net/little>

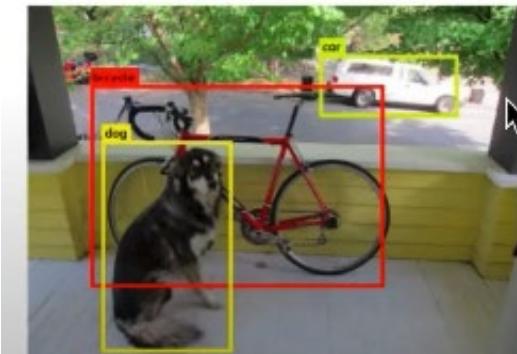


网络包含19个卷积层和5个max pooling层，而在YOLOv1中采用的GooleNet，包含24个卷积层和2个全连接层，因此Darknet-19整体上卷积卷积操作比YOLOv1中用的GoogleNet要少，这是计算量减少的关键。最后用average pooling层代替全连接层进行预测。

Stronger



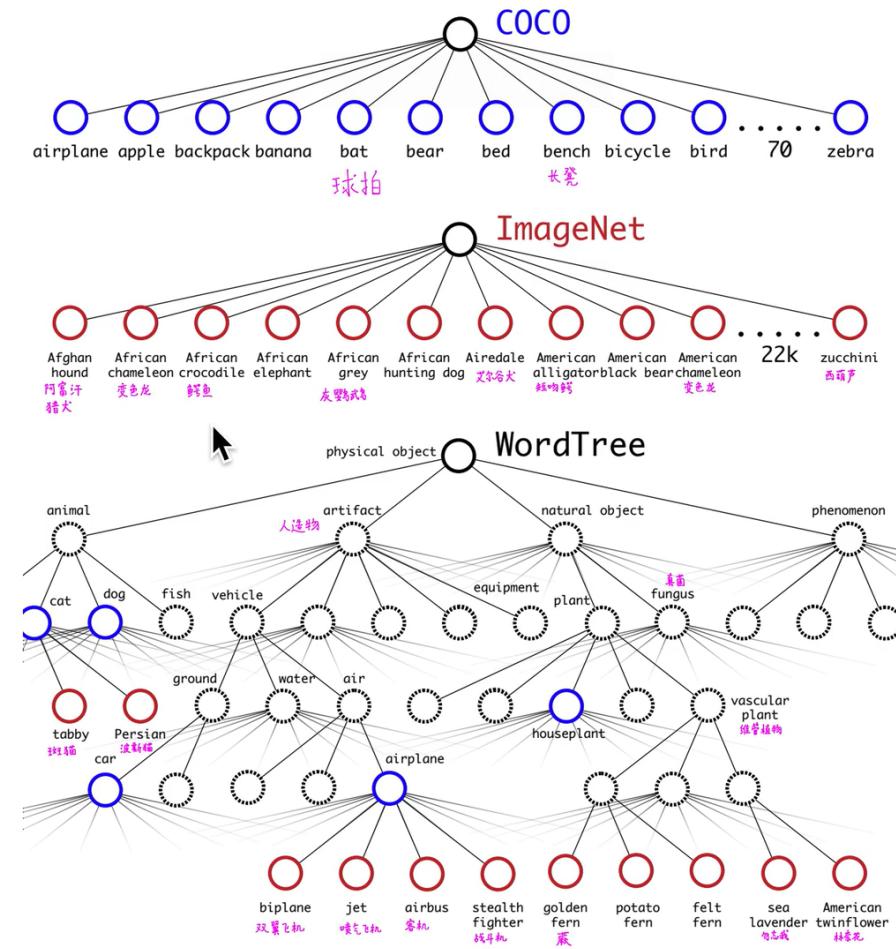
- 100k images
- 80 classes
- Detection labels



带标注的检测数据集量比较少，而带标注的分类数据集量比较大，因此YOLO9000主要通过结合分类和检测数据集使得训练得到的检测模型可以检测约9000类物体。

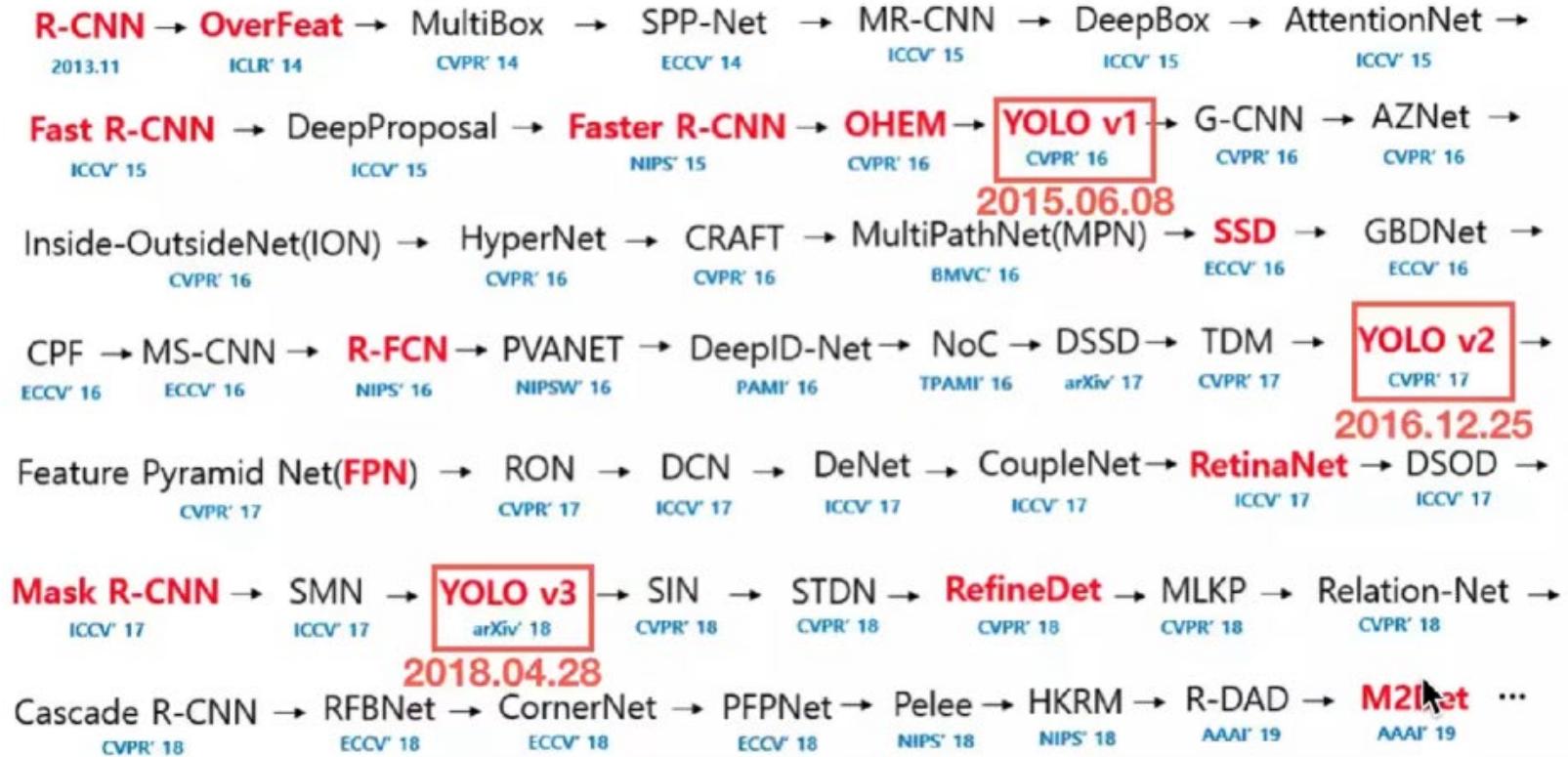


- 14 million images
- 22k classes
- Classification labels

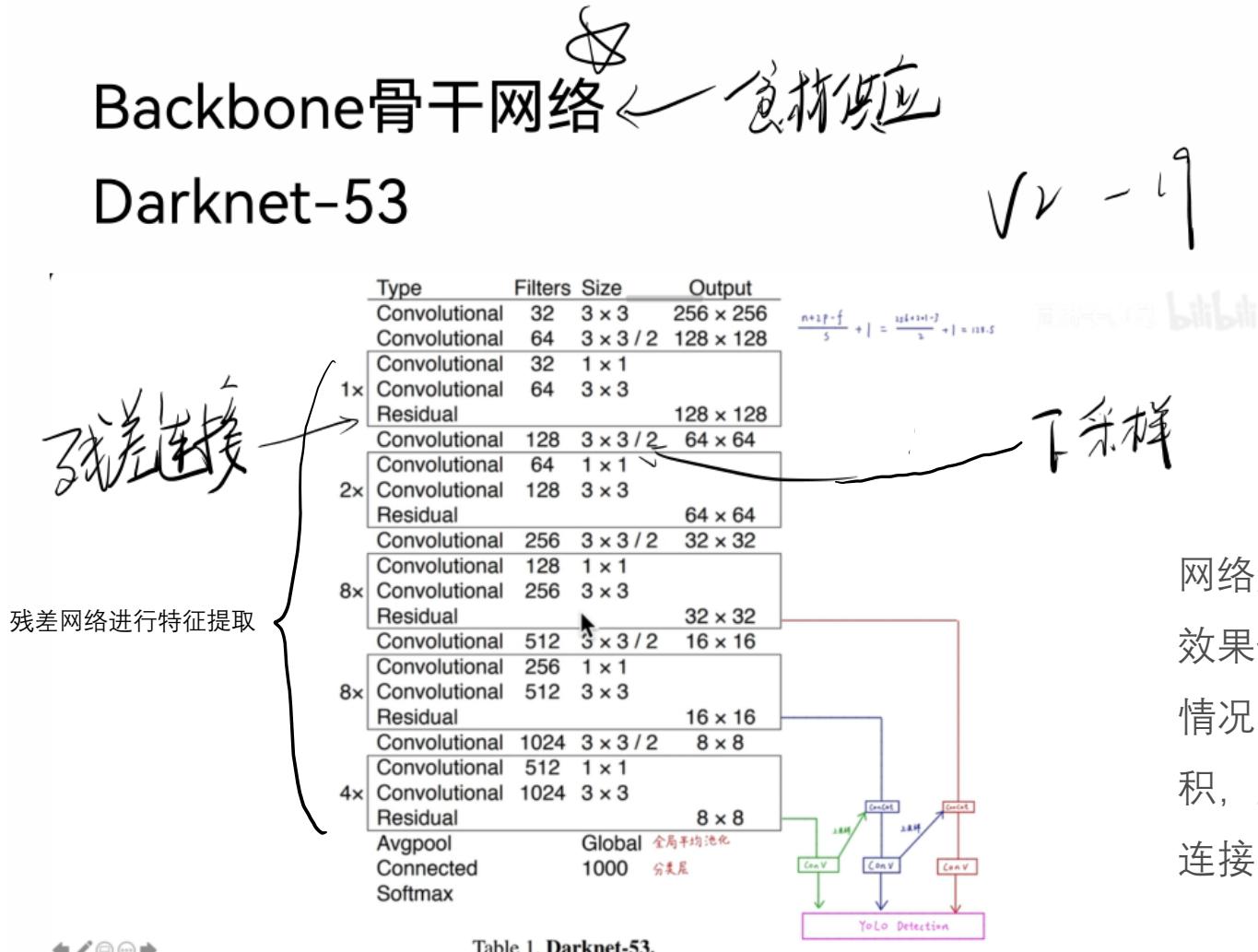


YOLOv3: An Incremental Improvement

Joseph Redmon Ali Farhadi
University of Washington



网络结构



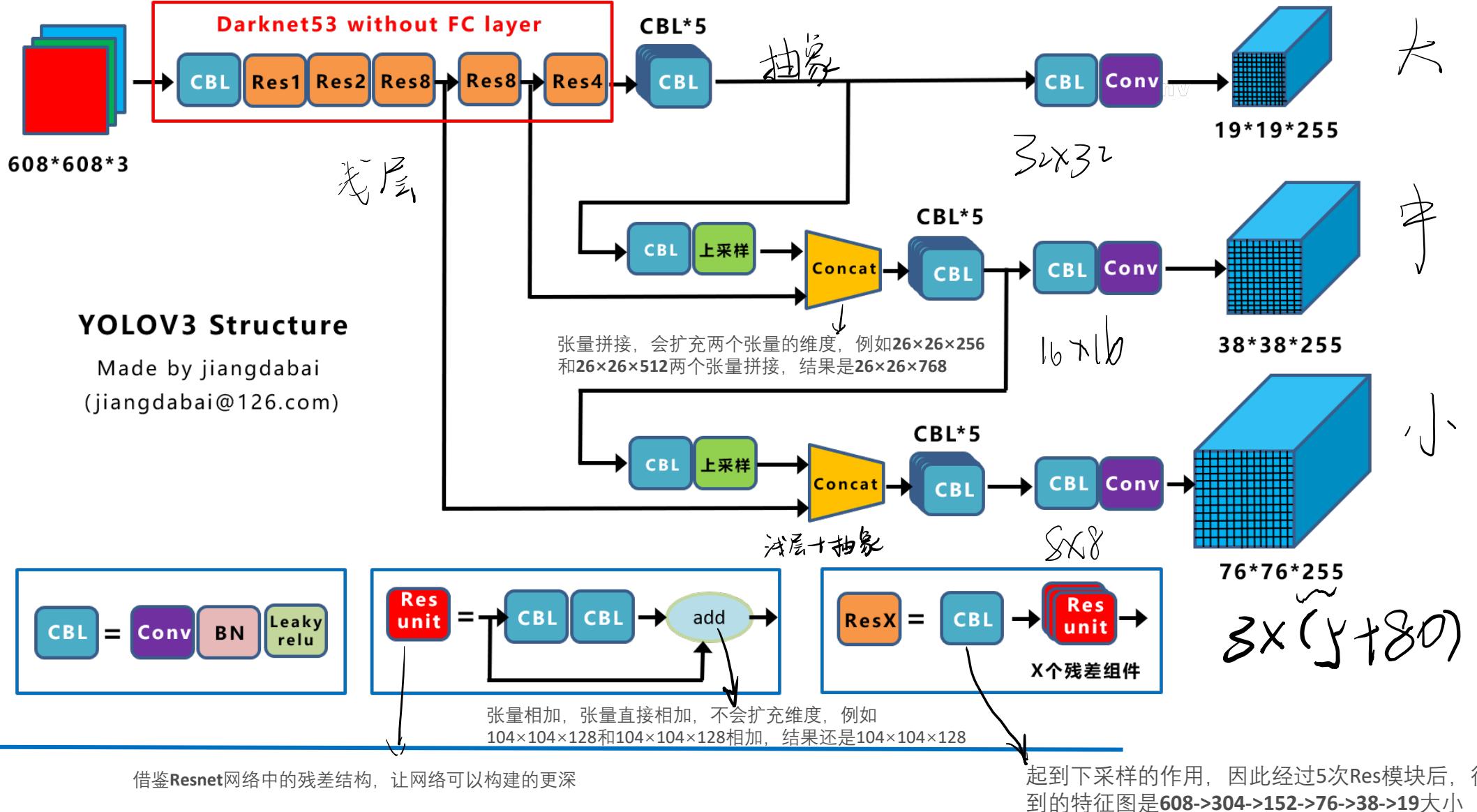
下采样 (subsampled) 的主要目的:

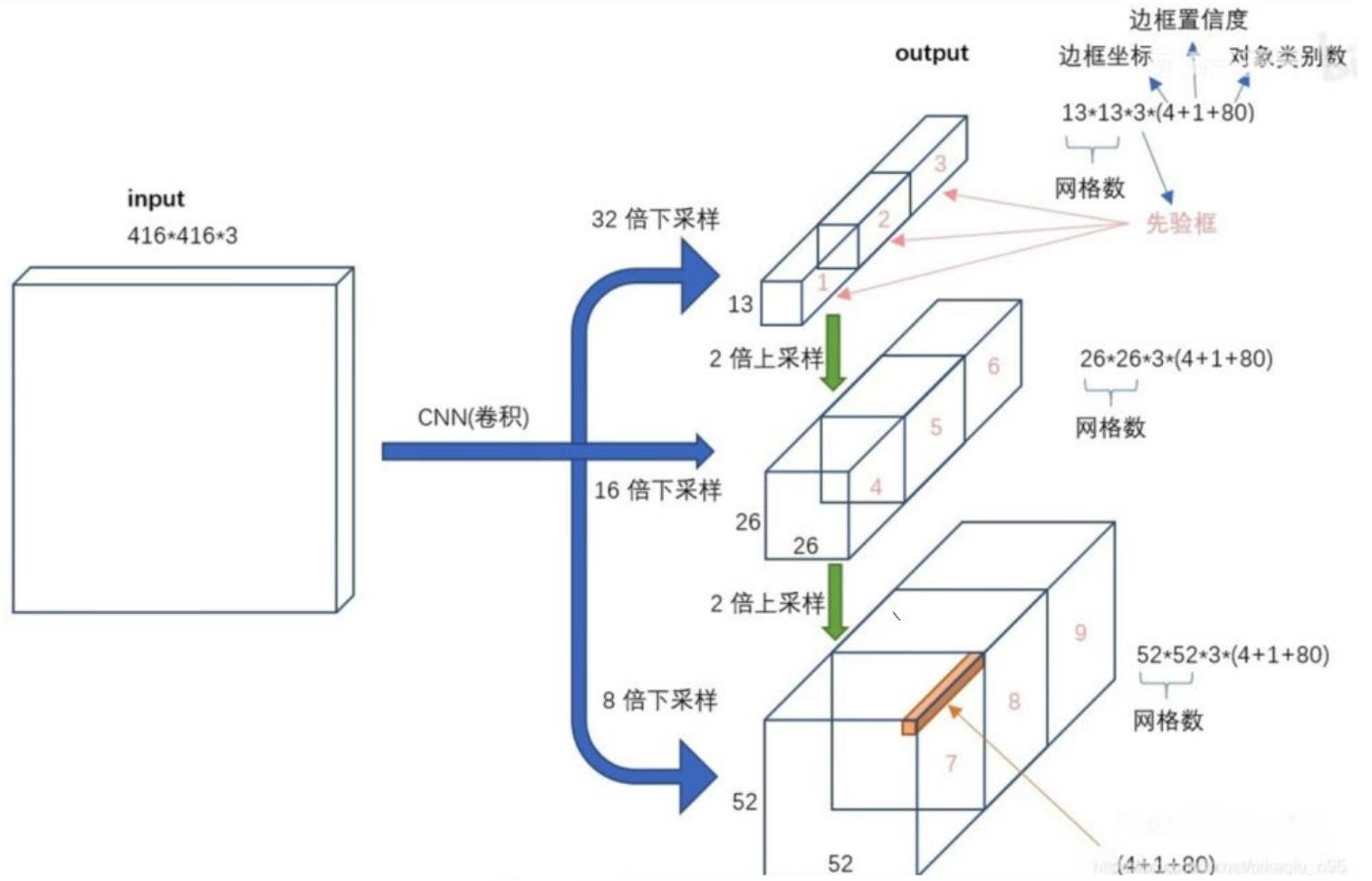
- 1、使得图像符合显示区域的大小;
- 2、生成对应图像的缩略图。

上采样 (upsampling) 的主要目的是放大原图像,从而可以显示在更高分辨率的显示设备上。

网络越深, 梯度消失的现象就越来越明显, 网络的训练效果也不会很好。残差神经网络就是为了在加深网络的情况下又解决梯度消失的问题。残差结构可以不通过卷积, 直接从前面一个特征层映射到后面的特征层 (跳跃连接), 有助于训练, 也有助于特征的提取, 容易优化。

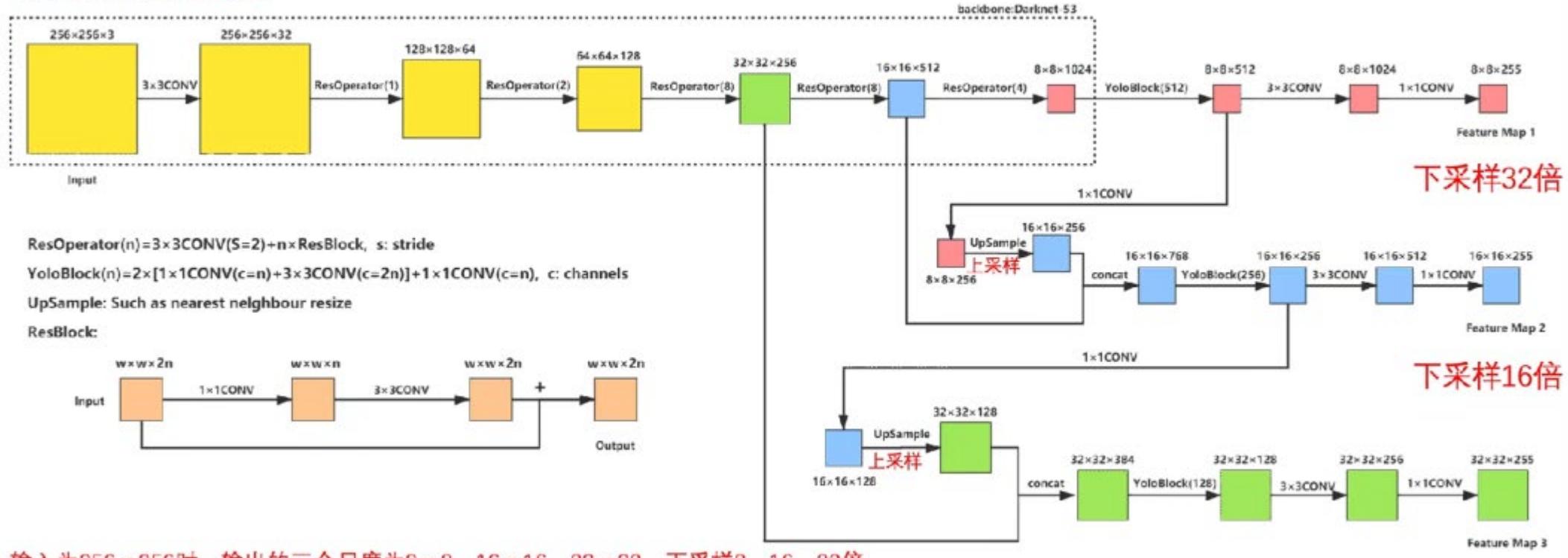
每个ResX中包含 $1+2\times X$ 个卷积层，因此整个主干网络Backbone中一共包含 $1+ (1+2\times 1) + (1+2\times 2) + (1+2\times 8) + (1+2\times 8) + (1+2\times 4) = 52$ ，再加上一个FC全连接层，即可以组成一个Darknet53分类网络。



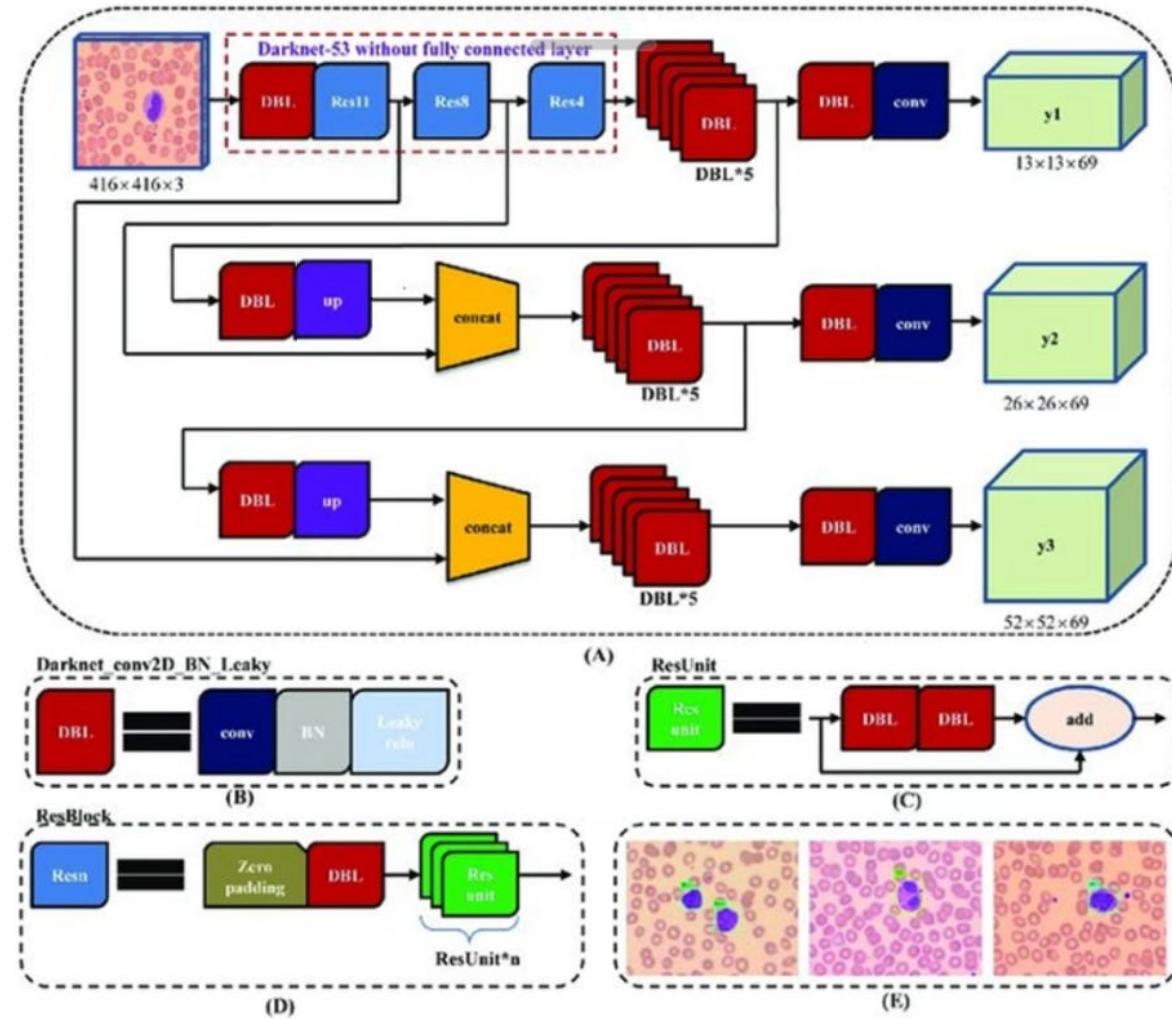


YOLOV3深度学习目标检测算法 神经网络结构

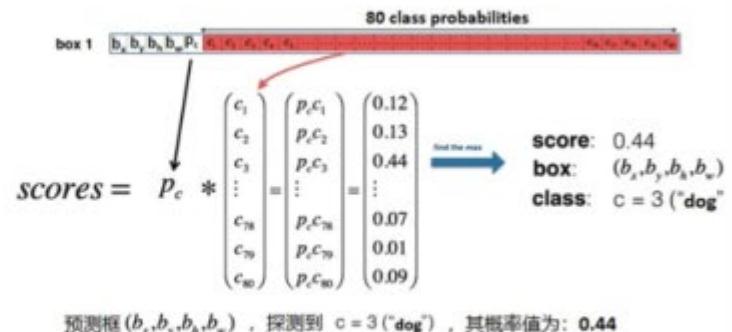
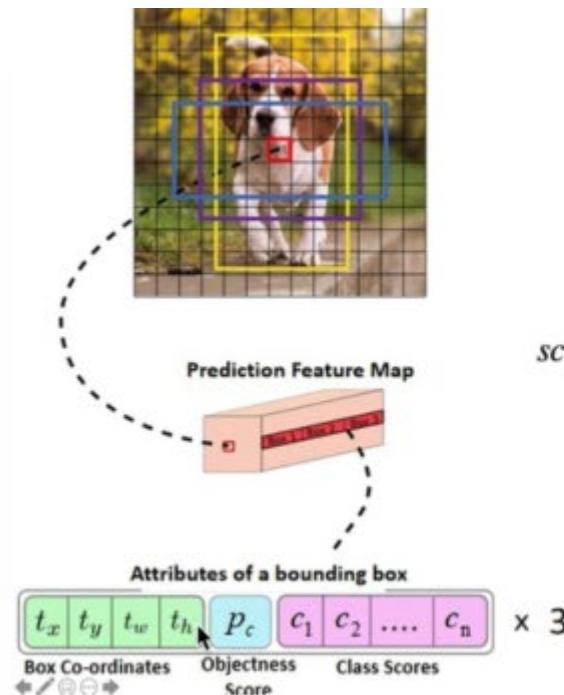
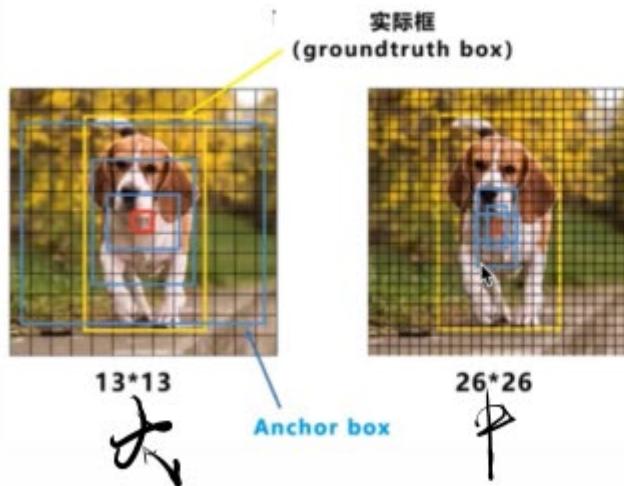
以输入尺寸 256×256 为例



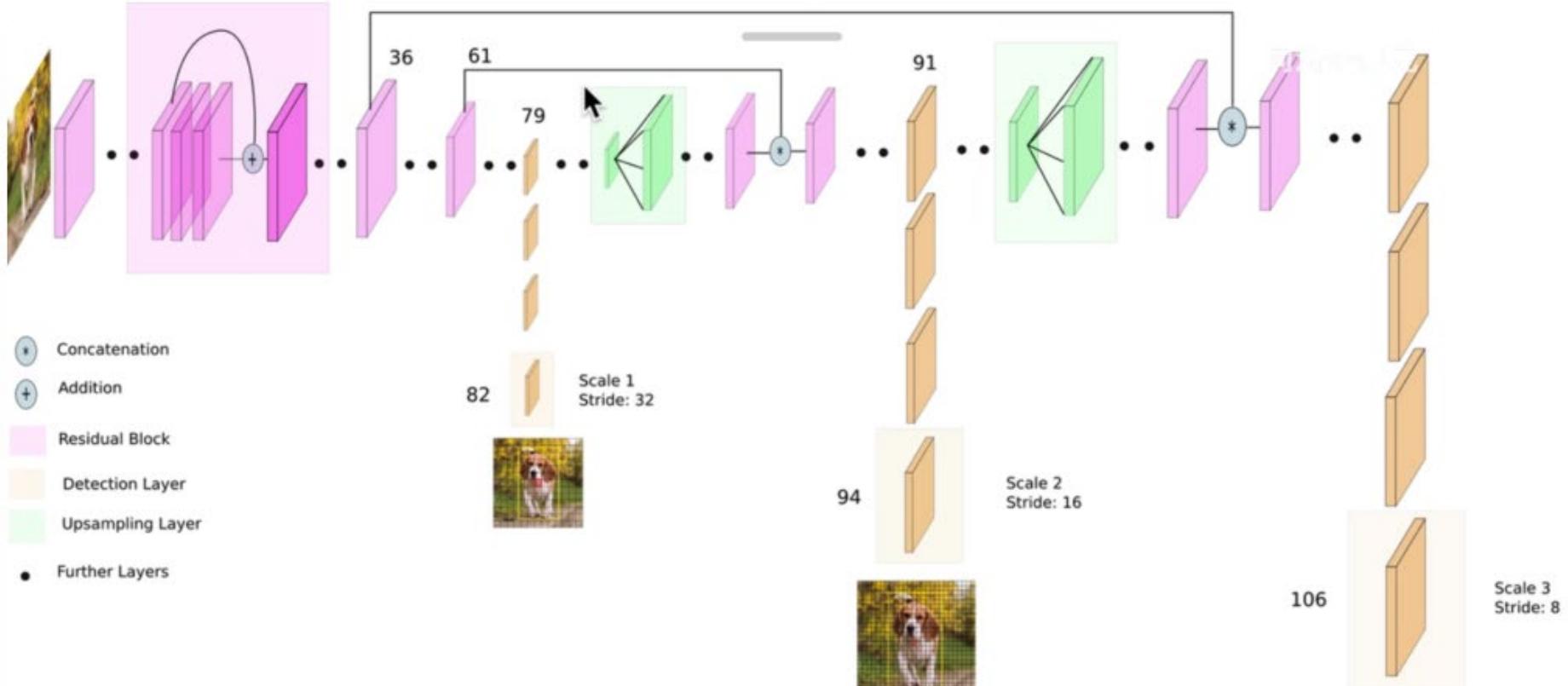
输入为 256×256 时，输出的三个尺度为 8×8 、 16×16 、 32×32 ，下采样8、16、32倍
输入为 416×416 时，输出的三个尺度为 13×13 、 26×26 、 52×52 ，下采样8、16、32倍



多尺度预测



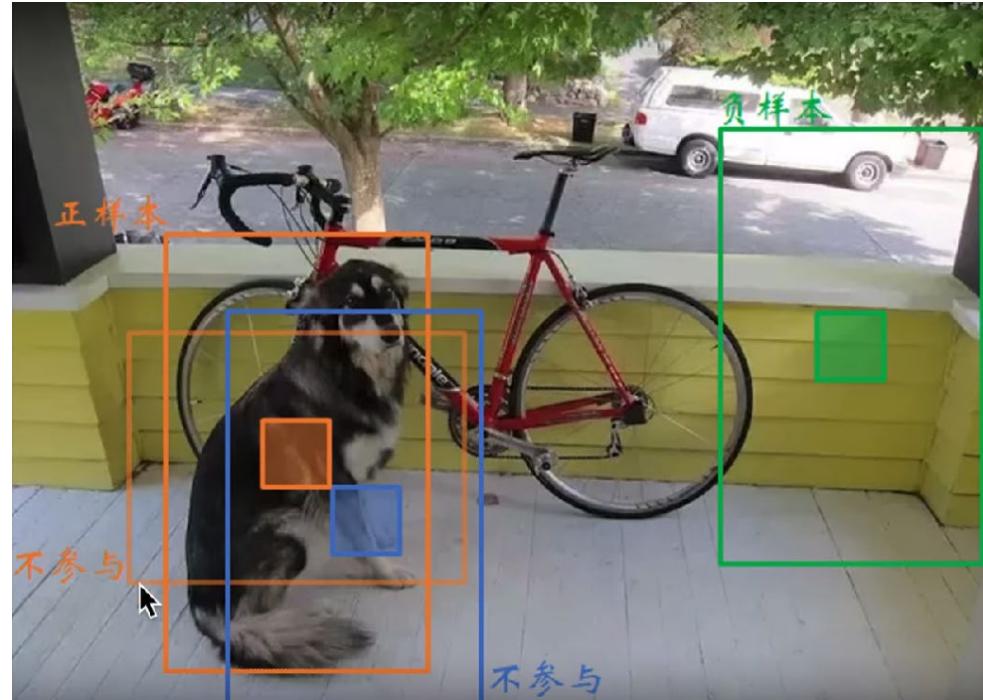
Bounding box输出的是框的位置（中心坐标与宽高），confidence以及N个类别。
anchor box只是一个尺度即只有宽高。



三次检测，每次对应的感受野不同，所以在输入为 416×416 时：

特征图	13*13	26*26	52*52
感受野	大	中	小
先验框	(116x90)	(156x198)	(373x326)
	(30x61)	(62x45)	(59x119)
	(10x13)	(16x30)	(33x23)

所以当输入为 416×416 时，实际总共有 $(52 \times 52 + 26 \times 26 + 13 \times 13) \times 3 = 10647$ 个 proposal box。



预测框：

正例 positive 与GT的IOU最大 对分类和定位产生贡献

负例 negative IOU<0.5 只对置信度学习产生贡献

忽略 ignore IOU>0.5但非最大 无贡献

多类别分类标签

YOLO9000不再常用：已有多类别标签数据集——谷歌OID数据集

原来分类网络中的softmax层都是假设一张图像或一个object只属于一个类别，但是在一些复杂场景下，一个object可能属于多个类，比如你的类别中有woman和person这两个类，那么如果一张图像中有一个woman，那么你检测的结果中类别标签就要同时有woman和person两个类，这就是多标签分类，需要用Logistic分类器来对每个类别做二分类。Logistic分类器主要用到sigmoid函数，该函数可以将输入约束在0到1的范围内，因此当一张图像经过特征提取后的某一类输出经过sigmoid函数约束后如果大于0.5，就表示该边界框负责的目标属于该类。

YOLOv3 目标检测 损失函数

预测框分为三种情况：

(不同代码实现细节可能不完全相同)

正例 positive 与 GT IOU 最大

负例 negative IOU < 0.5

忽略 ignore IOU > 0.5 但非最大

正样本坐标

λ_{coord}

$$\sum_{i=0}^{S^2} \sum_{j=0}^B$$

是否为正样本

中心点坐标

中心点宽高

$$1_{i,j}^{obj} \cdot [(b_x - \hat{b}_x)^2 + (b_y - \hat{b}_y)^2 + (b_w - \hat{b}_w)^2 + (b_h - \hat{b}_h)^2]$$

惩罚小框

正样本置信度 和类别
(标签为1)

是否为正样本 置信度标签为1

类别数 逐类别计算二元交叉熵损失

负样本置信度
(标签为0)

是否为正样本 置信度标签为1
类别概率接近于1, 太过强

$$\sum_{i=1}^n BCE(\hat{c}_i, c_i)$$

二元交叉熵 损失函数
(Binary Cross Entropy)

$$BCE = -\hat{c}_i \log(c_i) + (1 - \hat{c}_i) \log(1 - c_i)$$

遍历所有预测框
输入 416x416x3 时:
 $S^2 = 13 \times 13 + 26 \times 26 + 52 \times 52$
 $B = 3$

λ_{noobj}

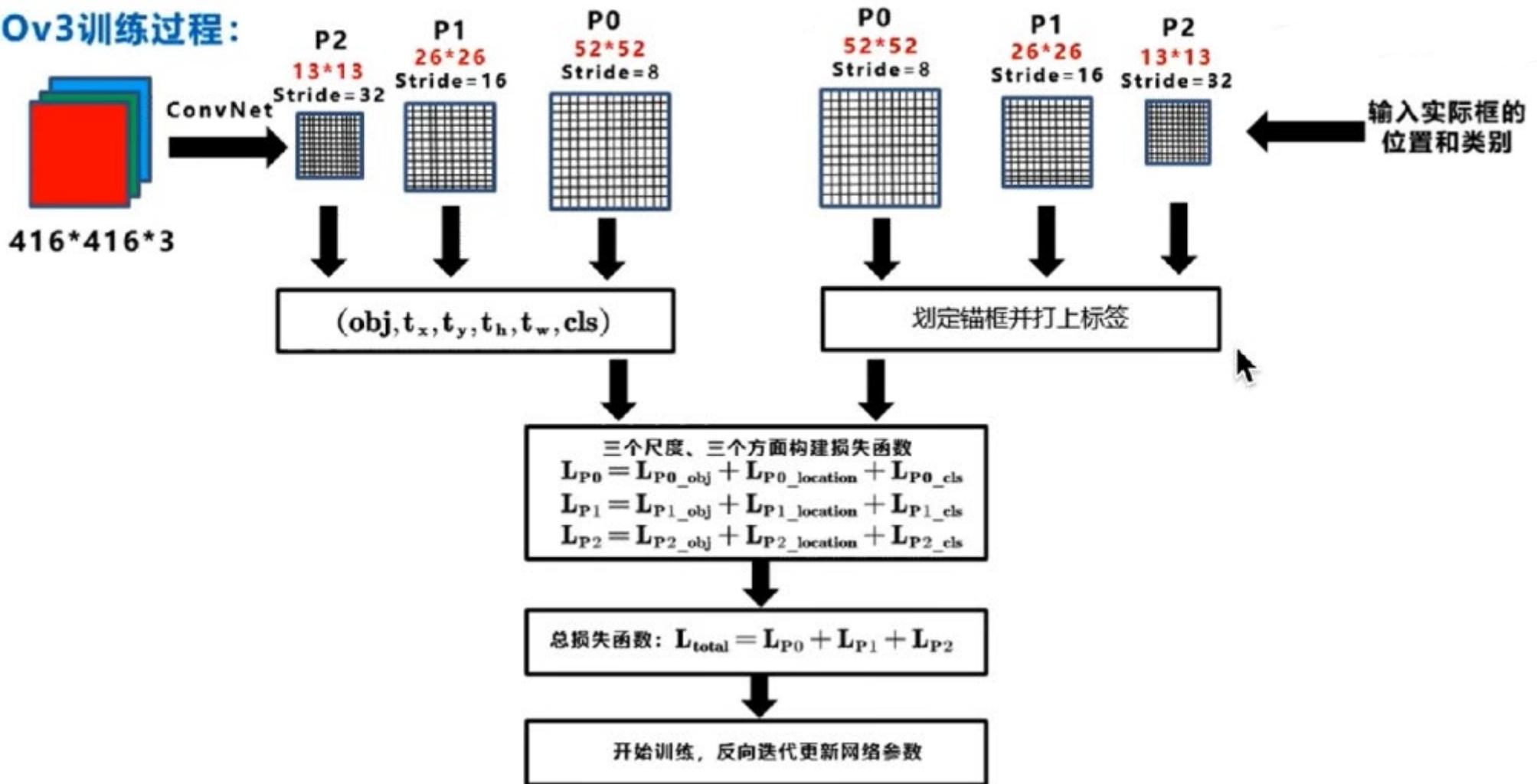
是否为负样本 置信度标签为0

$$\sum_{i=0}^{S^2} \sum_{j=0}^B 1_{i,j}^{noobj} \cdot [-\log(1 - p_c)]$$

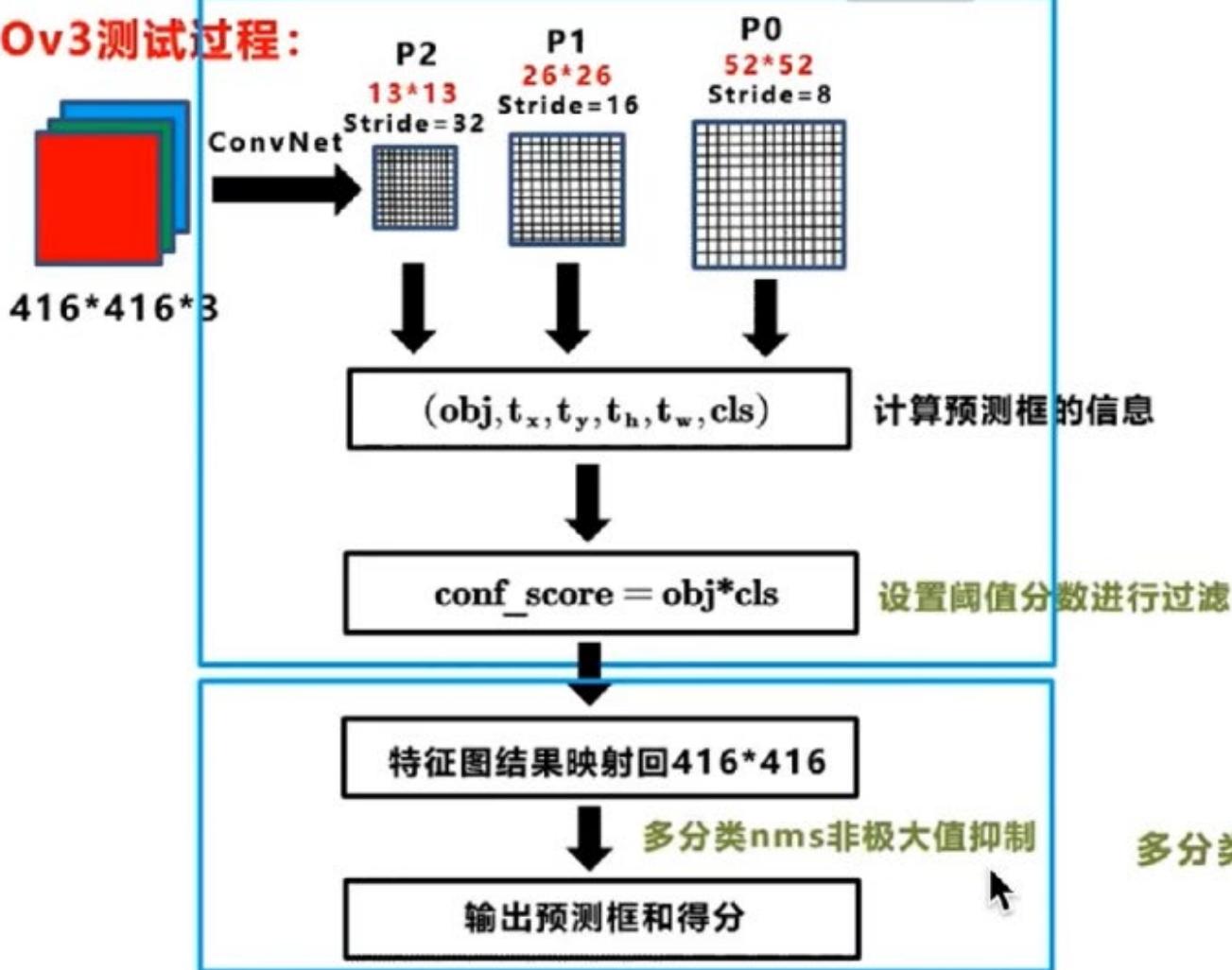
小目标/密集目标改进

- 1.Grid cell个数增加（兼容任意图片大小输入）
 - 2.Anchor
 - 3.多尺度预测（FPN）
 - 4.损失函数惩罚小框项
 - 5.网络结构（骨干网络，跨层连接）
-

YOLOv3训练过程：



YOLOv3测试过程：



第二部分：

- (1) 计算预测框的信息
- (2) 设置阈值，过滤得分低的框

第二部分：

多分类nms消除各个类别重叠较大的预测框