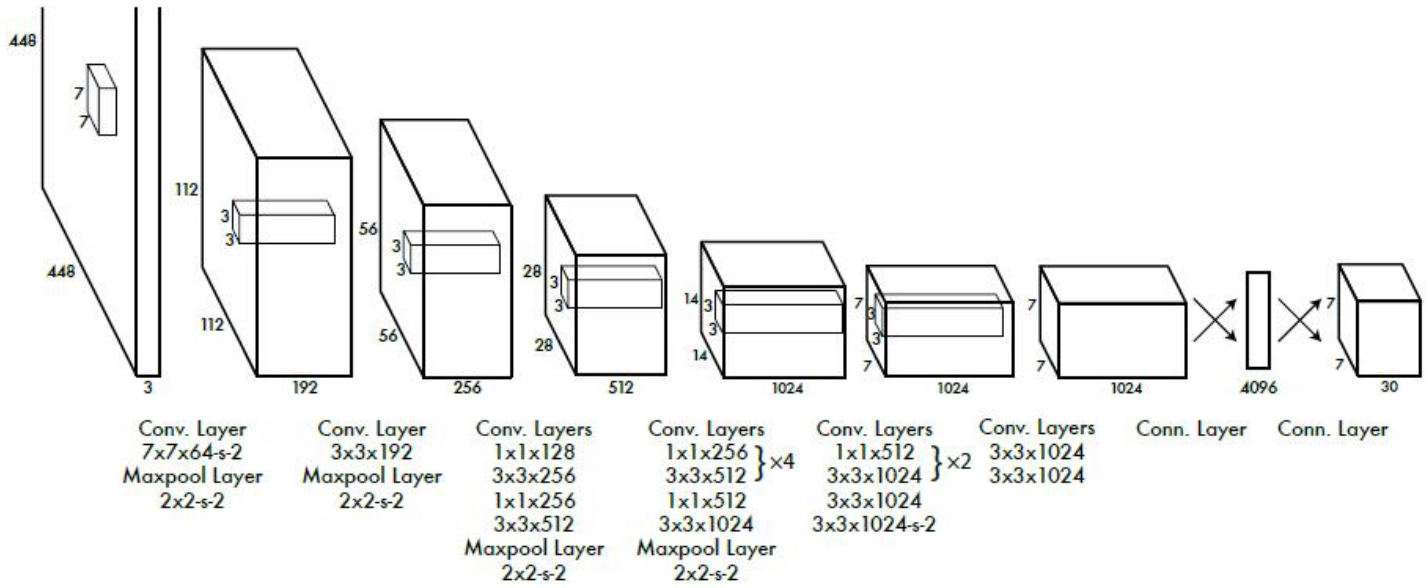
YOLO(You Only Look Once,2015-2020)

YOLOv1 (2015)

YOLO v1创造性地使用一阶结构完成了物体检测任务，直接预测物体的类别与位置，没有RPN网络，也没有类似于Anchor的预选框，因此速度很快。

网络结构



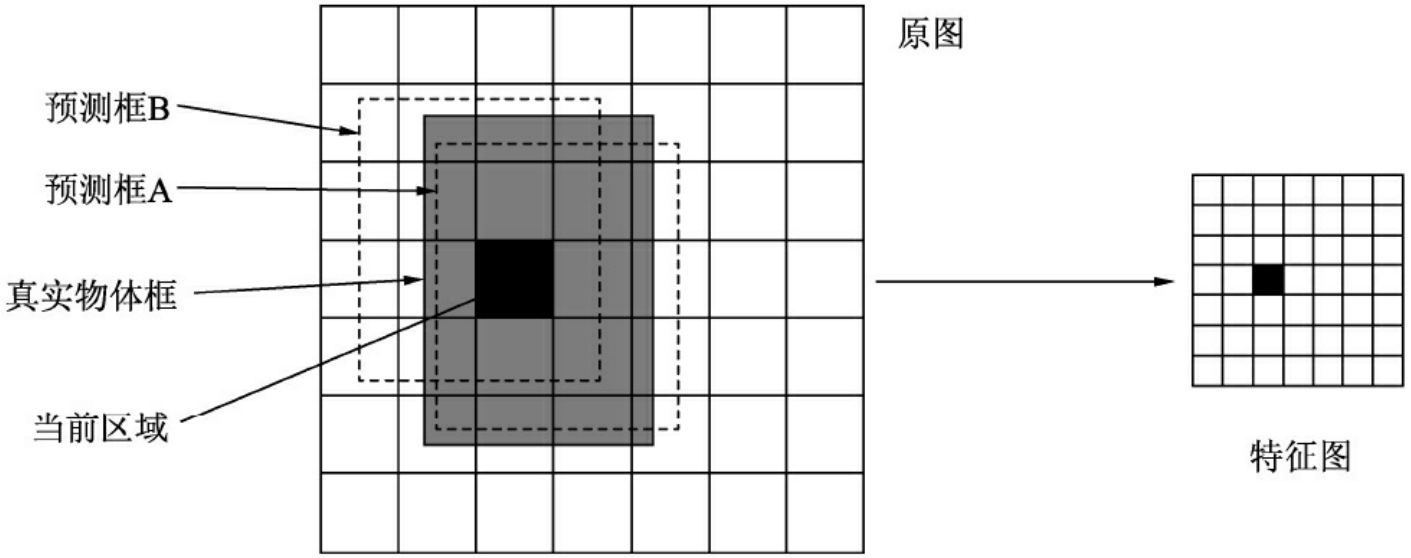
输入固定尺寸448*448，经过24个卷积层与两个全连接层，输出特征图大小为7*7*30。

- 在3*3的卷积后通常连接一个通道数更低的1*1卷积，这种方式既降低了计算量，同时也提升了模型的非线性能力。（用1×1 reduction layers 紧跟 3×3 convolutional layers 取代Goolenet的 inception modules ）
- 除了最后一层使用了线性激活函数外，其余层激活函数为Leaky ReLU。

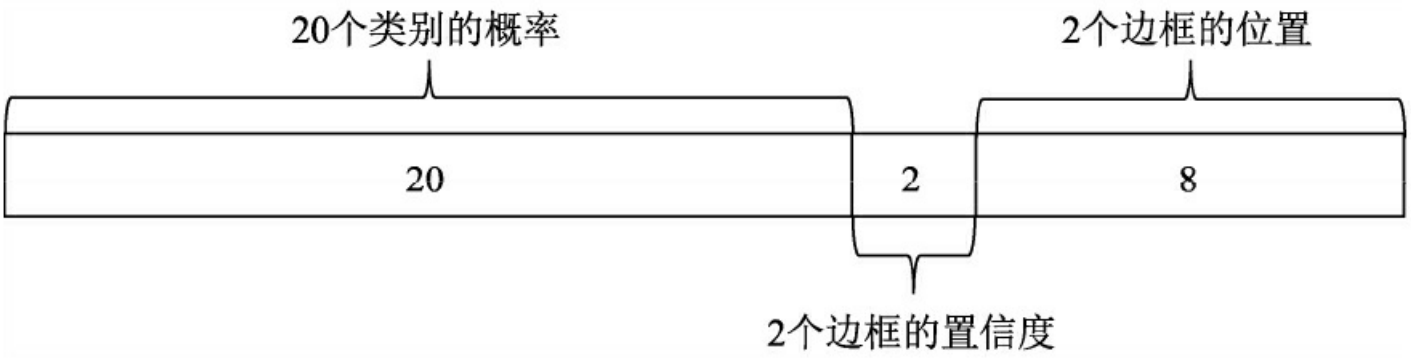
$$\begin{cases} x_i & if\ x_1 \geq 0; \\ \dfrac{x_i}{a_i} & if\ x_i < 0. \end{cases}$$

- 在训练中使用了Dropout与数据增强的方法防止过拟合。

特征图处理



YOLO v1将输入图像划分成7×7的区域，每一个区域对应于最后特征图上的一个点，该点的通道数为30，代
表了预测的30个特征（20类内置信度+2类间置信度+2*4回归框）。YOLO v1在每一个区域内预测两个边框，如图中的预测框A与B，这样整个图上一共预测7×7×2=98个框，这些边框大小与位置各不相同，基本可以覆盖整个图上可能出现的物体。



损失计算

- 当一个真实物体的中心点落在了某个区域内时，该区域就负责检测该物体。具体做法是将与该真实物体有最大IoU的边框设为正样本，这个区域的类别真值为该真实物体的类别，该边框的置信度真值为1。
- 除了上述被赋予正样本的边框，其余边框都为负样本。负样本没有类别损失与边框位置损失，只有置信度损失，其真值为0。

损失函数：

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

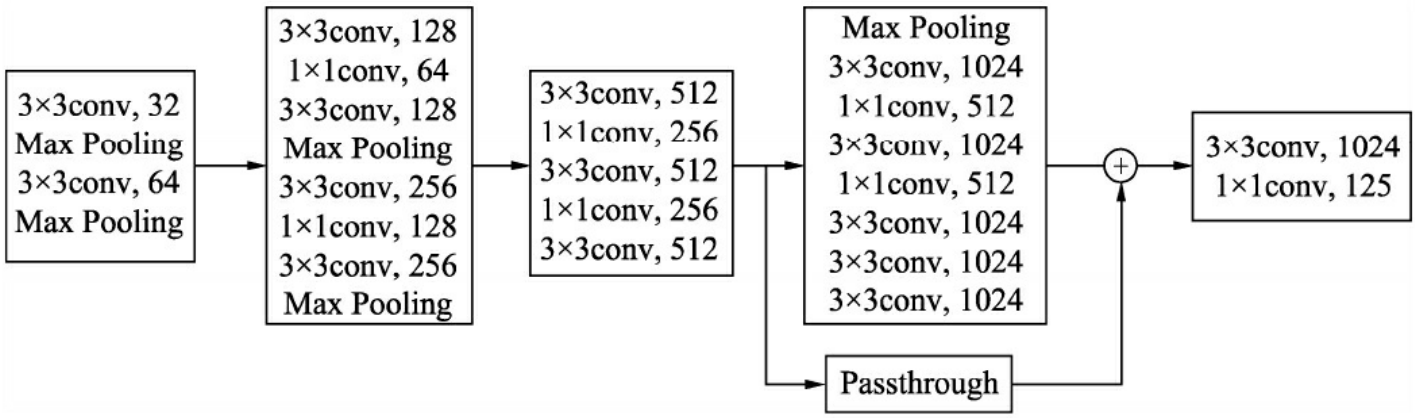
- 第一项为正样本中心点坐标的损失。λcoord的目的是为了调节位置损失的权重，YOLO v1设置λ_{coord}为5，调高了位置损失的权重。
- 第二项为正样本宽高的损失。由于宽高差值受物体尺度的影响，因此这里先对宽高进行了平方根处理，在一定程度上降低对尺度的敏感，强化了小物体的损失权重。
- 第三、四项分别为正样本与负样本的置信度损失，正样本置信度真值为1，负样本置信度为0。λ_{noobj}默认为0.5，目的是调低负样本置信度损失的权重。
- 最后一项为正样本的类别损失。

YOLOv2 (2016)

针对YOLO v1的不足，2016年诞生了YOLO v2。相比起第一个版本，YOLO v2预测更加精准（Better）、速度更快（Faster）、识别的物体类别也更多（Stronger）。

网络结构改进

使用DarkNet，精度与VGG相当，而浮点运算仅为VGG的1/5，因此速度极快。



- BN层：DarkNet使用了BN层，这一点带来了2%以上的性能提升。BN层有助于解决反向传播中的梯度消失与爆炸问题，可以加速模型的收敛，同时起到一定的正则化作用。BN层的具体位置是在每一个卷积之后，激活函数Leaky ReLU之前。
- 用连续3×3卷积替代了v1版本中的7×7卷积，这样既减少了计算量，又增加了网络深度。此外，DarkNet去掉了全连接层与Dropout层。
- Passthrough层：DarkNet还进行了深浅层特征的融合，具体方法是将浅层26×26×512的特征变换为13×13×2048，这样就可以直接与深层13×13×1024的特征进行通道拼接。这种特征融合有利于小物体的检测，也为模型带来了1%的性能提升。
- 由于YOLO v2在每一个区域预测5个边框，每个边框有25个预测值，因此最后输出的特征图通道数为125。其中，一个边框的25个预测值分别是20个类别预测、4个位置预测及1个置信度预测值。这里与v1有很大区别，v1是一个区域内的边框共享类别预测，而这里则是相互独立的类别预测值。

先验框设计

聚类提取先验框尺度

Faster RCNN中预选框（即Anchor）的大小与宽高是由人手工设计的，因此很难确定设计出的一组预选框是最贴合数据集的，也就有可能为模型性能带来负面影响。

针对此问题，YOLO v2通过在训练集上聚类来获得预选框，只需要设定预选框的数量k，就可以利用聚类算法得到最适合的k个框。在聚类时，两个边框之间的距离使用如下计算方法，即IoU越大，边框距离越近。

$$d(box, centroid) = 1 - IoU(box, centroid)$$

优化偏移公式

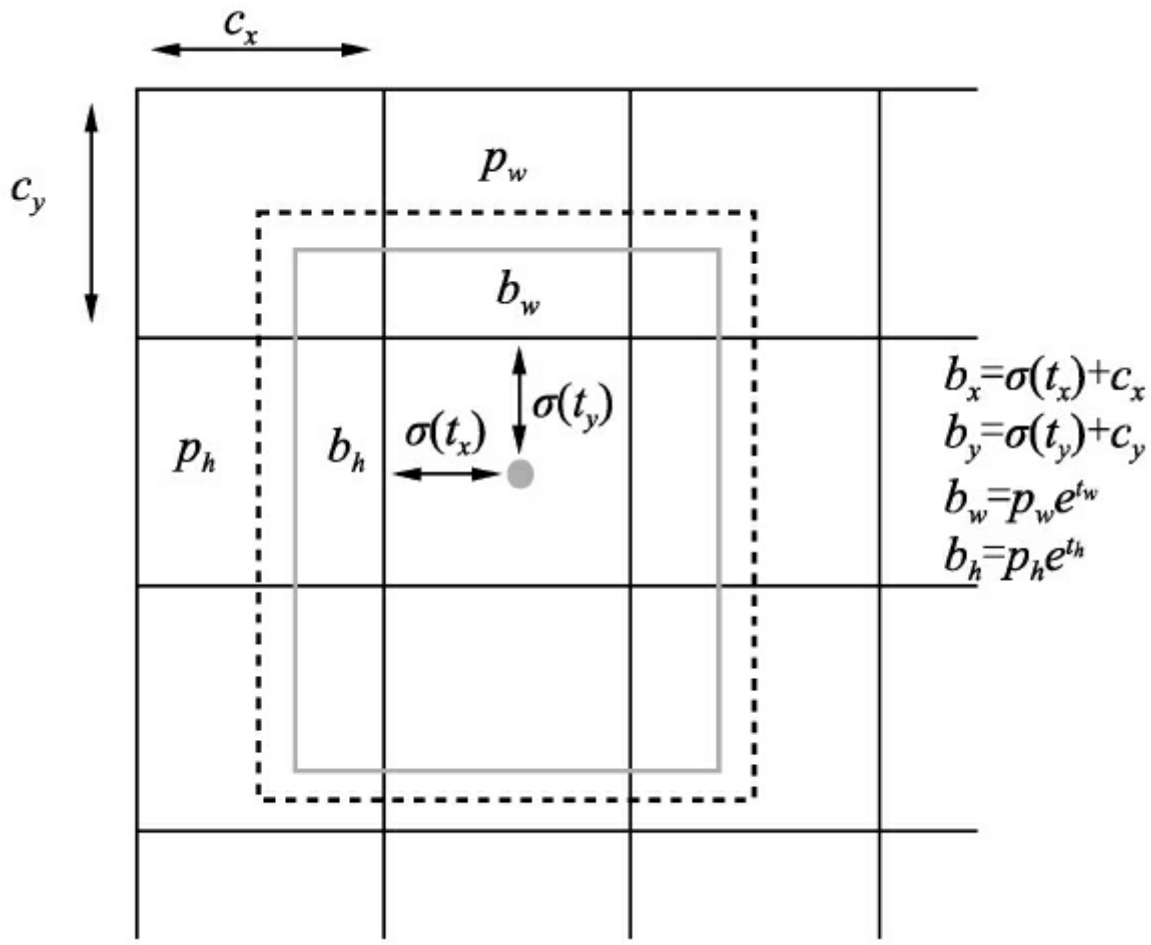
由于设计了先验框，因此设计预测先验框与真实物体的偏移量。在Faster RCNN中，中心坐标偏移公式为：

$$\begin{cases} x = (t_x \times w_a) + x_a \\ y = (t_y \times h_a) + y_a \end{cases}$$

YOLOv2认为这种预测方法没有对预测偏移进行限制，导致预测的边框中心可以出现在图像的任何位置，尤其是在训练初始阶段，模型参数还相对不确定。

因此YOLOv2提出了下面的预测公式：

$$
\begin{cases}
b_x = \sigma(t_x) + c_x \\
b_y = \sigma(t_y) + c_y \\
b_w = p_w e^{t_w} \\
b_h = p_h e^{t_h} \\
p_r(object) \times IoU(b, object) = \sigma(t_0)
\end{cases}
$$



- cx与cy代表中心点所处区域左上角的坐标，pw与ph代表了当前先验框的宽高，如图中的虚线框所示。
- σ(tx)与σ(ty)代表预测框中心点与中心点所处区域左上角坐标的距离，加上cx与cy即得到预测框的中心坐标。
- tw与th为预测的宽高偏移量。先验框的宽高乘上指数化后的宽高偏移量，即得到预测框的宽高。
- 公式中的σ代表Sigmoid函数，作用是将坐标偏移量化到(0,1)区间，这样得到的预测边框的中心坐标bx、by会限制在当前区域内，保证一个区域只预测中心点在该区域内的物体，有利于模型收敛。
- YOLO v1将预测值t0作为边框的置信度，而YOLO v2则是将做Sigmoid变换后的σ(t0)作为真正的置信度预测值。

损失函数

通过IoU设定决策阈值，以区分正、负样本，下面为带有先验框的损失函数：

$$
\begin{aligned}
Loss_t = & \sum_{i=0}^{W}\sum_{j=0}^{H}\sum_{k=0}^{A}(1_{\max IoU<Thresh} \times \lambda_{noobj} \times (-b_{ijk}^{o})^2 \\
& + 1_{t<12800} \times \lambda_{prior} \times \sum_{r\in(x,y,w,h)} (prior_k^r - b_{ijk}^r)^2 \\
& + 1_k^{truth} \times \lambda_{coord} \times \sum_{r\in(x,y,w,h)} (truth^r - b_{ijk}^r)^2 \\
& + \lambda_{obj} \times (IoU_{turth}^k - b_{ijk}^o)^2 \\
& + \lambda_{class} \times \sum_{c=1}^{C}(truth^c - b_{ijk}^c)^2)
\end{aligned}
$$

- 第一项为负样本的置信度损失，公式中$1_{\max IoU<Thresh}$表示最大IoU小于阈值，即负样本的边框，λ_{noobj}是负样本损失的权重，b_{ijk}^o为置信度$\sigma(t_o)$。
- 第二项为先验框与预测框的损失，只存在于前12800次迭代中，目的是使预测框先收敛于先验框，模型更稳定。
- 第三项为正样本的位置损失，表示筛选出的正样本，为权重。
- 后两项分别为正样本的置信度损失与类别损失，为置信度的真值。

工程技巧

多尺度训练

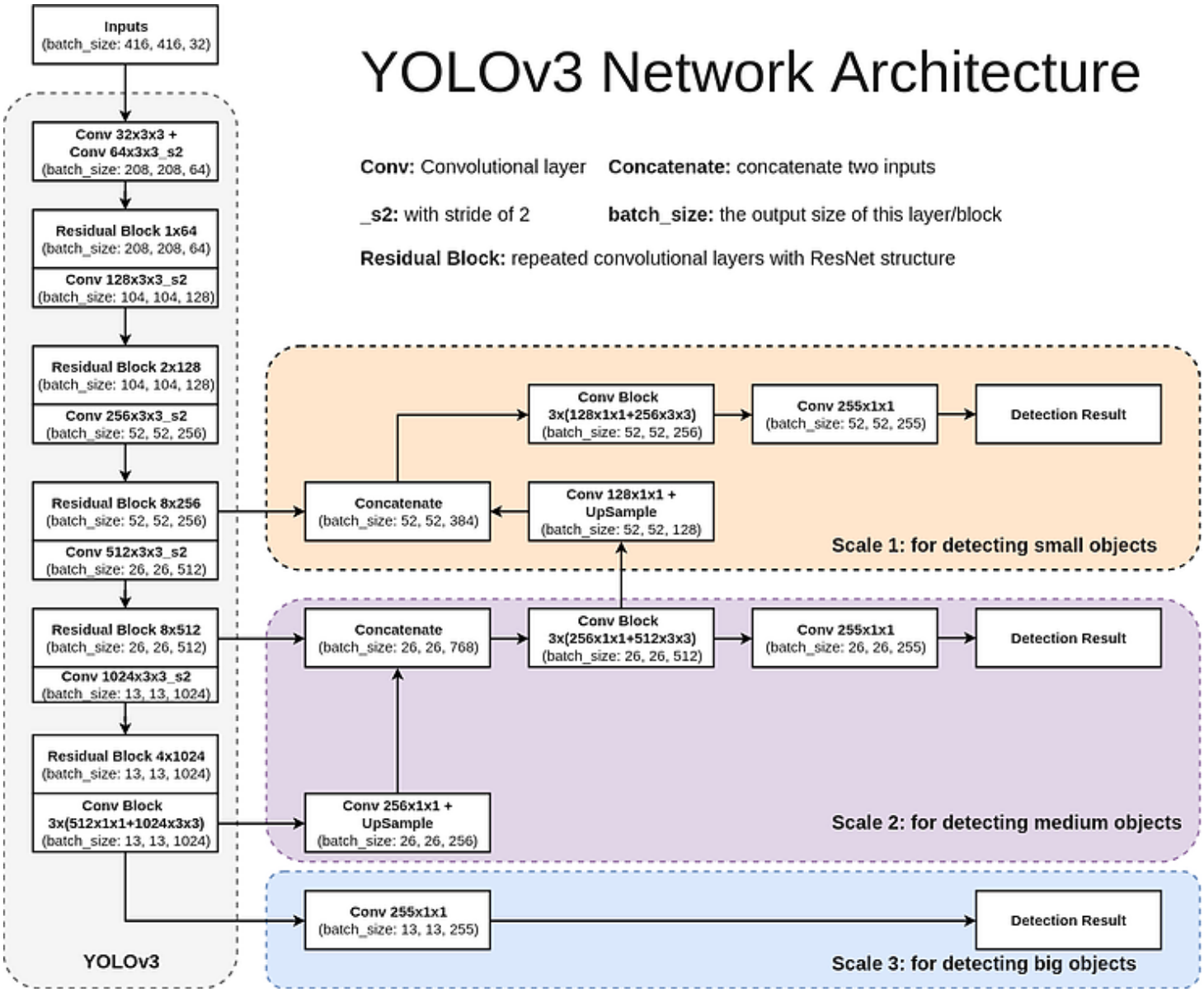
由于移除了全连接层，YOLOv2可以接受任意尺寸的输入图片。在训练阶段，为了使的模型对不同尺度物体的鲁棒，YOLOv2采取了多种尺度的图片作为训练的输入。

由于下采样率为32，为了满足整除的需求，YOLO v2选取的输入尺度集合为{320,352,384,...,608}，这样训练出的模型可以预测多个尺度的物体。并且，输入图片的尺度越大则精度越高，尺度越低则速度越快，因此YOLO v2多尺度训练出的模型可以适应多种不同的场景要求。

YOLOv3 (2018)

YOLO v3将当今一些较好的检测思想融入到了YOLO中，在保持速度优势的前提下，进一步提升了检测精度，尤其是对小物体的检测能力。

新网络结构DarkNet-53



值得注意的是，YOLO v3的速度并没有之前的版本快，而是在保证实时性的前提下追求检测的精度。如果追求速度，YOLO v3提供了一个更轻量化的网络tiny-DarkNet，在模型大小与速度上，实现了SOTA（State of the Art）的效果。

多尺度预测

从图中可以看到，YOLO v3输出了3个大小不同的特征图，从上到下分别对应深层、中层与浅层的特征。深层的特征图尺寸小，感受野大，有利于检测大尺度物体，而浅层的特征图则与之相反，更便于检测小尺度物体，这一点类似于FPN结构。

Softmax改为Logistic

YOLO v3的另一个改进是使用了Logistic函数代替Softmax函数，以处理类别的预测得分。原因在于，Softmax函数输出的多个类别预测之间会相互抑制，只能预测出一个类别，而Logistic分类器相互独立，可以实现多类别的预测。

实验证明，Softmax可以被多个独立的Logistic分类器取代，并且准确率不会下降，这样的设计可以实现物体的多标签分类，例如一个物体如果是Women时，同时也属于Person这个类别。