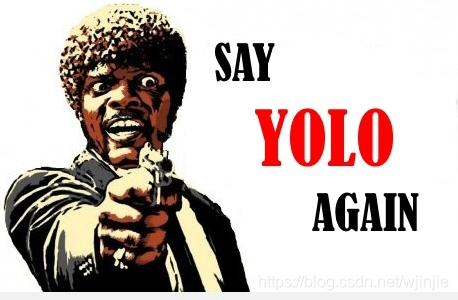
本文只是原文链接：<https://blog.csdn.net/wjinjie/article/details/107509243>的搬运罢了！！！



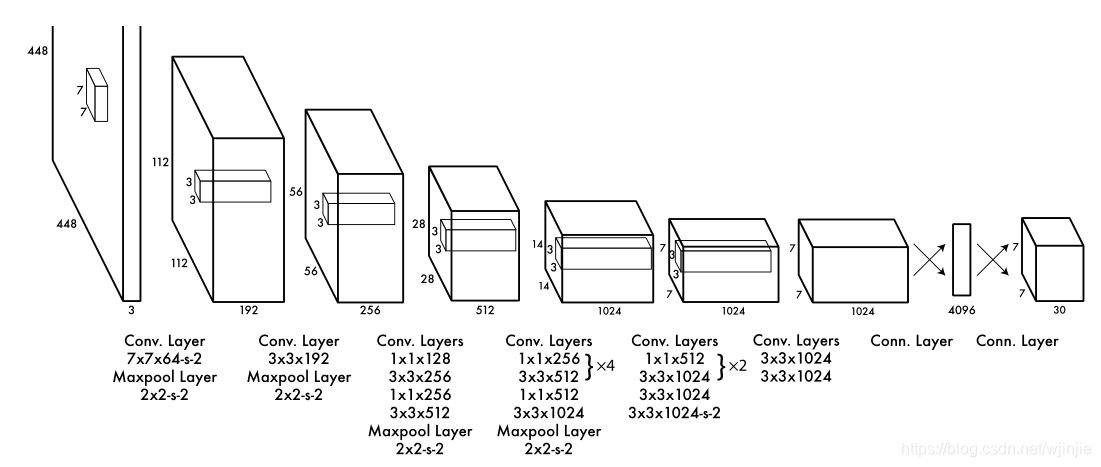
**一、开山之作：YOLOv1**

**1.1 简介**

在YOLOv1提出之前，R-CNN系列算法在目标检测领域独占鳌头。R-CNN系列检测精度高，但是由于其网络结构是双阶段（two-stage）的特点，使得它的检测速度不能满足实时性，饱受诟病。为了打破这一僵局，设计一种速度更快的目标检测器是大势所趋。

2016年，Joseph Redmon、Santosh Divvala、Ross Girshick等人提出了一种单阶段（one-stage）的目标检测网络。它的检测速度非常快，每秒可以处理45帧图片，能够轻松地实时运行。由于其速度之快和其使用的特殊方法，**作者将其取名为：You Only Look Once（也就是我们常说的YOLO的全称）**，并将该成果发表在了CVPR 2016上，从而引起了广泛地关注。

**1.2 网络结构**



现在看来，YOLOv1的网路结构非常明晰，是一种传统的one-stage的卷积神经网络：

**网络输入：**448×448×3的彩色图片。

（这里的意思是有一张448\*448的彩色照片，同时有红绿蓝三个通道）

**中间层：**由若干卷积层和最大池化层组成，用于提取图片的抽象特征。

**全连接层：**由两个全连接层组成，用来预测目标的位置和类别概率值。

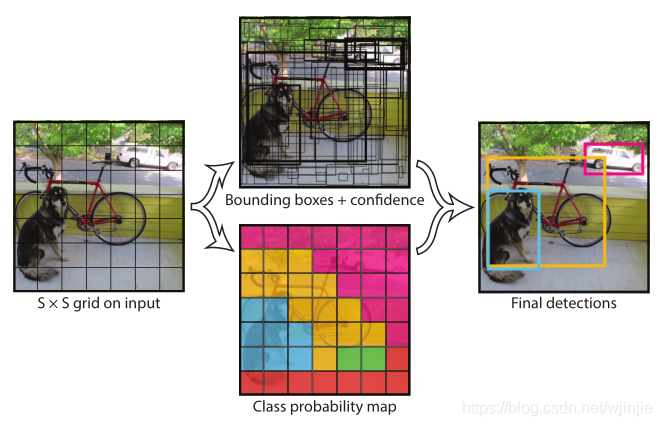
**网络输出：**7×7×30的预测结果。

（下面会讲解为什么输出的张量是7\*7\*30）

**1.3 实现细节**

**（1）检测策略**

YOLOv1采用的是“分而治之”的策略，**将一张图片平均分成7×7个网格，每个网格分别负责预测中心点落在该网格内的目标**。回忆一下，在Faster R-CNN中，是通过一个RPN来获得目标的感兴趣区域，这种方法精度高，但是需要额外再训练一个RPN网络，这无疑增加了训练的负担。**在YOLOv1中，通过划分得到了7×7个网格，这49个网格就相当于是目标的感兴趣（ROI）区域**。通过这种方式，我们就不需要再额外设计一个RPN网络，这正是YOLOv1作为单阶段网络的简单快捷之处！



具体实现过程如下：

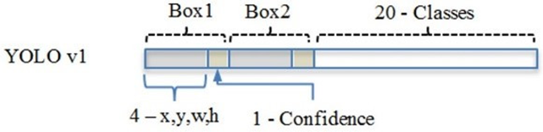
**1.**将一幅图像分成 S×S个网格（grid cell），如果某个 object 的中心落在这个网格中，则这个网格就负责预测这个object。

**2.**每个网格要预测 B 个bounding box，每个 bounding box 要预测 (x, y, w, h) 和 confidence 共5个值。

**3.**每个网格还要预测一个类别信息，记为 C 个类。

**4.**总的来说，S×S 个网格，每个网格要预测 B个bounding box ，还要预测 C 个类。网络输出就是一个 S × S × (5×B+C) 的张量。

**在实际过程中，YOLOv1把一张图片划分为了7×7个网格，并且每个网格预测2个Box（Box1和Box2），20个类别。所以实际上，S=7，B=2，C=20。那么网络输出的shape也就是：7×7×30。**

****

**（2）目标损失函数**

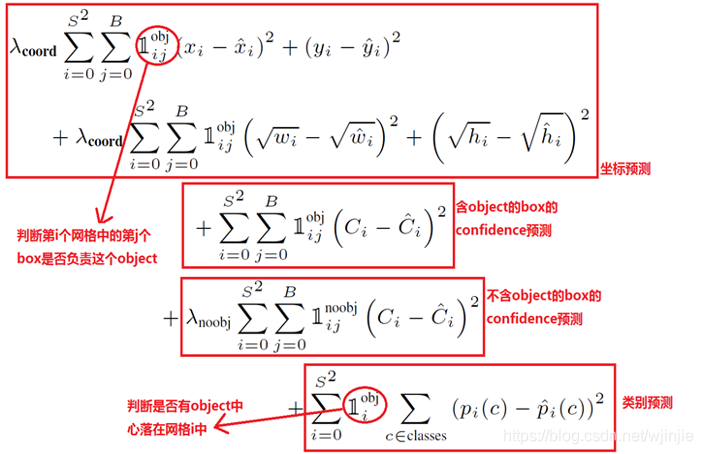
损失由三部分组成，分别是：坐标预测损失、置信度预测损失、类别预测损失。

使用的是差方和误差。需要注意的是，w和h在进行误差计算的时候取的是它们的平方根，原因是对不同大小的bounding box预测中，相比于大bounding box预测偏一点，小box预测偏一点更不能忍受。而差方和误差函数中对同样的偏移loss是一样。 为了缓和这个问题，作者用了一个比较取巧的办法，就是将bounding box的w和h取平方根代替原本的w和h。

定位误差比分类误差更大，所以增加对定位误差的惩罚，使

​

在每个图像中，许多网格单元不包含任何目标。训练时就会把这些网格里的框的“置信度”分数推到零，这往往超过了包含目标的框的梯度。从而可能导致模型不稳定，训练早期发散。因此要减少了不包含目标的框的置信度预测的损失，使



​**1.4 性能表现**

**（1）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）。

**（2）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。

**二、更快更准：YOLOv2**

**2.1 简介**

2017年，作者 Joseph Redmon 和 Ali Farhadi 在 YOLOv1 的基础上，进行了大量改进，提出了 YOLOv2 和 YOLO9000。**重点解决YOLOv1召回率和定位精度方面的不足。**

YOLOv2 是一个先进的目标检测算法，比其它的检测器检测速度更快。除此之外，该网络可以适应多种尺寸的图片输入，并且能在检测精度和速度之间进行很好的权衡。

相比于YOLOv1是利用全连接层直接预测Bounding Box的坐标，**YOLOv2借鉴了Faster R-CNN的思想，引入Anchor机制**。利用**K-means聚类**的方法在训练集中聚类计算出更好的Anchor模板，大大提高了算法的召回率。同时结合图像细粒度特征，将浅层特征与深层特征相连，有助于对小尺寸目标的检测。

**YOLO9000 使用 WorldTree 来混合来自不同资源的训练数据**，并使用联合优化技术同时在ImageNet和COCO数据集上进行训练，能够实时地检测超过9000种物体。由于 YOLO9000 的主要检测网络还是YOLOv2，所以这部分以讲解应用更为广泛的YOLOv2为主。

**2.2.1 K-means聚类**

**K-means聚类是一种常用的无监督学习算法**，用于将数据点分成具有相似特征的不同组（簇）。它的目标是将数据分成K个簇，使得每个数据点都属于离它最近的簇中心所代表的簇，同时最小化所有数据点到其所属簇中心的距离的平方和（也称为簇内平方和）。这意味着它尝试将数据点分成K个簇，以便每个簇内的数据点尽可能相似，而不同簇之间的数据点尽可能不同。

**K-means算法的步骤大致如下：**

1.随机初始化K个簇中心（通常是从数据集中选择K个随机数据点作为初始簇中心）。

2.将每个数据点分配给最近的簇中心，形成K个簇。

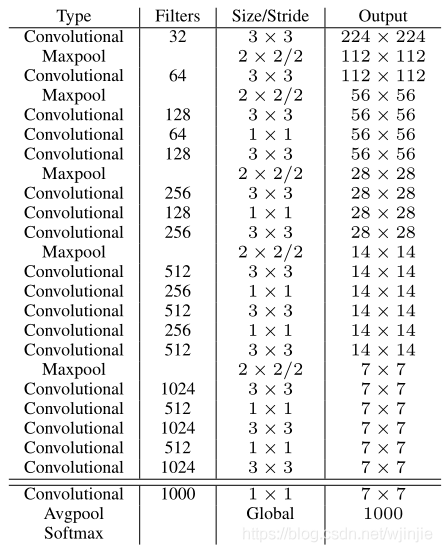
3.更新每个簇的中心为该簇中所有数据点的平均值。

4.重复步骤2和3，直到簇中心不再变化或者达到预设的迭代次数。

K-means算法的一个重要注意点是，**它可能会收敛到局部最优解，因此多次运行算法并选择最好的聚类结果是一个常见的做法**。此外，K值的选择也是一个关键问题，通常需要通过领域知识或者使用一些评估指标来确定最优的K值。

2.2 网络结构

YOLOv2 采用 Darknet-19 作为特征提取网络，其整体结构如下：



**改进后的YOLOv2: Darknet-19，总结如下：**

1.与VGG相似，使用了很多3×3卷积核；并且每一次池化后，下一层的卷积核的通道数 = 池化输出的通道 × 2。

2.在每一层卷积后，都增加了批量标准化（Batch Normalization）进行预处理。

3.采用了降维的思想，把1×1的卷积置于3×3之间，用来压缩特征。

4.在网络最后的输出增加了一个global average pooling层。

整体上采用了19个卷积层，5个池化层。

**关于Darknet-19：**

Darknet-19是一个用于目标检测、图像分类和其他计算机视觉任务的神经网络模型。它是由Joseph Redmon开发的，作为Darknet深度学习框架的一部分。Darknet-19由19个卷积层组成，因此得名。这个模型通常用于处理图像数据，尤其在目标检测领域中表现出色。

Darknet-19的架构相对较轻量，适合于在资源受限的环境中部署，同时保持较高的性能。它在一些图像分类和目标检测的基准数据集上表现出良好的性能，是一个被广泛使用的模型之一。

**关于1x1卷积核是如何降维的？**

1x1卷积核通过给每一个维度的二维张量进行权重计算，从而实现通道的下降。

例子：如果要将3通道下降为2通道，那么需要一个1\*1\*3的卷积核，通过相乘再相加最终得到一个二维张量。最后将输出通道设置为2，可以得到两个二维张量，也就将3通道下降为了2通道。

**为了更好的说明，这里将 Darknet-19 与 YOLOv1、VGG16网络进行对比：**

**VGG-16：** 大多数检测网络框架都是以VGG-16作为基础特征提取器，它功能强大，准确率高，但是计算复杂度较大，所以速度会相对较慢。因此YOLOv2的网络结构将从这方面进行改进。

**YOLOv1：** 基于GoogLeNet的自定义网络（具体看上周报告），比VGG-16的速度快，但是精度稍不如VGG-16。

**Darknet-19：** 速度方面，处理一张图片仅需要55.8亿次运算，相比于VGG306.9亿次，速度快了近6倍。精度方面，在ImageNet上的测试精度为：top1准确率为72.9%，top5准确率为91.2%。

**2.3 改进方法**

**（1）Batch Normalization**

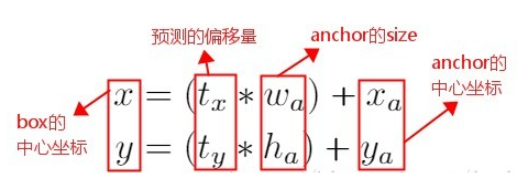
Batch Normalization 简称 BN ，意思是批量标准化。2015年由 Google 研究员在论文《Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift》中提出。

BN 对数据进行预处理（统一格式、均衡化、去噪等）能够大大提高训练速度，提升训练效果。**基于此，YOLOv2 对每一层输入的数据都进行批量标准化，这样网络就不需要每层都去学数据的分布，收敛会变得更快。**

**（2）引入 Anchor Box 机制**

在YOLOv1中，作者设计了端对端的网路，直接对边界框的位置（x, y, w, h）进行预测。这样做虽然简单，但是由于没有类似R-CNN系列的推荐区域，所以网络在前期训练时非常困难，很难收敛。**于是，自YOLOv2开始，引入了 Anchors box 机制，希望通过提前筛选得到的具有代表性先验框Anchors，使得网络在训练时更容易收敛。**

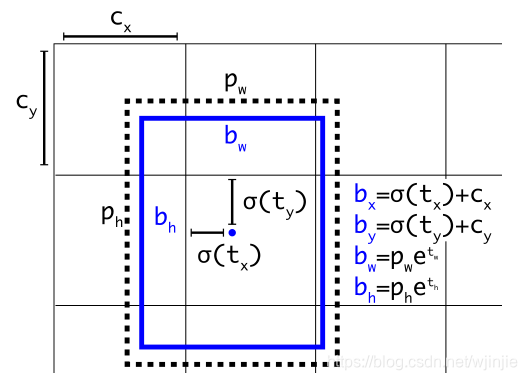
在 Faster R-CNN 算法中，是通过预测 bounding box 与 ground truth 的位置偏移值、间接得到bounding box的位置。其公式如下：



这个公式是无约束的，预测的边界框很容易向任何方向偏移。因此，每个位置预测的边界框可以落在图片任何位置，这会导致模型的不稳定性。

因此 YOLOv2 在此方法上进行了一点改变：预测边界框中心点相对于该网格左上角坐标（ C x , C y ）的相对偏移量，同时为了将bounding box的中心点约束在当前网格中，使用 sigmoid 函数将 , 归一化处理，将值约束在0-1，这使得模型训练更稳定。

下图为 Anchor box 与 bounding box 转换示意图，其中蓝色的是要预测的bounding box，黑色虚线框是Anchor box。

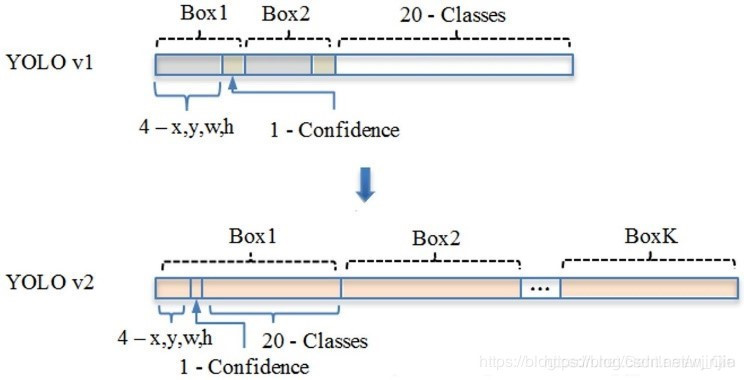


YOLOv2 在最后一个卷积层输出 13×13 的 feature map，意味着一张图片被分成了13×13个网格。每个网格有5个anchor box来预测5个bounding box，每个bounding box预测得到5个值： , , 和（类似YOLOv1的confidence）。

置信度的计算公式为：  
IOU 的全称是 Intersection over Union，即“交集 over 联合”，常用于评估对象检测算法的性能。**意思是预测框和实际框之间的重合度的百分比。**



**（3）输出张量的改变**



**YOLOv1 有一个致命的缺陷就是：**一张图片被分成7×7的网格，一个网格只能预测一个类，当一个网格中同时出现多个类时，就无法检测出所有类。针对这个问题，YOLOv2做出了相应的改进：

1.首先将YOLOv1网络的FC层和最后一个Pooling层去掉，使得最后的卷积层的输出可以有更高的分辨率特征。

2.然后缩减网络，用416×416大小的输入代替原来的448×448，使得网络输出的特征图有奇数大小的宽和高，进而使得每个特征图在划分单元格的时候只有一个中心单元格（Center Cell）。YOLOv2通过5个Pooling层进行下采样，得到的输出是13×13的像素特征。

3.借鉴Faster R-CNN，YOLOv2通过引入Anchor Boxes，预测Anchor Box的偏移值与置信度，而不是直接预测坐标值。

4.采用Faster R-CNN中的方式，每个Cell可预测出9个Anchor Box，共13×13×9=1521个（YOLOv2确定Anchor Boxes的方法见是维度聚类，每个Cell选择5个Anchor Box）。

**比YOLOv1预测的98个bounding box 要多很多，因此在定位精度方面有较好的改善。**

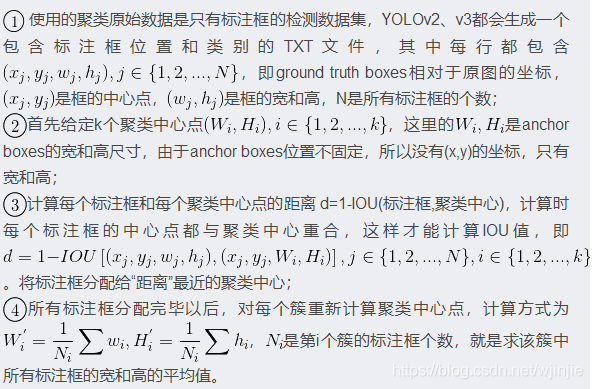
**（4）聚类方法选择Anchors**

Faster R-CNN 中 Anchor Box 的大小和比例是按经验设定的，不具有很好的代表性。若一开始就选择了更好的、更有代表性的先验框Anchor Boxes，那么网络就更容易学到准确的预测位置了！

YOLOv2 使用 K-means 聚类方法得到 Anchor Box 的大小，选择具有代表性的尺寸的Anchor Box进行一开始的初始化。传统的K-means聚类方法使用标准的欧氏距离作为距离度量，这意味着大的box会比小的box产生更多的错误。因此这里使用其他的距离度量公式。聚类的目的是使 Anchor boxes 和临近的 ground truth boxes有更大的IOU值，因此自定义的距离度量公式为 ：



到聚类中心的距离越小越好，但IOU值是越大越好，所以使用 1 - IOU；这样就保证距离越小，IOU值越大。具体实现方法如下：

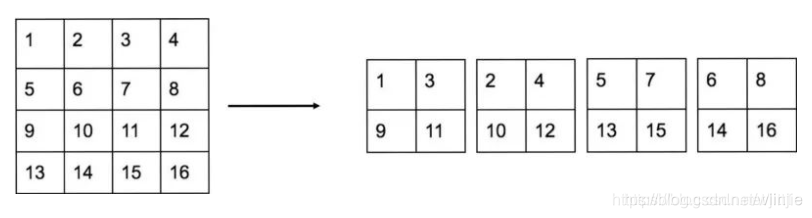


**（5）Fine-Grained Features**

细粒度特征，可理解为不同层之间的特征融合。YOLOv2通过添加一个Passthrough Layer，把高分辨率的浅层特征连接到低分辨率的深层特征（把特征堆积在不同Channel中）而后进行融合和检测。具体操作是：先获取前层的26×26的特征图，将其同最后输出的13×13的特征图进行连接，而后输入检测器进行检测（而在YOLOv1中网络的FC层起到了全局特征融合的作用），以此来提高对小目标的检测能力。

Passthrough层与ResNet网络的shortcut类似，以前面更高分辨率的特征图为输入，然后将其连接到后面的低分辨率特征图上。前面的特征图维度是后面的特征图的2倍，passthrough层抽取前面层的每个2×2的局部区域，然后将其转化为channel维度，对于26×26×512的特征图，经Passthrough层处理之后就变成了13×13×2048的新特征图（特征图大小降低4倍，而channles增加4倍），这样就可以与后面的13×13×1024特征图连接

如下图所示：



**三、巅峰之作：YOLOv3**

**3.1 简介**

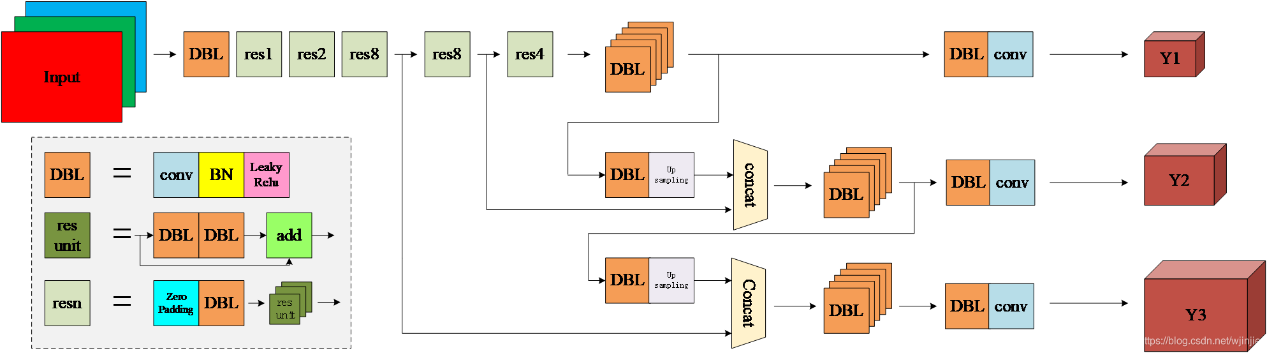
2018年，作者 Redmon 又在 YOLOv2 的基础上做了一些改进。特征提取部分采用darknet-53网络结构代替原来的darknet-19，利用特征金字塔网络结构实现了多尺度检测，分类方法使用逻辑回归代替了softmax，在兼顾实时性的同时保证了目标检测的准确性。

从YOLOv1到YOLOv3，每一代性能的提升都与backbone（骨干网络）的改进密切相关。在YOLOv3中，作者不仅提供了**darknet-53**，还提供了轻量级的**tiny-darknet**。如果你想检测精度与速度兼具，可以选择darknet-53作为backbone；如果你希望达到更快的检测速度，精度方面可以妥协，那么tiny-darknet是你很好的选择。总之，YOLOv3的灵活性使得它在实际工程中得到很多人的青睐！

**3.2 网络结构**

相比于 YOLOv2 的 骨干网络，YOLOv3 进行了较大的改进。借助残差网络的思想，YOLOv3 将原来的 darknet-19 改进为darknet-53。论文中给出的整体结构如下：

**为了更加清晰地了解darknet-53的网络结构，可以看下面这张图：**

****

**为了更好的理解此图，下面我将主要单元进行说明：**

**DBL：** 一个卷积层、一个批量归一化层和一个Leaky ReLU组成的基本卷积单元。

**res unit：** 输入通过两个DBL后，再与原输入进行add；这是一种常规的残差单元。残差单元的目的是为了让网络可以提取到更深层的特征，同时避免出现梯度消失或爆炸。

**resn：** 其中的n表示n个res unit；所以 resn = Zero Padding + DBL + n × res unit 。

**concat：** 将darknet-53的中间层和后面的某一层的上采样进行张量拼接，达到多尺度特征融合的目的。这与残差层的add操作是不一样的，拼接会扩充张量的维度，而add直接相加不会导致张量维度的改变。

**Y1、Y2、Y3：** 分别表示YOLOv3三种尺度的输**出。**

**与darknet-19对比可知，darknet-53主要做了如下改进：**

1.没有采用最大池化层，转而采用步长为2的卷积层进行下采样。

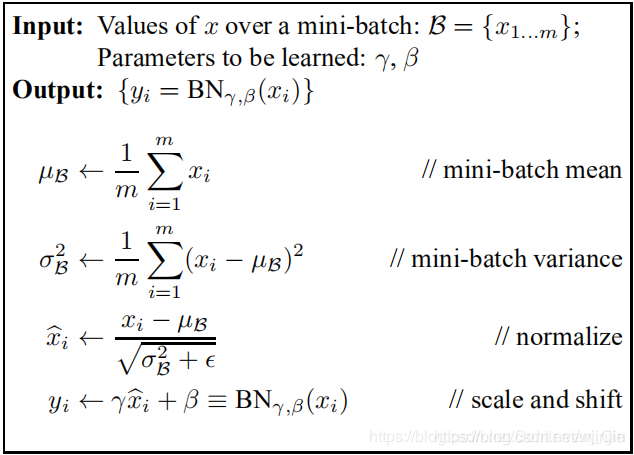
2.为了防止过拟合，在每个卷积层之后加入了一个BN层和一个Leaky ReLU。

3.引入了残差网络的思想，目的是为了让网络可以提取到更深层的特征，同时避免出现梯度消失或爆炸。

4.将网络的中间层和后面某一层的上采样进行张量拼接，达到多尺度特征融合的目的。

**BN算法的实现：**

在卷积或池化之后，激活函数之前，对每个数据输出进行标准化，实现方式如下图所示：

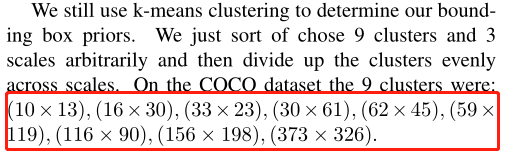


如上图所示，前三行是对Batch进行数据归一化（如果一个Batch中有训练集每个数据，那么同一Batch内数据近似代表了整体训练数据），第四行引入了附加参数 γ 和 β，这两个参数的具体取值可以参考上面提到的 Batch Normalization 这篇论文。

**3.3 改进之处**

**YOLOv3最大的改进之处还在于网络结构的改进，由于上面已经讲过。因此下面主要对其它改进方面进行介绍：**

为了能够预测多尺度的目标，YOLOv3 选择了三种不同shape的Anchors，同时每种Anchors具有三种不同的尺度，一共9种不同大小的Anchors。在COCO数据集上选择的9种Anchors的尺寸如下图红色框所示：

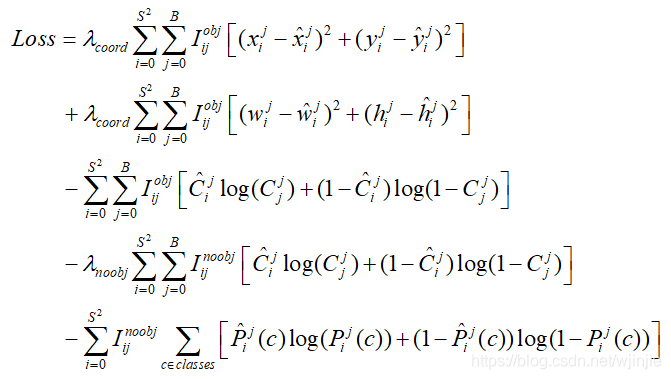
****

借鉴特征金字塔网的思想，YOLOv3设计了3种不同尺度的网络输出Y1、Y2、Y3，目的是预测不同尺度的目标。由于在每一个尺度网格都负责预测3个边界框，且COCO数据集有80个类。所以网络输出的张量应该是：N ×N ×[3∗(4 + 1 + 80)]。由下采样次数不同，得到的N不同，最终Y1、Y2、Y3的shape分别为：[13, 13, 255]、[26, 26, 255]、[52, 52, 255]。可见参见原文：



**（2）损失函数**

对于神经网络来说，损失函数的设计也非常重要。但是YOLOv3这篇文中并没有直接给出损失函数的表达式。下面通过对源码的分析，给出YOLOv3的损失函数表达式：



对比**YOLOv1**中的损失函数很容易知道：位置损失部分并没有改变，仍然采用的是**sum-square error**的损失计算方法。但是置信度损失和类别预测均由原来的sum-square error改为了交叉熵的损失计算方法。**对于类别以及置信度的预测，使用交叉熵的效果应该更好！**

**（3）多标签分类**

YOLOv3在类别预测方面将YOLOv2的单标签分类改进为多标签分类，在网络结构中将YOLOv2中用于分类的softmax层修改为逻辑分类器。在YOLOv2中，算法认定一个目标只从属于一个类别，根据网络输出类别的得分最大值，将其归为某一类。然而在一些复杂的场景中，单一目标可能从属于多个类别。

比如在一个交通场景中，某目标的种类既属于汽车也属于卡车，如果用softmax进行分类，softmax会假设这个目标只属于一个类别，这个目标只会被认定为汽车或卡车，这种分类方法就称为单标签分类。如果网络输出认定这个目标既是汽车也是卡车，这就被称为多标签分类。

为实现多标签分类就需要用逻辑分类器来对每个类别都进行二分类。逻辑分类器主要用到了sigmoid函数，它可以把输出约束在0到1，如果某一特征图的输出经过该函数处理后的值大于设定阈值，那么就认定该目标框所对应的目标属于该类。

**四、大神接棒：YOLOv4**

YOLOv4在传统的YOLO基础上，加入了这些实用的技巧，实现了检测速度和精度的最佳权衡。实验表明，在Tesla V100上，对MS COCO数据集的实时检测速度达到65 FPS，精度达到43.5%AP。

**YOLOv4的独到之处在于：**

1.是一个高效而强大的目标检测网络。它使我们每个人都可以使用 GTX 1080Ti 或 2080Ti 的GPU来训练一个超快速和精确的目标检测器。这对于买不起高性能显卡的我们来说，简直是个福音！

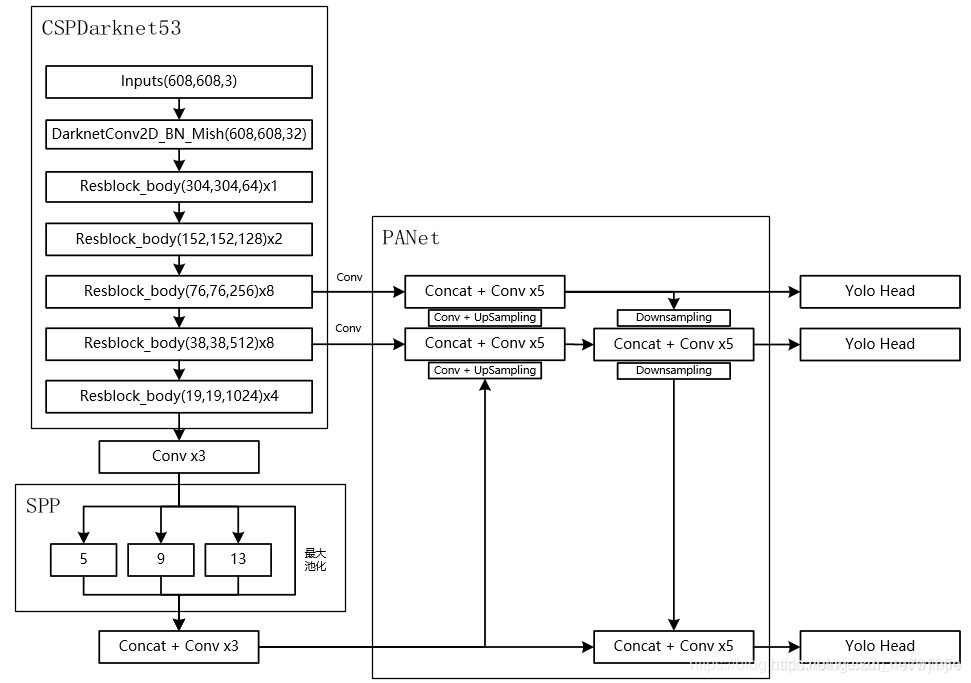
2.在论文中，验证了大量先进的技巧对目标检测性能的影响，真的是非常良心!

3.对当前先进的目标检测方法进行了改进，使之更有效，并且更适合在单GPU上训练；这些改进包括CBN、PAN、SAM等。

**4.2 网络结构**

**最简单清晰的表示： YOLOv4 = CSPDarknet53（主干） + SPP附加模块（颈） + PANet路径聚合（颈） + YOLOv3（头部）**

**完整的网络结构图如下：**

****

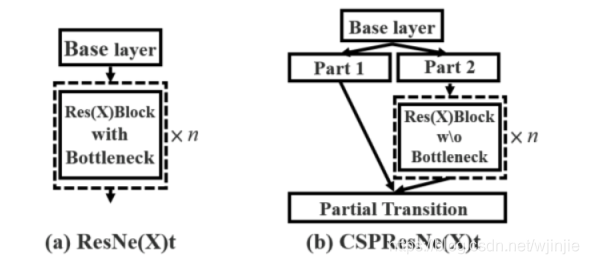
1. **CSPDarknet53**

CSPNet全称是Cross Stage Partial Network，在2019年由Chien-Yao Wang等人提出，用来解决以往网络结构需要大量推理计算的问题。**作者将问题归结于网络优化中的重复梯度信息。**CSPNet在ImageNet dataset和MS COCO数据集上有很好的测试效果，同时它易于实现，在ResNet、ResNext和DenseNet网络结构上都能通用。

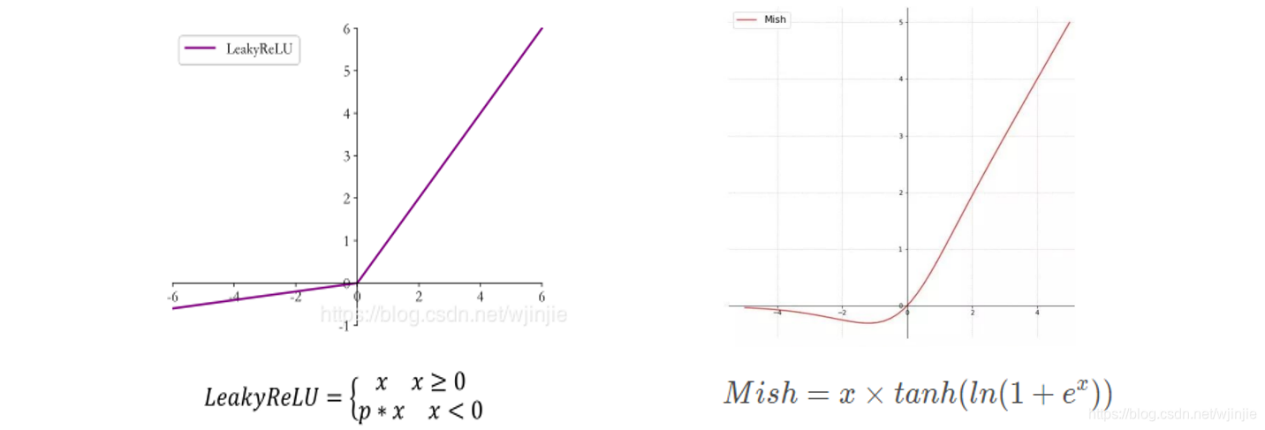
**CSPNet的主要目的是能够实现更丰富的梯度组合，同时减少计算量。**这个目标是通过将基本层的特征图分成两部分，然后通过一个跨阶段的层次结构合并它们来实现的。

而在YOLOv4中，将原来的Darknet53结构换为了CSPDarknet53，这在原来的基础上主要进行了两项改变：

**1.将原来的Darknet53与CSPNet进行结合。在**前面的YOLOv3中，我们已经了解了Darknet53的结构，它是由一系列残差结构组成。进行结合后，CSPnet的主要工作就是将原来的残差块的堆叠进行拆分，把它拆分成左右两部分：主干部分继续堆叠原来的残差块，支路部分则相当于一个残差边，经过少量处理直接连接到最后。具体结构如下：

****

**2.** **使用MIsh激活函数代替了原来的Leaky ReLU。**在YOLOv3中，每个卷积层之后包含一个批量归一化层和一个Leaky ReLU。而在YOLOv4的主干网络CSPDarknet53中，使用Mish代替了原来的Leaky ReLU。Leaky ReLU和Mish激活函数的公式与图像如下：



**3.CSPne减小计算量的方法**

举一个具体的例子来说明 CSPNet 是如何减少计算量的：

假设有一个输入特征图大小为 64x64x256，需要对其进行卷积操作，并使用 CSP 结构。在 CSP 结构中，特征图被一分为二，即变成两个 64x64x128 的特征图。然后，其中一个 64x64x128 的特征图上进行卷积操作，假设卷积核大小为 3x3，输出特征图大小仍为 64x64x128。另一个 64x64x128 的特征图直接进行跳跃连接，不进行额外的计算。最后，将卷积操作的结果与跳跃连接的结果进行拼接，得到一个 64x64x256 的输出特征图。

在这个例子中，通过 CSP 结构，只需要对其中一个 64x64x128 的特征图进行卷积操作，而另一个特征图直接进行跳跃连接，省去了一部分卷积操作的计算量，从而减少了总体的计算量。

**（2）SPP （Spatial Pyramid Pooling空间金字塔池化）**

SPP 指的是 Spatial Pyramid Pooling，是一种用于解决输入图像大小不一致的问题的技术。

在YOLOv4中，作者引入SPP，是因为它显著地增加了**感受野**，分离出了最重要的上下文特征，**并且几乎不会降低的YOLOv4运行速度**。如下图所示，就是SPP中经典的空间金字塔池化层。

**感受野（Receptive Field）**是指神经网络中某一层输出的特征图上的一个像素点，对应于输入图像中的区域大小。换句话说，感受野表示了该神经元（或特征图上的一个位置）对输入图像的感受范围大小。

**SPP 的关键思想是将图像划分成多个区域，并对每个区域分别进行池化操作，然后将所有区域的池化结果进行拼接，形成一个固定长度的特征向量。**这样一来，不同大小的图像都可以经过 SPP 池化层得到相同长度的特征向量，从而可以输入到后续的网络中进行处理。

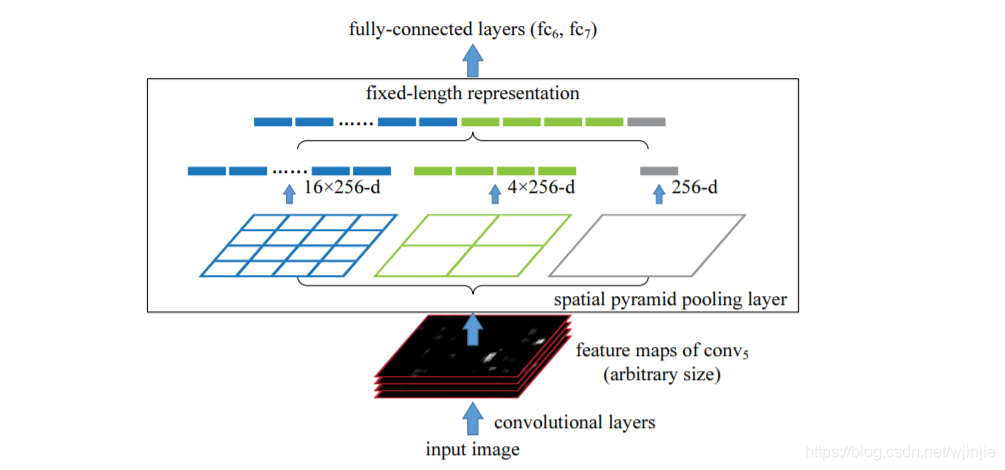
**SPP 主要由以下几个步骤组成：**

1.将图像划分成多个区域： 根据需要对图像进行划分，可以使用不同尺寸和比例的区域，以适应不同大小和比例的输入图像。

2.在每个区域上进行池化操作： 对每个区域分别进行池化操作，通常使用最大池化来提取该区域的主要特征。

3.拼接所有区域的池化结果： 将所有区域的池化结果进行拼接，形成一个固定长度的特征向量。

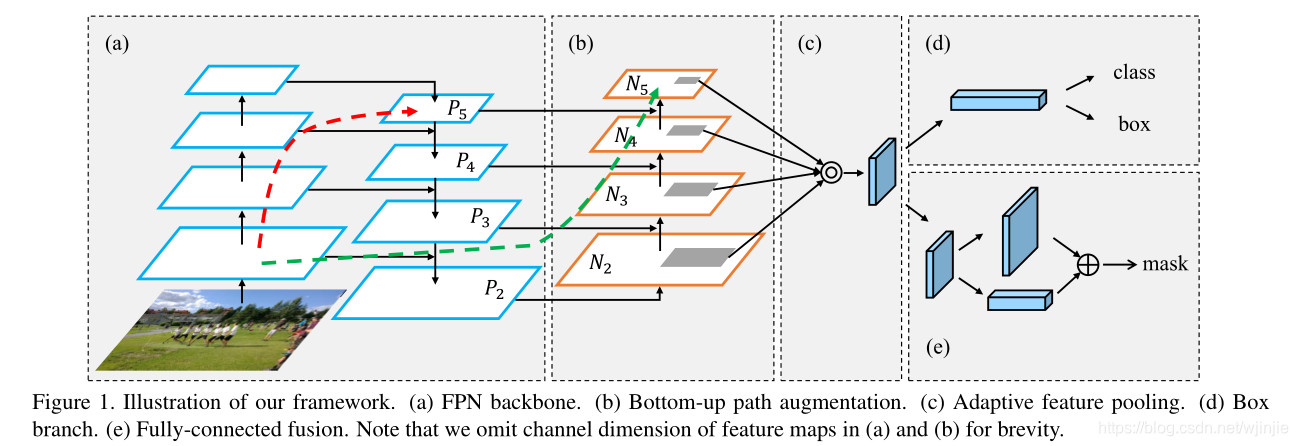
在YOLOv4中，具体的做法就是：分别利用四个不同尺度的最大池化对上层输出的feature map进行处理。最大池化的池化核大小分别为13x13、9x9、5x5、1x1，其中1x1就相当于不处理。



**（3）PANet**

PANet整体上可以看做是在Mask R-CNN上做多处改进，充分利用了特征融合，比如引入Bottom-up path augmentation结构，充分利用网络浅特征进行分割；引入Adaptive feature pooling使得提取到的ROI特征更加丰富；引入Fully-conneFcted fusion，通过融合一个前背景二分类支路的输出得到更加精确的分割结果。

下图是PANet的示意图，主要包含FPN、Bottom-up path augmentation、Adaptive feature pooling、Fully-connected fusion四个部分**。**

****

**FPN**发表于CVPR2017，主要是通过融合高低层特征提升目标检测的效果，尤其可以提高小尺寸目标的检测效果。

**Bottom-up Path Augmentation**的引入主要是考虑网络浅层特征信息对于实例分割非常重要，因为浅层特征一般是边缘形状等特征。

**Adaptive Feature Pooling**用来特征融合。也就是用每个ROI提取不同层的特征来做融合，这对于提升模型效果显然是有利无害。

**Fully-connected Fusion**是针对原有的分割支路（FCN）引入一个前背景二分类的全连接支路，通过融合这两条支路的输出得到更加精确的分割结果。

**一些解释：**

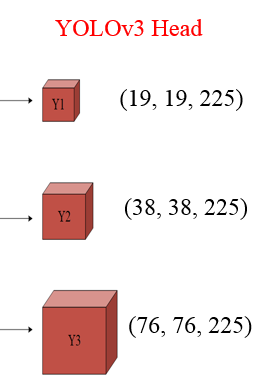
**原有的分割支路（FCN）：** 这是一个传统的语义分割网络，用于从输入图像中生成分割掩码。通常使用卷积神经网络（CNN）来实现，通过多次卷积和上采样操作来逐像素地预测每个像素点的类别。

**前背景二分类的全连接支路：** 这是一个新引入的支路，其作用是对每个像素点进行前景和背景的二分类。这个支路通常由全连接层组成，将网络的高层特征映射到前景和背景的概率空间，然后使用交叉熵等损失函数进行训练。

在YOLOv4中，作者使用PANet代替YOLOv3中的FPN作为参数聚合的方法，针对不同的检测器级别从不同的主干层进行参数聚合。并且对原PANet方法进行了修改, 使用张量连接(concat)代替了原来的捷径连接(shortcut connection)。

**（4）YOLOv3 Head**

在YOLOv4中，继承了YOLOv3的Head进行多尺度预测，提高了对不同size目标的检测性能。YOLOv3的完整结构在上文已经详细介绍，下面我们截取了YOLOv3的Head进行分析：

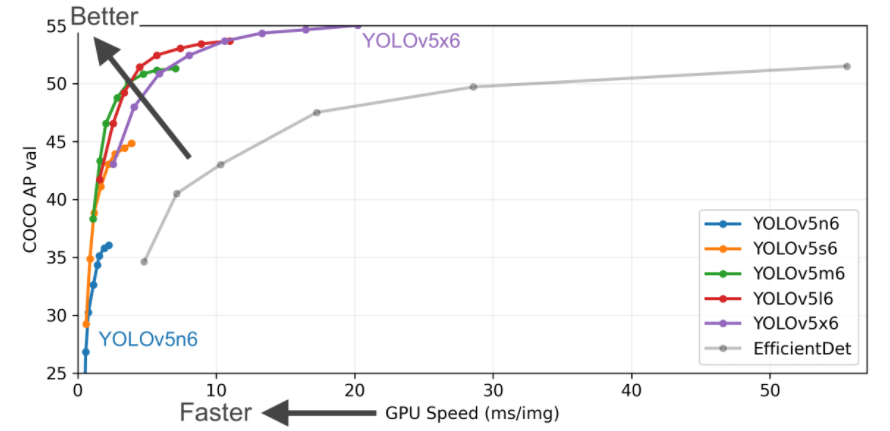


**YOLOv4学习了YOLOv3的方式，采用三个不同层级的特征图进行融合，并且继承了YOLOv3的Head。**从上图可以看出，在COCO数据集上训练时，YOLOv4的3个输出张量的shape分别是：**（19，19，225）、（38，38，255）、（76，76，225）。这是因为COCO有80个类别，并且每一个网格对应3个Anchor boxes，而每个要预测的bounding box对应的5个值**（ , , ，）所以有：3 x (80+5)=255。

**五、终局之战：YOLOv5**

**5.1 简介**

YOLOv5是YOLO系列的一个延申，您也可以看作是基于YOLOv3、YOLOv4的改进作品。YOLOv5没有相应的论文说明，但是作者在Github上积极地开放源代码，通过对源码分析，我们也能很快地了解YOLOv5的网络架构和工作原理。



**5.2 网络结构**

YOLOv5官方代码中，一共给出了5个版本，分别是 YOLOv5n、YOLOv5s、YOLOv5m、YOLOv5l、YOLO5x 五个模型。这些不同的变体使得YOLOv5能很好的在精度和速度中权衡，方便用户选择。

**本文中，我们以较为常用的YOLOv5s进行介绍，下面是YOLOv5s的整体网络结构示意图：**



**1、Input**

和YOLOv4一样，对输入的图像进行Mosaic数据增强。Mosaic数据增强的作者也是来自Yolov5团队的成员，通过**随机缩放、随机裁剪、随机排布的方式对不同图像进行拼接。**

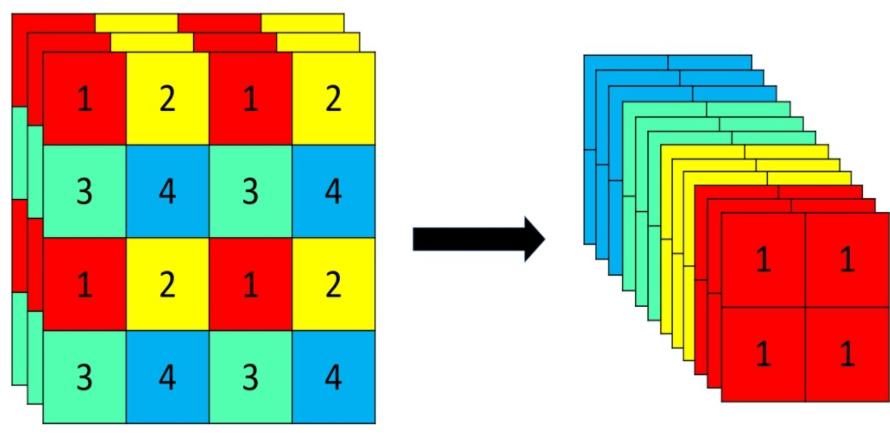
采用Mosaic数据增强方法，不仅使图片能丰富检测目标的背景，而且能够提高小目标的检测效果。并且在BN计算的时候一次性会处理四张图片！



**2、Backbone**

骨干网路部分主要采用的是：Focus结构、CSP结构。其中 Focus 结构在YOLOv1-YOLOv4中没有引入，作者将 Focus 结构引入了YOLOv5，用于直接处理输入的图片，通过降维和压缩输入特征图，从而减少计算量和提高感受野，同时提高目标检测的精度和模型的表达能力。

Focus重要的是切片操作，如下图所示，4x4x3的图像切片后变成2x2x12的特征图。



以YOLOv5s的结构为例，原始608x608x3的图像输入Focus结构，采用切片操作，先变成304x304x12的特征图，**再经过一次32个卷积核的卷积操作**，最终变成304x304x32的特征图。

备注：

**再经过一次32个卷积核的卷积操作**

假设输入特征图的大小为 304x304x12，经过一次 3x3 的卷积操作，使用了 32 个卷积核。在每个位置上，卷积核都与输入特征图的对应区域进行卷积操作，然后将所有卷积的结果在每个位置上进行求和，得到一个大小不变的输出特征图。

**3、Neck**

在网络的颈部，采用的是：**FPN+PAN结构**，进行丰富的特征融合，这一部分和YOLOv4的结构相同。

**FPN（Feature Pyramid Network）：** FPN 是一种用于目标检测任务的网络结构，旨在解决多尺度目标检测的问题。它通过自顶向下和自底向上的特征融合方式，生成了具有不同尺度信息的特征金字塔，使得模型能够在不同尺度下检测目标。

**PAN（Path Aggregation Network）：** PAN 是一种用于语义分割任务的网络结构，旨在解决多尺度语义分割的问题。它通过引入了横向连接和上下文特征融合模块，实现了不同尺度特征的融合和信息传递，从而提高了语义分割的性能。

FPN 和 PAN 结构可以相互结合，以进一步提高模型的性能。结合 FPN 和 PAN 可以实现多尺度特征的提取和融合，同时保持空间信息的一致性，从而在目标检测和语义分割任务中取得更好的效果。

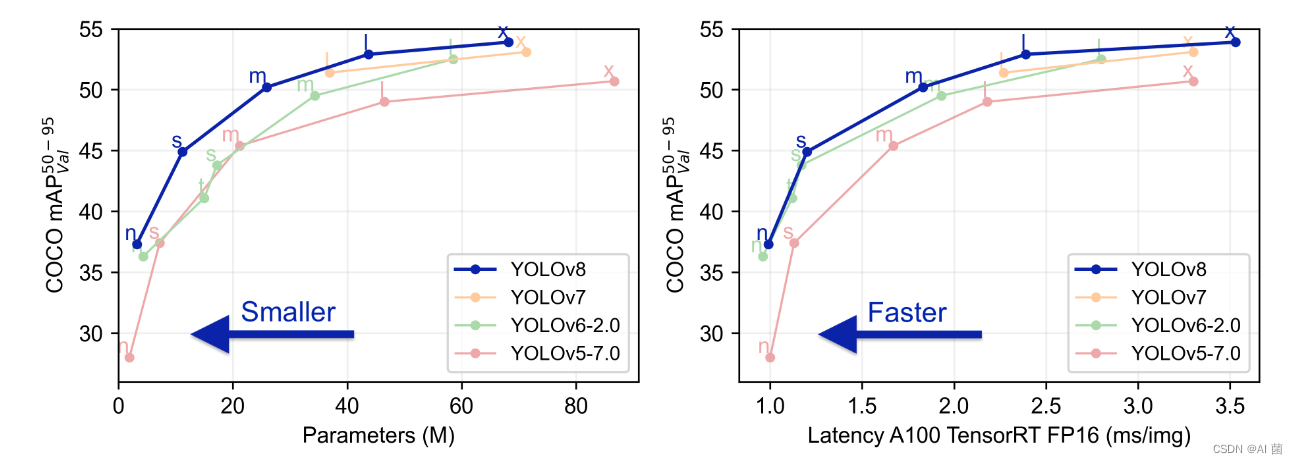
**4、Head**

对于网络的输出，遵循YOLO系列的一贯做法，采用的是耦合的Head。并且和YOLOv3、YOLOv4类似，采用了三个不同的输出Head，进行多尺度预测。

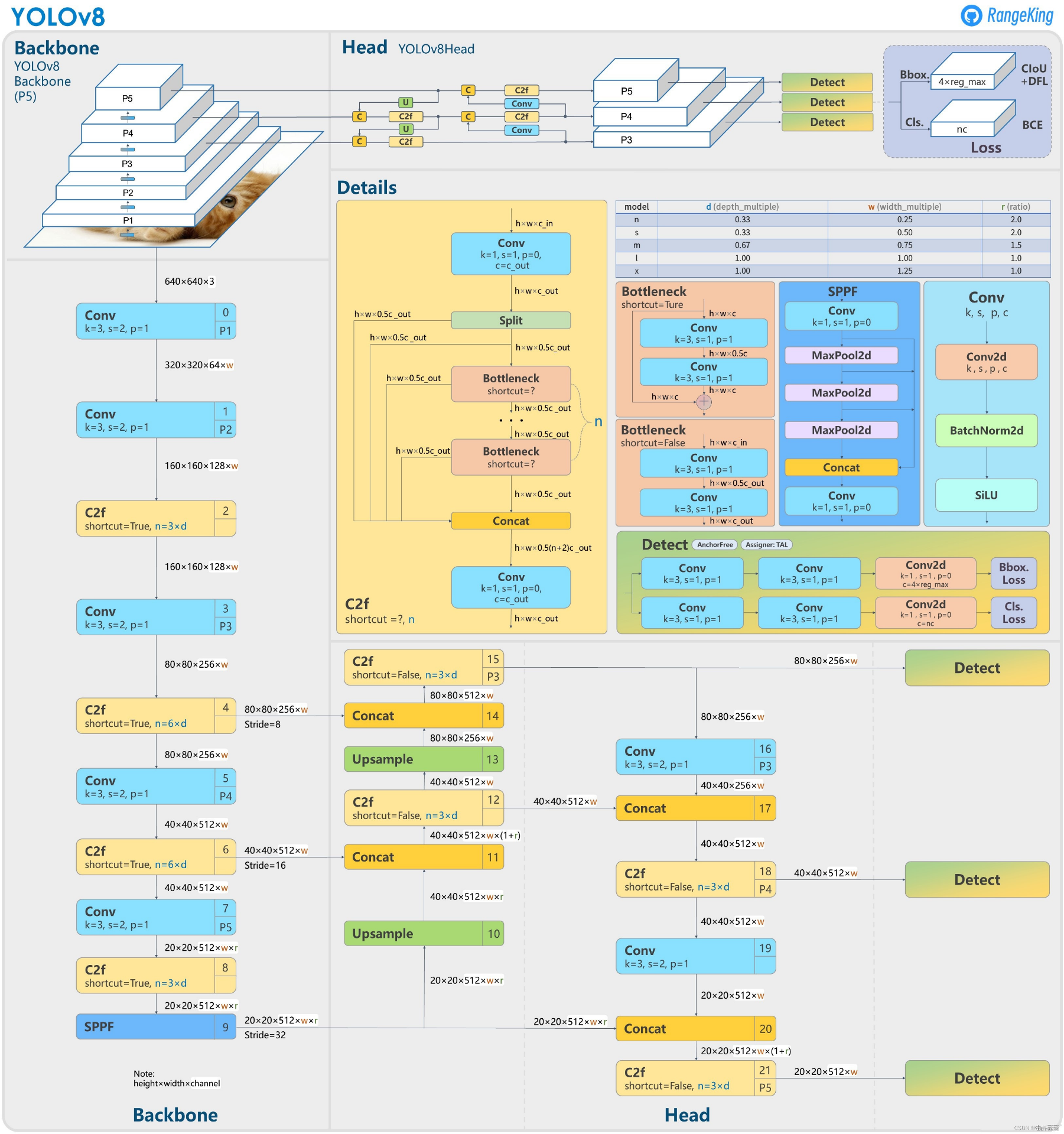
**六、梅开二度：YOLOv8**

**6.1 简介**

YOLOv8 与YOLOv5出自同一个团队，是一款前沿、最先进（SOTA）的模型，基于先前 YOLOv5版本的成功，引入了新功能和改进，进一步提升性能和灵活性。YOLOv8 设计快速、准确且易于使用，使其成为各种物体检测与跟踪、实例分割、图像分类和姿态估计任务的绝佳选择。



**6.2 网络结构**

****整体结构上与YOLOv5类似： **CSPDarknet（主干） + PAN-FPN（颈） + Decoupled-Head（输出头部），但是在各模块的细节上有一些改进，并且整体上是基于anchor-free的思想，这与yolov5也有着本质上的不同。**