**YOLO v1的原理及实现过程**

本文链接：<https://blog.csdn.net/weixin_42278173/article/details/81778217>

目标检测是一件比较实际的且具有挑战性的计算机视觉任务，其可以看成图像分类与定位的结合，给定一张图片，目标检测系统要能够识别出图片的目标并给出其位置，由于图片中目标数是不定的，且要给出目标的精确位置，目标检测相比分类任务更复杂。目标检测的一个实际应用场景就是无人驾驶，如果能够在无人车上装载一个有效的目标检测系统，那么无人车将和人一样有了眼睛，可以快速地检测出前面的行人与车辆，从而作出实时决策。

**本文的内容如下：**

* YOLO 与其他检测方法的区别
* 基本概念的介绍
* 网络结构
* LOSS函数
* 训练与测试

**YOLO 与其他检测方法的区别：**

YOLO的全称叫做“You Only Look Once”，简单来说，YOLO可以做到将一张图片输入，直接输出最终结果，包括框和框内物体的名称及score(得分)。而很多其他检测方法如Faster R-CNN便是通过两次检测来达到目的。具体的两次检测过程如下所述：

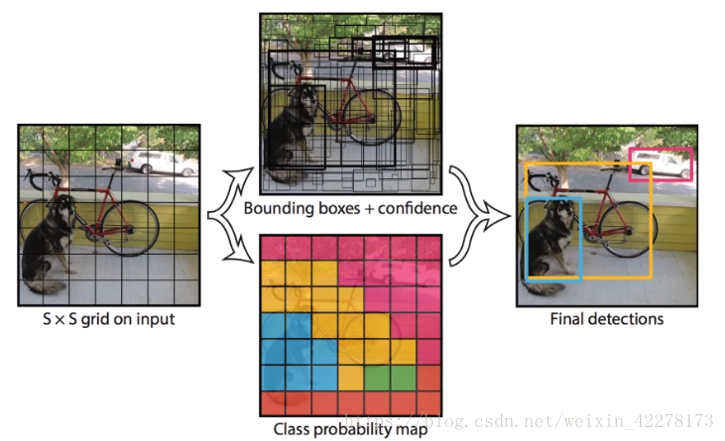
* Input Image经过CNN特征提取，首先来到Region Proposal Network 输出classification，第一次并不是判定物体在COCO数据集上对应的目标类的某一类，而是输出一个binary的值p,可以理解为p属于[0,1]，人工设定一个threshold=0.5。
* RPN网络的意义是如果一个Region的p大于等于threshold，那么就认为这个Region中可能是目标类别中的某一类，这些被选出的region又叫做ROI(Region of Interest)。这一步会同时框定出ROI感兴趣区域的大致位置，即输出Bounding Box。
* 然后将ROI输入到普通的分类网络，即可得到整个网络最终的输出classification,这里的class真正对应了COCO数据集里80个类别的具体类别之一。在这一步之后，还要对bbox进行尺寸的微调，输出第一张图右上方的Bounding-box regression。

因此，Region Proposal的作用为：

1. 避免影响网络的分类性能，如果不进行第一步的筛选，anchor框定的region会有许多类似于蓝天，草地这种背景类别，对于物体分类没有益处。
2. 减少参数，增加网络分类性能。

YOLO的优势是：不需要region proposal，所谓的region proposal即需要进行多于一次的识别过程。以Faster R-CNN为代表的目标检测的方法就是两次检测的过程。

**基本概念的介绍**



1. 首先将原图像分成S x S grid，在介绍yolo的论文里S=7，即将图像分为7 x 7 grid。
2. 每个grid cell都会生成B个 Bounding Box 去进行物体的框定和分类，B在论文里给的是2，因此在此例中有98个Bounding Box。
3. Confidence包含两方面，边界框**含有目标的可能性**以及**边界框的准确度**。
4. 利用**NMS (非极大值抑制) 算法**得到最后的final 的detections。
5. 对于每一个单元格要给出C个类别概率值，其表征的是由该单元格负责预测的边界框其目标属于各个类别的概率。这些概率值其实是在各个边界框置信度下的条件概率，即：

这里写图片描述

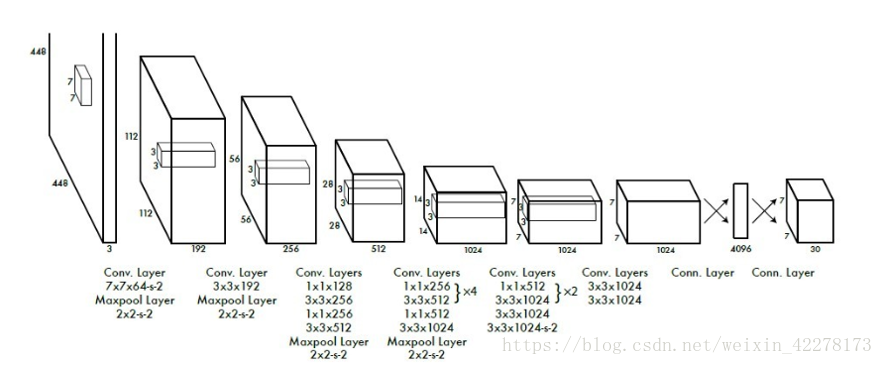
其中[http://latex.codecogs.com/gif.latex?Pr(object)*IOU_%7bpred%7d%5e%7btruth%7d](http://www.codecogs.com/eqnedit.php?latex=Pr(object)*IOU_%7bpred%7d%5e%7btruth%7d)为**上述3.**中的confidence，而[http://latex.codecogs.com/gif.latex?IOU_%7bpred%7d%5e%7btruth%7d](http://www.codecogs.com/eqnedit.php?latex=IOU_%7bpred%7d%5e%7btruth%7d)是指的预测框和实际框（Ground Truth）的交并比。

这边我们提到了边界框，边界框的大小和位置用4个值来表示：(x,y,w,h)，其中(x, y)是边界框的中心坐标，w和h是边界框的宽与高。具体来说，(x, y)是相对于每个单元格左上角坐标点偏移值，并且单位是相对于单元格大小的，而w和h预测值是相对于整个图片的宽和高的比例，这样理论上4个元素的大小在[0, 1]范围内，而且每个边界框的预测值实际上包含5个元素：(x, y, w, h, c)。前四个元素表征边界框的大小和位置，最后一个值是置信度。

总结一下：每个单元格需要预测(B\*5+C)个值。输入的图片为S x S网络，因此最终的预测值为S x S x (B x 5 + C)大小的张量，在论文里给出的例子为S=7，B=2， 因此最终的张量大小为7 x 7 x 30。

**网络结构**

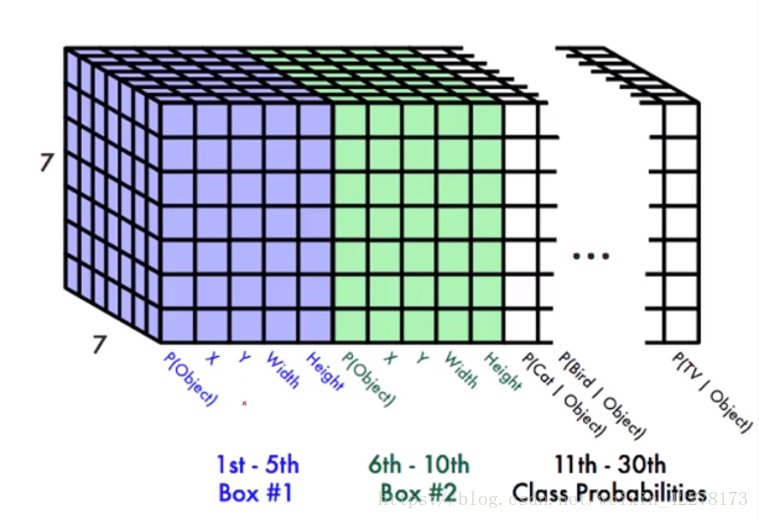
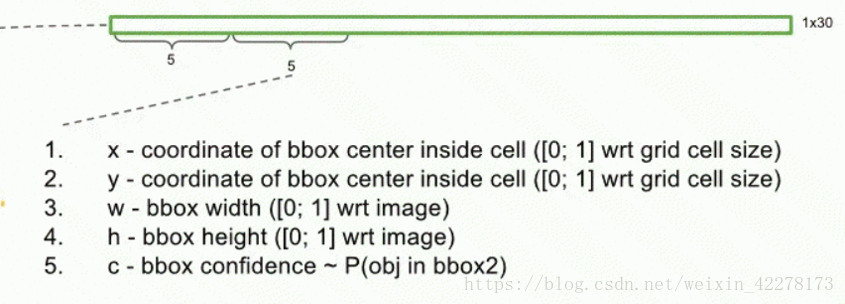
Yolo采用卷积网络来提取特征，然后使用全连接层来得到预测值。网络结构参考GooLeNet模型（2014年Christian Szegedy提出的一种全新的深度学习结构）。YOLO检测层包含24个卷积层和2个全连接层。



（这里一开始我数不出来为什么有24个卷积层，其实要仔细看图，你会发现有一些“x4” “x2”，把这些考虑进去你就会发现确实是24层）

最后得到的7 x 7 x 30 代表的是最后的输出，代表一共49个cell，每个cell拥有30个值，其中有20个值为类别概率值，即该cell检测出来的属于某类物体的概率。而剩下的10个值可以分成两部分，分表代表cell两个Bounding Box各自的参数部分。

我们取一个cell来看，一个cell有30个元素，如图所示：

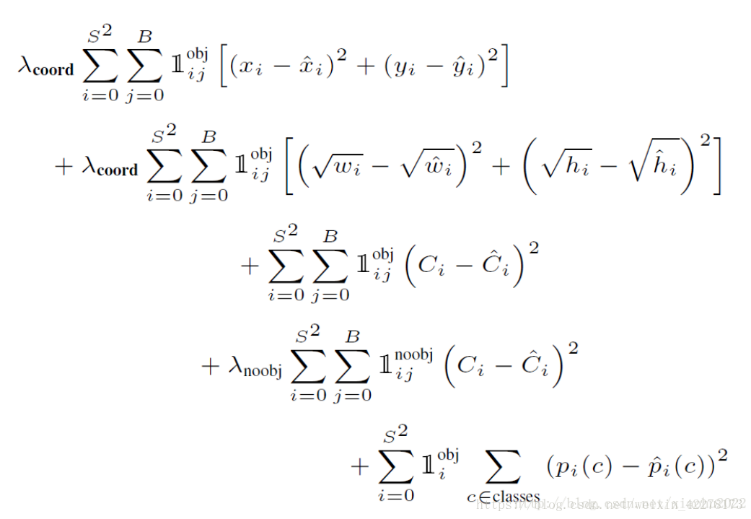
这里写图片描述

如果在grid cell里没有物体存在，则Pr(object)=0，存在的意思是指物体的ground truth中心点在这个cell里面。另外我们发现，一个grid cell里面虽然有两个Bounding Box, 但是它们共享同一组分类概率，因此同一个cell只能识别同一个物体。

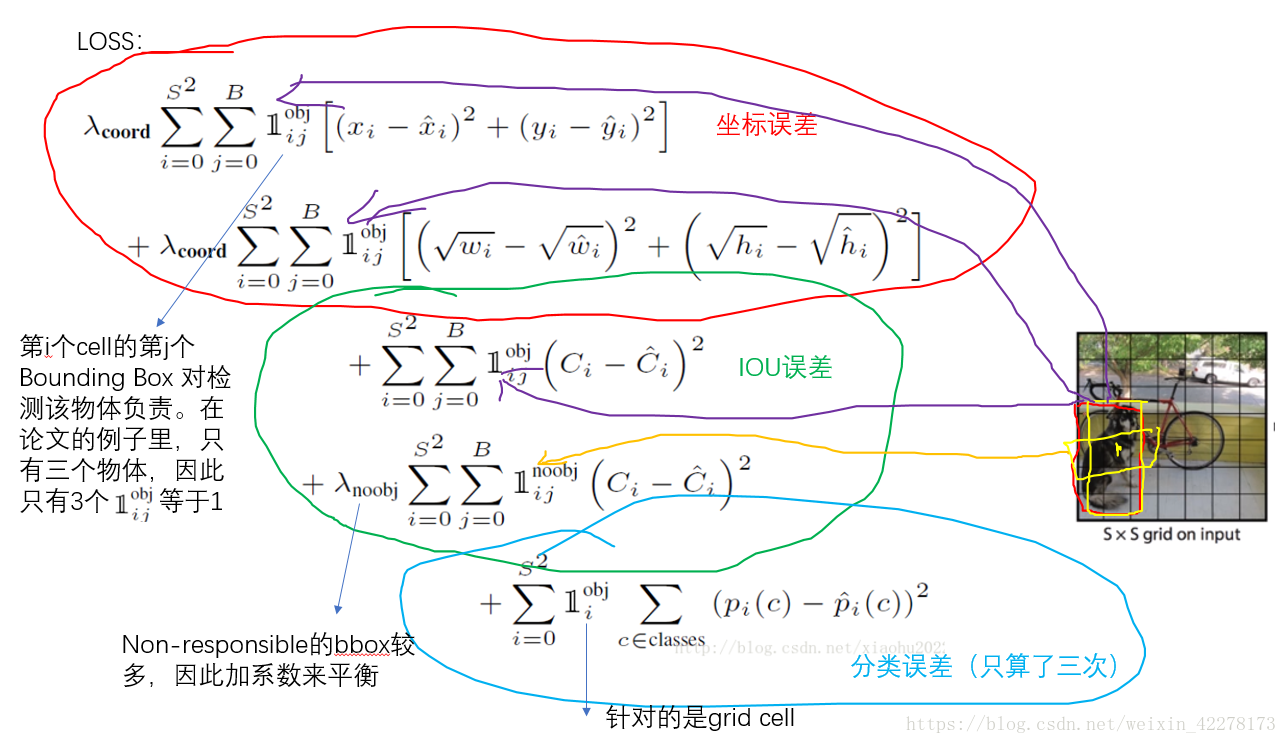
边界框的大小和位置用4个值来表示：（x, y, w, h), 其中（x，y）是边界框的中心坐标，w和h是边界框的宽与高。(x, y)是相对于每个单元格左上角坐标点偏移值，并且单位是相对于单元格大小的，而w和h预测值是相对于整个图片的宽和高的比例，这样理论上4个元素的大小在[0, 1]范围内，而且每个边界框的预测值实际上包含5个元素：(x, y, w, h, c)。前四个元素表征边界框的大小和位置，最后一个值是置信度。

**LOSS函数**

LOSS函数和其他神经网络一样，是优化的目标，在yolo v1里LOSS函数相当复杂，大体可以分为三部分，下面具体介绍一下LOSS函数：



为了便于说明，我们将各部分用不同颜色圈起来：



1. 红色圈代表的是坐标误差，极好理解。这里要注意，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bbox预测中，相比于大bbox预测偏一点，小box预测偏一点更不能忍受。而sum-square error loss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box的width和height取平方根代替原本的height和width。
2. 绿色圈我们命名为IOU误差，主要是针对confidence的误差，由于confidence的计算方式（如上所述），主要受到IOU的影响，因此我们叫做IOU误差。
3. 蓝色圈称为分类误差，与bounding box无关，只和cell有关，是针对每个cell做概率的差值计算。
4. 这里写图片描述符号代表第i个cell的第j个Bounding Box 对检测该物体负责。在论文的例子里，只有三个物体，因此只有3个该项为1，凡是不负责的均为这里写图片描述

除此之外，针对各项的系数，还有一些tips：

1. 损失函数的设计目标就是让坐标（x,y,w,h），confidence，classification 这个三个方面达到很好的平衡。
2. 8维的localization error和20维的classification error同等重要显然是不合理的。更重视8维的坐标预测，给这些损失前面赋予更大的loss weight。
3. 如果一个网格中没有object（一幅图中这种网格很多），那么就会将这些网格中的box的confidence push到0，相比于较少的有object的网格，这种做法是会导致网络不稳定甚至发散。
4. 对没有object的bbox的confidence loss，赋予小的loss weight。

**训练的最终结果是得到一个网络，网络参数是根据不停地优化LOSS函数而得到的。**

**训练与测试：**

训练：训练集的图片都进行各个物体的人工标注（4个顶点的坐标），通过运算，得到x y w h，然后图片被分成S x S个小格，针对每个小格有b个BBOX，然后经过优化LOSS，不停地使得BBOX逼近Ground Truth。

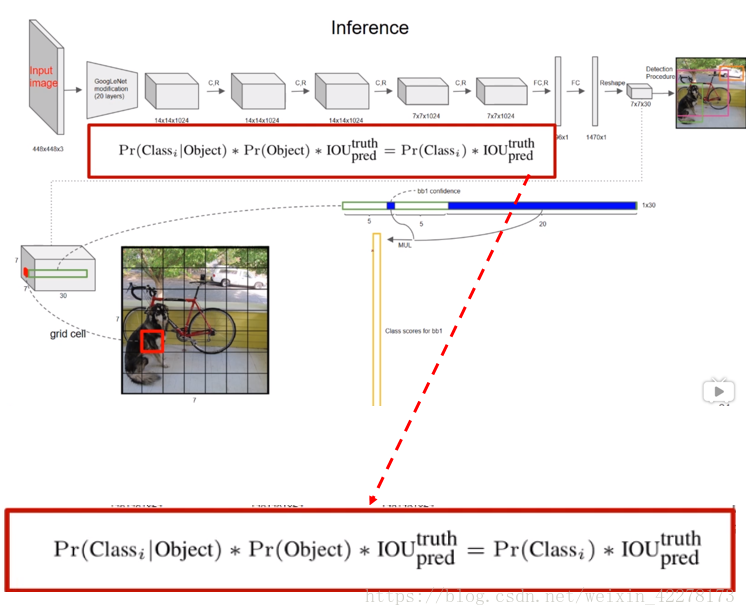
测试：利用训练好的网络，输入图片，直接给出box和对应的物体。

**训练过程：**

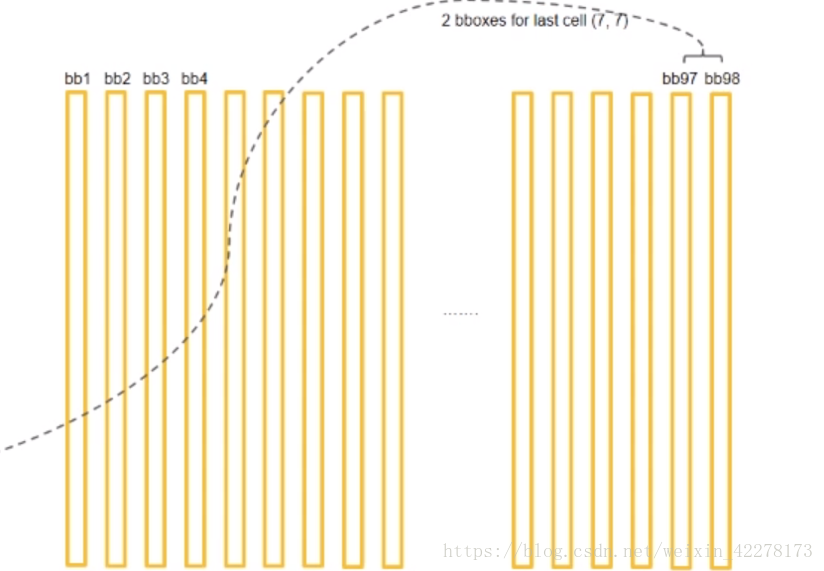
1. 预训练分类网络：在 ImageNet 1000-class competition dataset上预训练一个分类网络，这个网络是Figure3中的前20个卷机网络+average-pooling layer+ fully connected layer （此时网络输入是224\*224）。
2. 训练检测网络：预训练之后，在预训练得到的20层卷积层之上加上随机初始化的4个卷积层和2个全连接层。由于检测任务一般需要更高清的图片，所以将网络的输入从224x224增加到了448x448。

**测试过程：**

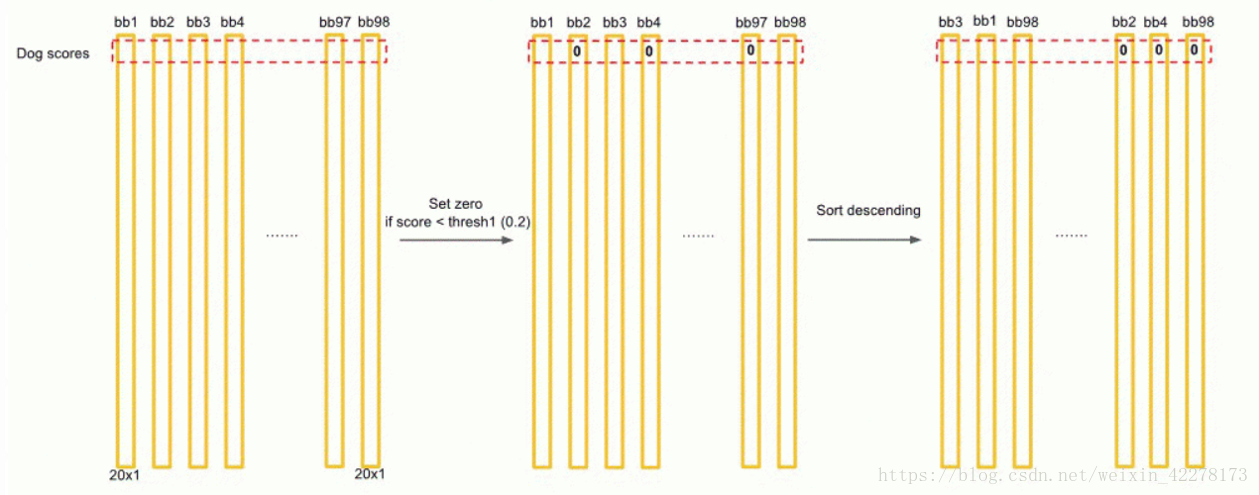
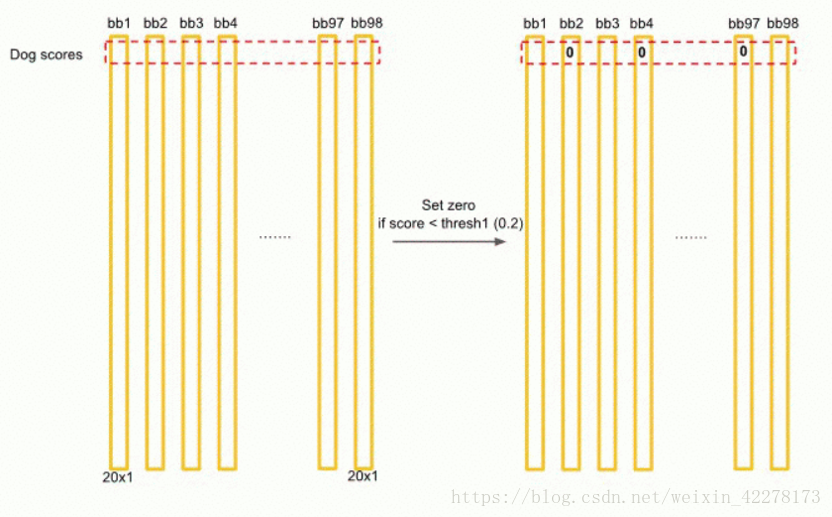
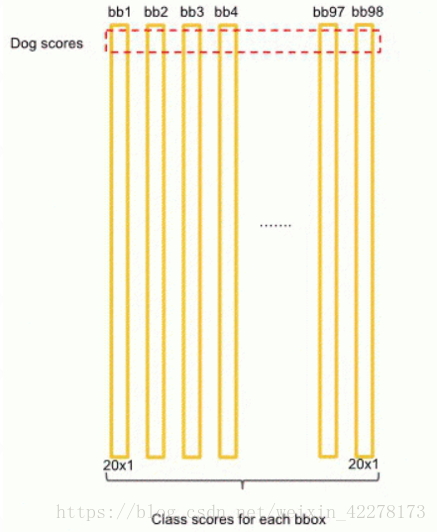
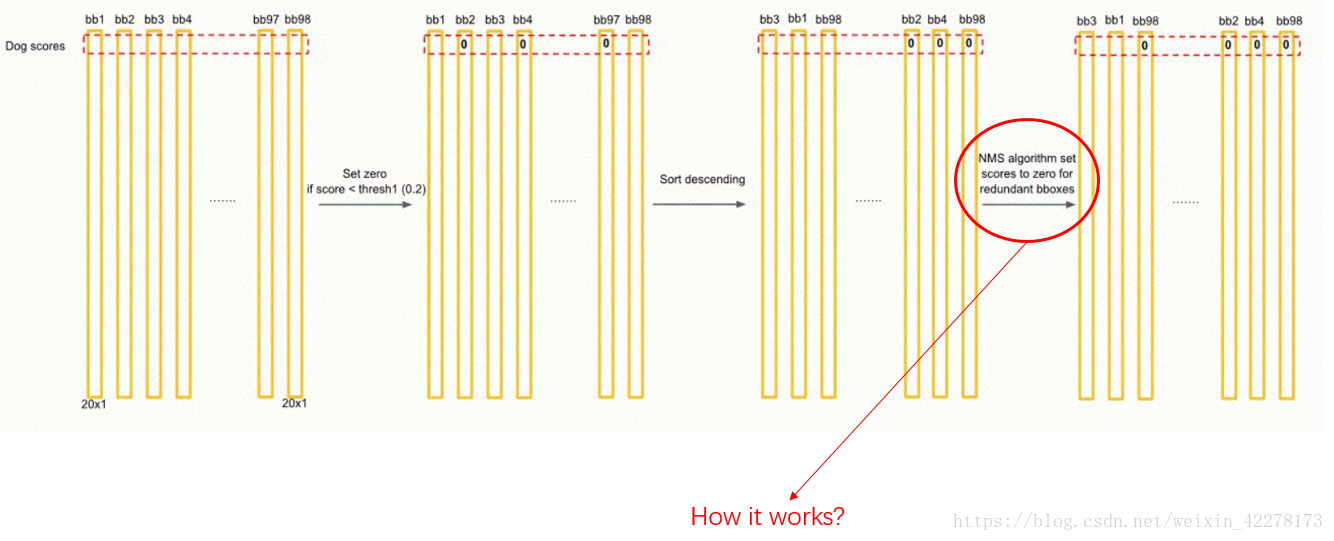
输入图片，网络会按照与训练时相同的分割方式将测试图片分割成S x S的形状，因此，划分出来的每个网格预测的class信息和Bounding box预测的confidence信息相乘，就得到了每个Bounding box的class-specific confidence score，即得到了每个Bounding box预测具体物体的概率和位置重叠的概率。



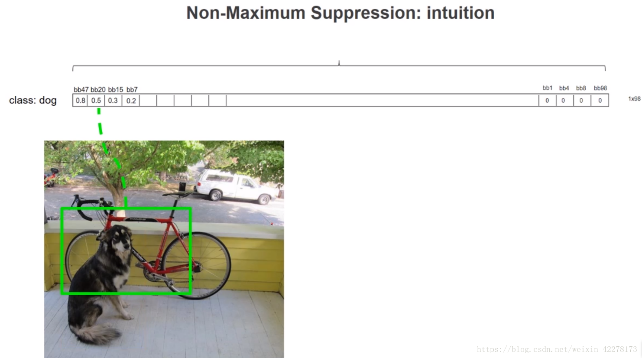
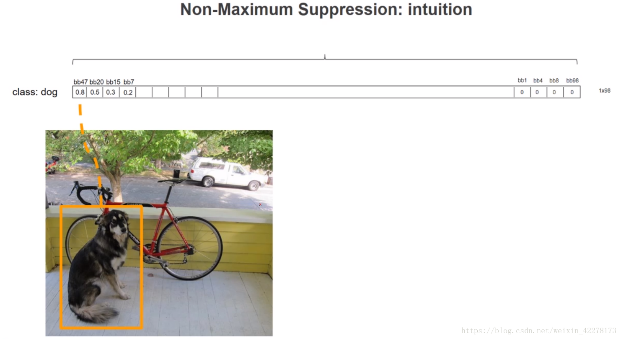
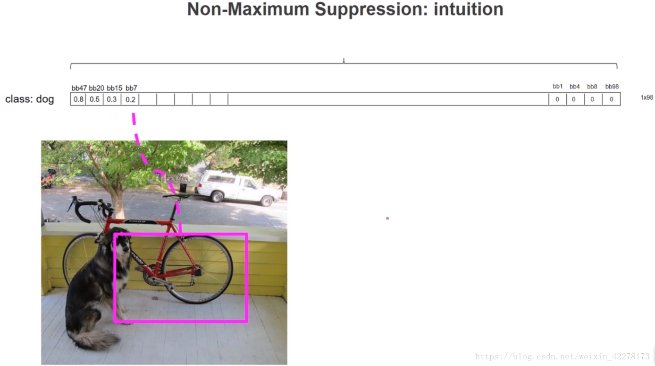
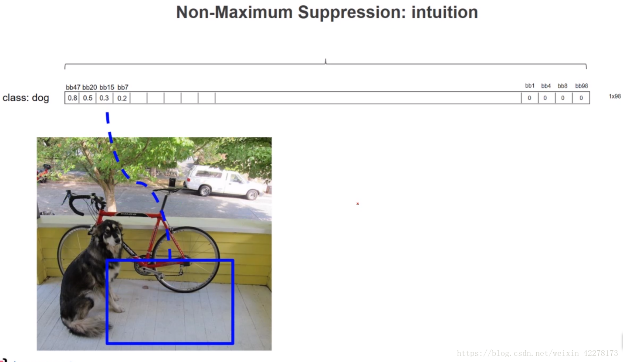
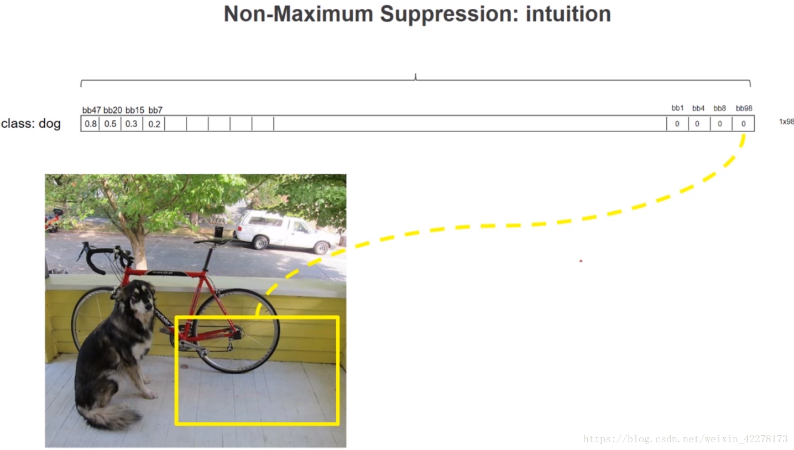
对于98个Bounding Box都这么运算，最后可以得到：



每个“条”一共有20个值，分别是20个物体的得分，因此一共有98\*20个值，我们按照类别把它们分为20类。之后的过程如下（先以第一类假设为“dog”举例）：

直观来感受一下非极大值抑制的过程：

非极大值抑制：抑制的过程是一个迭代-遍历-消除的过程。

1.将所有框的得分排序，选中最高分及其对应的框。

2.遍历其余的框，如果和当前最高分框的重叠面积(IOU)大于一定阈值，我们就将框删除。

3.从未处理的框中继续选一个得分最高的，重复上述过程。（具体可见其他博主整理，或者bilibili yolov1讲解）

针对其他类别，比如cat，vehicle等，均采用这种方式，直到将所有的score值得到。然后，针对每一个Bounding Box，查看其最大的值，如果在20个值里面最大值非0，那么我们就把这个bounding box认定为是识别对应物体的框。

**当然，yolo v1有相当的缺陷：**

1. YOLO对相互靠的很近的物体，还有很小的群体 检测效果不好，这是因为一个网格中只预测了两个框，并且只属于一类。
2. 对测试图像中，同一类物体出现的新的不常见的长宽比和其他情况是。泛化能力偏弱。