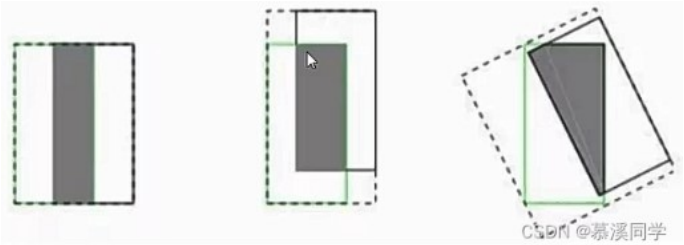
评价指标 IoU, Precision, Recall, mAP

IoU (Intersection over Union)

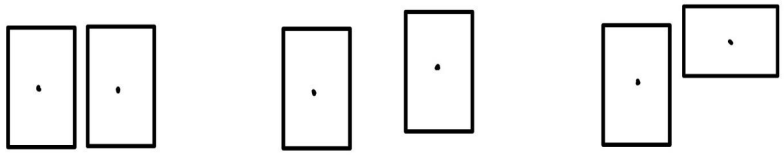
IoU = area of overlap / area of union

局限性：

1. if overlap = 0，IoU = 0，无法进行梯度计算
2. 相同的 IoU 反映不出实际检测框与真实框之间的情况。如下图，虽然三组框的 IoU 值相等，但是检测框与真实框之间的相对位置却完全不一样



3. 再或者，由于三组框都没有相交，他们的 IOU 值相等且都为 0，这种情况 IOU 也不能很好的评价检测框与真实框的位置关系。



TP FP TN FN

T 或者 F 代表的是该样本 识别结果是否正确

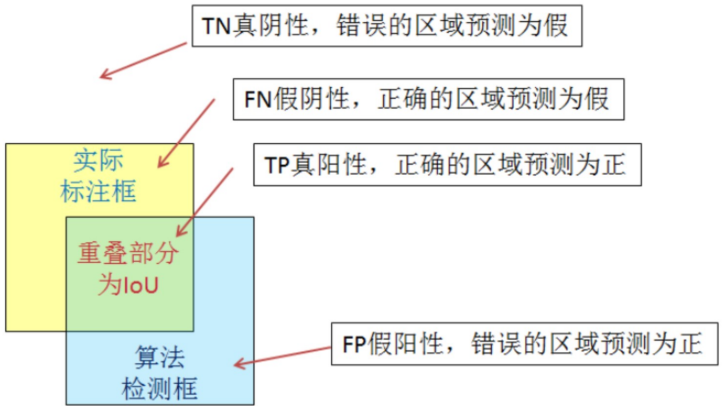
P 或者 N 代表的是该样本 被识别成了正样本还是负样本

TP: 被模型识别为正样本，实际为正样本

FP: 被模型识别为正样本，实际为负样本

TN: 被模型识别为负样本，实际为负样本

FN: 被模型识别为负样本，实际为正样本



FP → 误检，把背景认成物体
FN → 漏检，把物体认为背景
在 YOLO 中往往不关注 TN（被模型预测为负类的负样本）

①在 YOLO 中 Positive 是物体，Negative 是背景，TN 表示背景确实是背景，且因为 label 只有正样本，没有负样本

②因为负样本（背景）数量极大，在损失函数计算当中只会给予很小的比重

实际部署时，无法实际获取 TN。其次 FN 也很难获取，漏检只有人工才能知道。

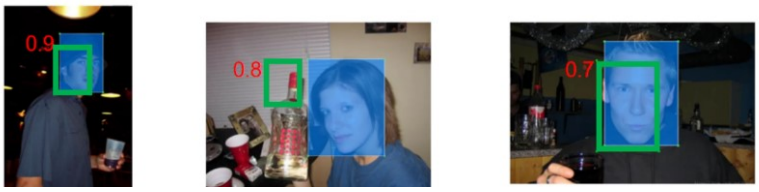
精度 Precision = TP / (TP + FP) 识别正确的结果占所有识别结果的比例

召回率 Recall = TP / (TP + FN) 识别正确的结果占实际正样本的比例

高 precision -> FP 少 -> 误检少 -> 高阈值 -> 模型严格（宁缺勿滥）
高 recall -> FN 少 -> 漏检少 -> 低阈值 -> 模型宽松（能检尽检）

在 yolo 检测中，高于置信度阈值->正样本，低于置信度阈值->负样本

置信度（即只有IOU大于阈值时，检测框才有效）
基于置信度阈值计算精度（Precision）和召回率（Recall）。下图蓝色表示真实框，绿色表示预测框。
当阈值=0.9，只有第一张图有效，其余两张为漏检。TP=1、FP=0、FN=2。**Precision** $=\frac{1}{1+0}=\frac{1}{1}$；**Recall** $=\frac{1}{1+2}=\frac{1}{3}$；



AP & mAP

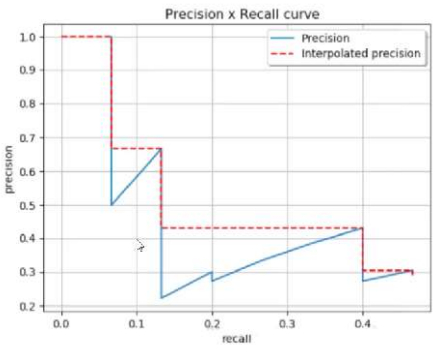
P-R 曲线图：取不同的置信度阈值，可以获得不同的 Precision 和不同的 Recall。以 Recall 为横轴，Precision 为纵轴，并将 Precision[0, 1]范围内的每个点对应 recall 的值连接起来形成一条折线。如左图的蓝色曲线

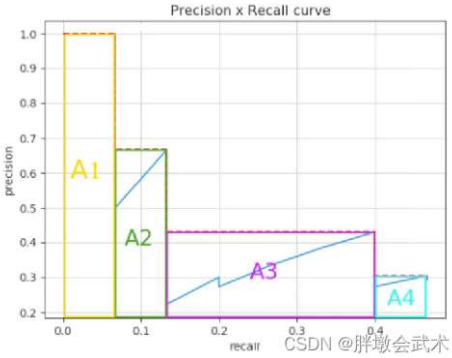
AP（Average Precison）：

（1）以 y = precision 为平行线，从上向下，取 P-R 曲线图在每个 y 值对应的最大值，然后连接形成阶梯状曲线。如左图的红色曲线

（2）阶梯曲线与 x 轴围成的面积之和就是 AP 值（A1+A2+A3+A4），如右图

mAP（mean Average Precision）：对所有类别对应的 AP 值求和取均值，就得到整个数据集上的 mAP。mAP 综合衡量了 Precision 和 Recall，理论上最优值为 1。



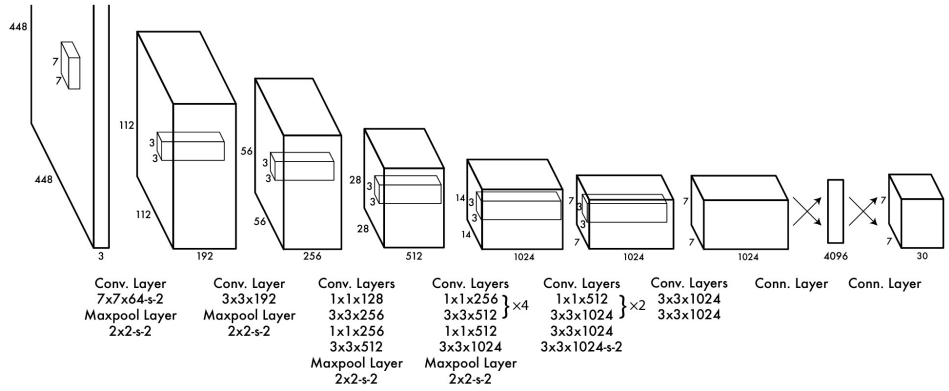


CSDN @胖墩会武术

YOLO v1

1. Network Design 网络结构

Network：24 Conv. Layers, 4 Pooling Layers, 2 Conn. Layers
Input: 448 x 448 x 3 的图像
Output: 7 x 7 x 30 的张量
Activation Function: 最后一层使用 Linear activation，其他每层都使用 leaky ReLU



Notes on network:
① “s-2”表示 stride 为 2，没有写的为 1；默认 Same Padding
② Same Padding 值为 P = (F - 1) / 2
③ 计算下一层的尺寸公式为 N = (W – F + 2P) / S + 1
其中，N 为 output，W 为 input，F 为 filter，S 为 stride，结合②可简化为 N = (W-1)/S+1，对于 S=2 的情况 N=W/2+1/2，向下取整就是 N=W/2
④ 通道数由上一层 Kernel 的通道数决定
⑤ Pooling 不改变通道数，且没有 padding
⑥ 最后一层 Conv 后尺寸为 7x7x1024，如何变成 4096？先将 7x7x1024 展平成一维向量，先连接 256 再连接 4096（减少运算量）
⑦ 4096 如何变成 7x7x30？将 4096 连接到 1470，然后 reshape 成 7x7x30（仅仅是将 1470 个元素按顺序填充到一个新的三维张量中）

Q. 为什么是 4 个 Pool Layer 而不是 6 个?
第一层和倒数第三层卷积层的步长不是 1 是 2

Q. 两个 FC 层的作用?
第一个：(Flatten)把该输入图像的所有卷积特征整合到一起 (1x4096)
第二个：进行维度转换，最后得到与**目标检测网络输出维度**相同的维度（7x7x30）

Q. FC 层有什么弊端，为什么 YOLO v2 以后就不再使用?
①参数爆炸，仅一个 1x4096 的 FC 层参数量就达到了 10^8 级
②Flatten 展平操作会丢失位置信息，“破坏特征的空间结构信息”

Q. Output 的 7 * 7 * 30 的张量，代表什么?
输入图像被划分成 7x7 的网格（Grid），每个网格单元（Grid Cell）输出 **30** 个数值

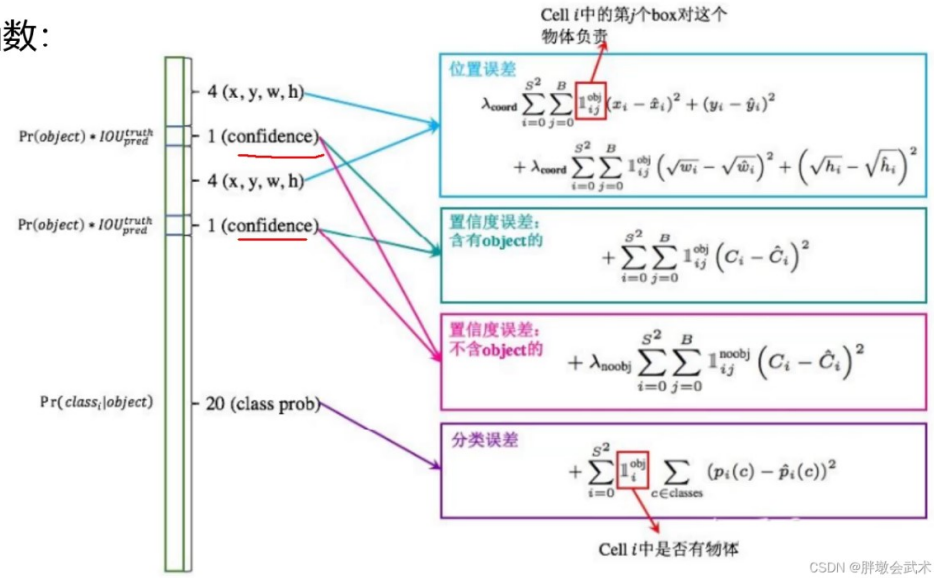
30 = 2 * (4 + 1) + 20
① **2**: 每个网格输出 2 个 bboxes
② **4**: 输出的位置信息（x_c, y_c, w, h）和 **1**: Confidence 置信度
③ Confidence = P (Object) * IOU (truth, predict)，P (Object)指的是该检测框内存在目标（不限类别）的概率。反映了目标存在的可能性，以及该框与实际目标的匹配程度。
④ x_c, y_c 表示 bbox 的中心坐标相较于该 bbox 归属的 grid cell 左上角的偏移量；w, h 表示 bbox 的宽高与原始图像的宽和高的比值；均归一化到（0，1）
⑤ **20**: 20 个类别概率（该 Grid Cell 属于各个类别的可能性）

7×7=49 个 gird cell，最多只能检测 49 个物体。每一个 grid cell 只能检测一个类，因此 YOLO 对多而小的物体性能比较差。

2. Train 训练阶段

Loss Function

- 损失函数：



till YOLOv7 still **these 3 losses (Localization loss, Objectness loss, Classification loss)**

Pre-training

使用 **224x224** 的 ImageNet 图像训练分类网络。网络结构为 YOLO v1 网络的前 20 个卷积层+一个临时的平均池化层（Global Average Pooling, GAP）+ 一个临时的全连接层（用于 ImageNet 1000 分类）。

Fine-tuning

使用 **448x448** 的目标检测数据集训练 YOLO v1 网络

Q. Why pre-training using 224*224?

1. ImageNet 数据集的特点 2.图片尺寸小，batch_size 可以更大。At that time, it is believed that larger batch size achieves better model performance by more stable training.

Q. **Pre-train 和 Fine-tune 阶段的图像输入尺寸不同有什么影响？**
①模型在从训练分类任务过渡到训练检测任务时需要适应分辨率的变换，在 v2 改进

3. Infer 推理阶段

1. Resize image to 448*448. 2. Run model. 3. Non-Maximum Suppression.

Q. There are FC layers in YOLOv1, why changing input size (from 224 to 448) do not need changes in architecture in YOLOv1?
要区分 pre-training 和 fine-tuning 这两个阶段。pre-training 阶段分类任务使用 224x224，而 fine-tune 阶段检测任务使用 448x448，两个训练阶段网络结构不同，FC 层不同。

x	0.25	0.75	0.50	-
y	0.60	0.30	0.50	-
w	0.34	0.78	0.41	-
h	0.57	0.50	0.17	-
Objectness	1	1	1	0
Dog	1	0	0	0
Bicycle	0	1	0	0
Car	0	0	1	0
Desk	0	0	0	0
...				

NMS (Non-Maximum Suppression) 非极大值抑制
概念：NMS 算法主要解决的是一个目标被多次检测的问题
1. 对 20 类第一类计算 7*7*2=98 个全概率
2. 设置 threshold，将低于阈值的全概率置为 0，余下按降序排列
3. 将后 97 个 bbox_cur 与 bbox_max 对比，将 IOU 大于 0.5 的置为 0
4. 第一个 bbox_max 全部比较完后，再将次极大和其他对比
5. 对第一个类进行 NMS 操作后处理剩余 19 类
6. 得到稀疏矩阵(20*98)，将非 0 的 bbox 画在原图上
7. 对于同一个 bbox，取全概率最大的类别作为检测结果

Q. NMS 只发生在推理阶段，训练阶段是不能用 NMS 的，为什么？
在训练阶段，模型还在学习边界框和类别信息，不需要用 NMS。训练时需要对所有候选框都进行梯度更新，删除候选框会导致梯度信息缺失，影响模型的学习效果。训练的目标是让模型自身学会减少重叠检测，从而在推理时减少对 NMS 的依赖。

Reference

[1] 三万字硬核详解：yolov1、yolov2、yolov3、yolov4、yolov5、yolov7-CSDN 博客
[2] 1.2 YOLO 入门教程：YOLOv1(2)-浅析 YOLOv1 - 知乎