YOLO 系列综述-Part 1：

YOLOv1：单阶段目标检测的革命性突破

目录

1. 简介

实时物体检测已经成为众多应用中的一个重要组成部分，横跨自主车辆、机器人、视频监控和增强现实等各个领域。在各种物体检测算法中，YOLO（You Only Look Once）框架因其在速度和准确性方面的显著平衡而脱颖而出，能够快速、可靠地识别图像中的物体。自成立以来，YOLO 系列已经经历了多次迭代，每次都是在以前的版本基础上解决局限性并提高性能（见表格 1）。

表格 1：历代 YOLO 发布时间、主要改动及作者，由笔者同学、四川大学硕士贺宇劼整理。其相关文章链接：
https://blog.csdn.net/qq_54478153/article/details/139477271

时间	版本	主要改动	作者
2015	v1		Joseph Redmon & Ali Farhadi
2016	v2	引入Batch Normalization 和锚框(anchor boxes)	Joseph Redmon & Ali Farhadi
2018	v3	更高效的骨干网络、引入特征金字塔和PAN	Joseph Redmon & Ali Farhadi
2020	v4	引入SPP模块、CSP模块、Mish激活函数、CmBN归一化	Alexey Bochkovskiy
2020	v5	引入Focus结构	Ultralytics
2021	x	引入解耦头 (Decoupled Head)、不再预设锚框	旷视
2022	v6	引入RepConv，量化相关内容，简化解耦头	美团
2022	v7	Efficient Layer Aggregation Network (ELAN)模块，MP模块，Rep	台湾中央研究院
	v8	C2f模块	Ultralytics
2024	v9	Generalized Efficient Layer Aggregation Network (GELAN)	台湾中央研究院
2024	v10	整体高效的网络设计、空间-通道解耦下采样	清华

笔者旨在通过撰写一系列文章，回顾 YOLO 系列框架的发展，从最初的 YOLOv1 到最新的 YOLOv10，阐释每个版本的关键创新、差异和改进。除了讨论每个 YOLO 版本的具体进展外，该系列还强调了在整个框架的发展过程中出现的速度和准确性之间的权衡问题。分析了在选择最合适的 YOLO 模型时，应当考虑具体应用的背景和要求的重要性。

本文为该系列的第一章，将对 YOLOv1 进行分析。我们首先探讨了原始 YOLO 模型的基本概念和架构，这为 YOLO 系列的后续进展奠定了基础。随后，我们深入探讨了 YOLOv1 的细节设定，如网络设计、损失函数、锚框的调整。通过分析这些基础设定，方便在后续研究中，对 YOLO 框架的演变及其对物体检测的影响有一个基本的理解。

文章撰写时间短，未能仔细校对。可能存在笔误、描述不准确、重要内容缺失等问题。希望大家能指出，笔者会随时修改补充。

注：该系列文章中，所有加粗并配有下划线的关键词，均配有交叉引用。

曹倬瑄

2024/6/14 于惠州

2. YOLOv1：基本概念与架构

2.1. 引言&背景

在 YOLOv1 问世之前，CNN 系列算法在目标检测领域占据主导地位。尽管 R-CNN 系列在检测精度上表现出色，但其基于**双阶段（two-stage）**的网络结构限制了检测速度，难以满足实时性需求，这在学术界和工业界都引起了广泛的讨论和批评。为了解决这一挑战，设计一种既快速又准确的目标检测器成为研究的热点。YOLOv1 的提出，标志着实时对象检测技术的重大进步。它通过**单阶段（One-Stage）**神经网络直接从图像预测**边界框（bounding box）**和**类别概率**，不仅极大提升了检测速度，还保持了较高的准确性。

YOLOv1 的核心创新在于将对象检测重新定义为一个回归问题，而非传统的分类问题。这种设计允许 YOLOv1 在单次**模型前向过程（Model Forward Process）**中，同时预测图像中所有对象的位置和类别。具体来说，YOLOv1 将整个图像作为输入，通过单一的神经网络直接输出边界框的位置和类别，实现了检测过程的简化和加速。

YOLOv1 的提出，不仅在理论上提供了一种新的视角，更在实践中推动了实时对象检测技术的发展。它的设计哲学和实现方法，为后续的对象检测算法提供了宝贵的参考和启示。

2.2. 架构概述

YOLOv1 的架构由以下几个关键部分组成：

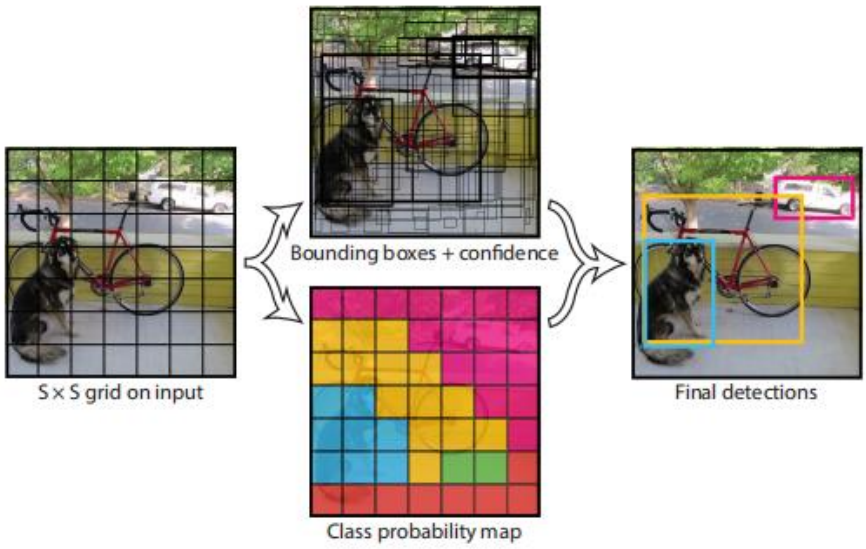
- **图像分割**：YOLOv1 将输入图像划分为一个 S×S（在模型代码中 S=7）的网格。每个网格单元负责预测中心点落在该网格内的对象。
- **边界框预测**：每个网格单元预测 B 个边界框，每个边界框由(x, y, w, h)组成，其中(x, y)表示边界框中心相对于网格左上角的位置，w 和 h 分别表示边界框的宽度和高度。
- **置信度评分**：YOLOv1 为每个边界框预测一个置信度，该置信度反映了模型对边界框内包含对象的信心以及预测框与真实框的一致性。
- **类别概率**：每个网格单元还预测 C 个条件类别概率，表示在该网格单元内存在对象的条件下，各个类别的概率。
- **损失函数**：YOLOv1 使用自定义的损失函数进行训练，该函数考虑了边界框坐标的误差、置信度的误差以及类别概率的误差。
- **网络结构**：YOLOv1 的网络结构受到 GoogLeNet 的启发，包含 24 个卷积层和 2 个全连接层，以及特别设计的 1×1 卷积层来减少特征维度。
- **预训练与微调**：YOLOv1 的卷积层首先在 ImageNet 数据集上进行预训练，然后在 PASCAL VOC 数据集上进行微调，以学习特定于对象检测的特征。

3. YOLOv1 核心技术点分析

3.1. 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。在 Faster R-CNN 中，是通过一个区域提议网络（Region Proposal Network, 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 S×S 个网格，这些网格就相当于是目标的感兴趣区域。通过这种方式，我们就不需要再额外设计一个 RPN 网络。**图片 1** 展示了其建模过程。



图片 1：检测建模过程

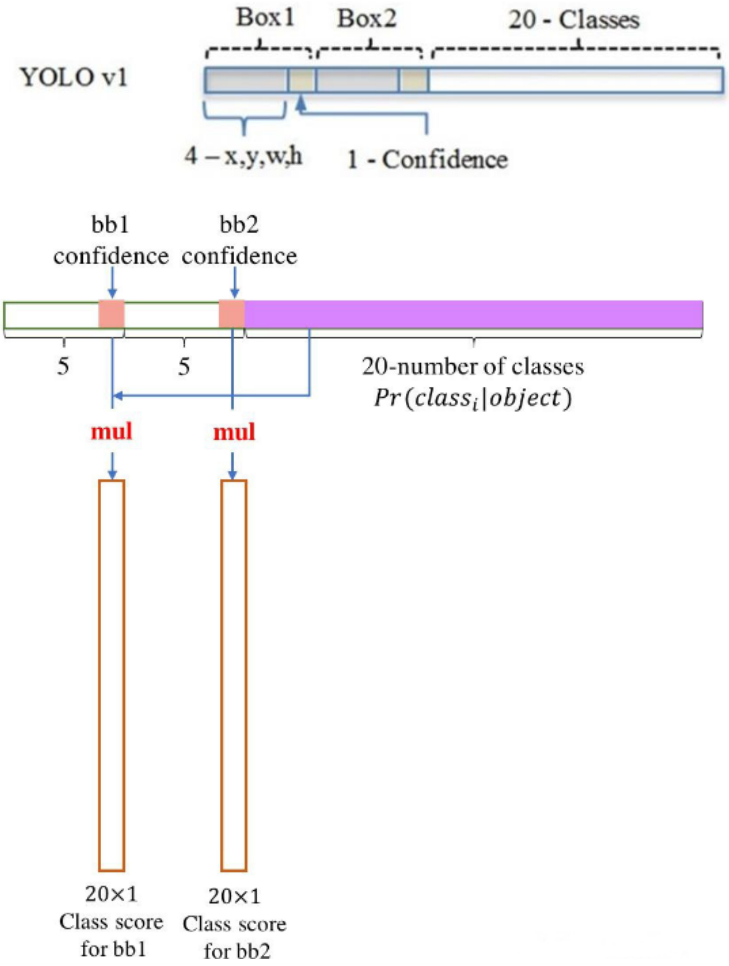
具体实现过程如下：

- 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。
- 每个网格要预测 B 个边界框，每个边界框要预测 (x, y, w, h) 和 confidence 共 5 个值。
  - (x, y)是边界框中心相对于网格左上角的位置，该坐标是通过将边界框的实际中心位置标准化到[0, 1]范围内来实现的，这样它们就可以表示为相对于网格大小的比例值。
  - w 和 h 是边界框的宽度和高度，这两个值也是相对于整个图像的尺寸进行标准化的，这样它们也是在[0, 1]范围内的值。
  - confidence（置信度）是模型对这个边界框包含对象的信心。是一个介于 0 和 1 之间的值，表示模型对预测框包含对象的信心。它通过以下公式计算：

$$Confidence=Pr(Object)\times IOU_{truth}^{pred}$$

其中，Pr(Object)是网格中存在对象的概率， $predIOU_{truth\ pred}$ 是预测边界框与真实边界框之间的交并比。

- 每个网格单元还预测 C 个条件类别概率，表示在该网格单元内存在对象的情况下，对象属于每个类别的概率。这些概率是通过 Softmax 函数（附：机器学习常见激活函数）得到的。
- 总的来说，S×S 个网格，每个网格要预测 B 个边界框，还要预测 C 个类。网络输出就是一个 S × S × (5×B+C) 的张量。**图片 2** 中示了具体的边界框结构。



图片 2：YOLOv1 边界框结构

3.2. 网络结构

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。

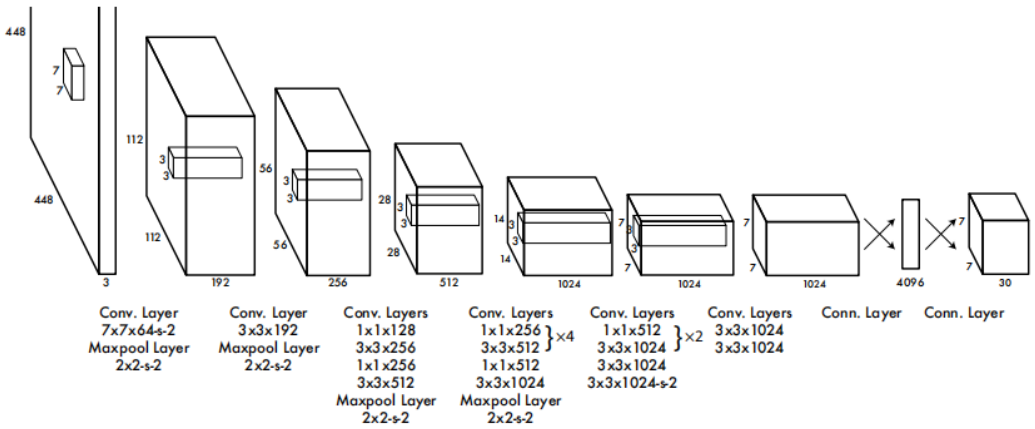
YOLOv1 架构包括 24 个卷积层，然后是两个全连接层（FC）。预测边界框坐标和概率。除了最后一个层使用线性激活函数外，所有层都使用了漏整流线性单元（Leaky ReLU 函数）激活。

其网络输入为 448×448×3 的彩色图片，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。一些卷积层交替使用 3 × 3 卷积与 1 × 1 卷积相结合以减少通道。对于最后一个卷积层，它输出一个形状为 (7, 7, 1024) 的张量。然后张量被展平。使用 2 个全连接层作为线性回

归的一种形式，它输出 7×7×30 参数，然后 reshape 为 (7, 7, 30)，即每个位置 2 个边界框。**表格 2、图片 3** 展示了其卷积结构

表格 2：YOLOv1 卷积结构分析表

	Type	Filters	Size/Stride	Output
	Conv	64	7 × 7 / 2	224 × 224
	Max Pool		2 × 2 / 2	112 × 112
	Conv	192	3 × 3 / 1	112 × 112
	Max Pool		2 × 2 / 2	56 × 56
1×	Conv	128	1 × 1 / 1	56 × 56
	Conv	256	3 × 3 / 1	56 × 56
	Conv	256	1 × 1 / 1	56 × 56
	Conv	512	3 × 3 / 1	56 × 56
	Max Pool		2 × 2 / 2	28 × 28
4×	Conv	256	1 × 1 / 1	28 × 28
	Conv	512	3 × 3 / 1	28 × 28
	Conv	512	1 × 1 / 1	28 × 28
	Conv	1024	3 × 3 / 1	28 × 28
	Max Pool		2 × 2 / 2	14 × 14
2×	Conv	512	1 × 1 / 1	14 × 14
	Conv	1024	3 × 3 / 1	14 × 14
	Conv	1024	3 × 3 / 1	14 × 14
	Conv	1024	3 × 3 / 2	7 × 7
	Conv	1024	3 × 3 / 1	7 × 7
	Conv	1024	3 × 3 / 1	7 × 7
	FC		4096	4096
	Dropout 0.5			4096
	FC		7 × 7 × 30	7 × 7 × 30

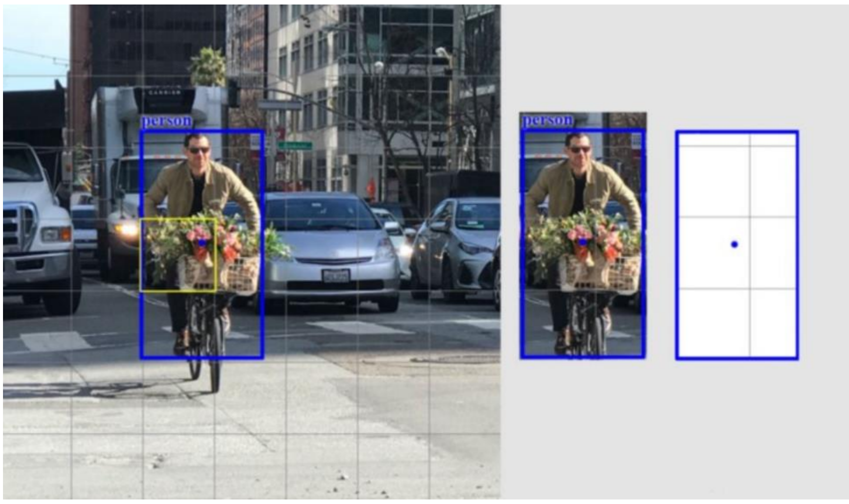


图片 3：YOLOv1 网络结构图

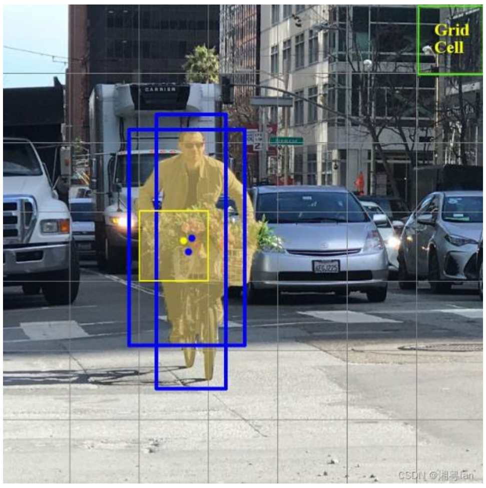
3.1. 网络细节

对于预测的 B 个边界框，每个框都有一个置信度（confidence）分数，每个网格单元只检测一个对象，并预测物体在 C 个类别的概率（物体属于每一种类别的可能性）。并使用**非极大值抑制（NMS）**避免重复检测。

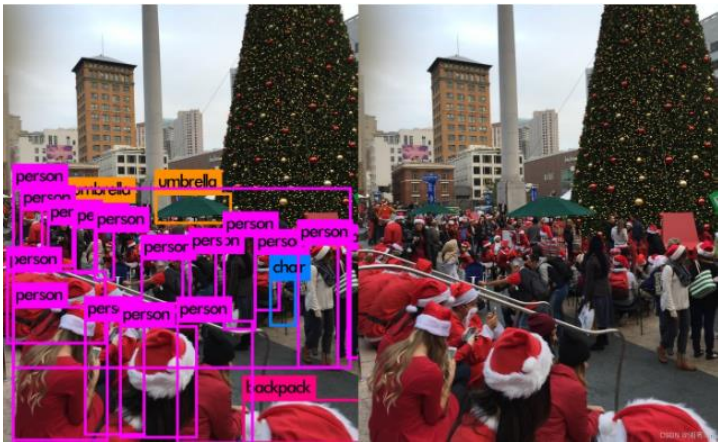
详细流程呈现于图片 4、图片 5、图片 6。



图片 4：YOLOv1 输入图像划分为 7× 7 格。每个网格单元仅预测一个目的。例如，上面的黄色网格单元尝试预测蓝色边界框中心（蓝点）位于网格单元内的“人”对象。



图片 5：每个网格单元预测固定数量 B 的边界框，在这个例子中，黄色网格单元进行两个边界框预测（蓝色框）来定位人的位置（B = 2）。



图片 6：单对象规则限制了检测到的对象的接近程度。为此，YOLOv1 对临近的物体检测有一些限制。例如，本图中，左下角有 9 个圣诞老人，但 YOLO 只能检测到 5 个

3.2. 目标损失函数

YOLOv1 预测每个网格单元的 2 个边界框。为了计算 TP（True positive）的损失，作者只希望其中一个边界框负责检测物体。为此，作者选择与 ground truth 具有最高 IoU 的那个边界框。这样便可以使得边界框预测的更加精确。并且预测出来的某些尺寸和纵横比都会变得更好。

YOLO 用预测（predictions）和 ground truth 之间的平方和误差（sum-squared error）来计算损失。损失函数包括：分类预测损失、坐标预测损失（预测的 bounding box 和 ground truth 之间的差距），置信度预测损失（框的客观真实性）。

3.2.1. 类别预测损失

公式 1：类别预测计算公式

$$\sum_{i=0}^{S^2} \mathbb{1}_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

此公式判断是否有 object 中心落在网格中，如果检测到一个物体，则该单元格的分类损失是每个类的类条件概率的平方差（**公式 1**）。

3.2.2. 坐标预测损失

公式 2：坐标预测损失公式

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$
$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

坐标预测损失可算出预测边界框位置和大小的误差（只计算负责检测物体的边界框）（公式 2）。

使用差方和误差。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边界框预测中，相比于大边界框预测偏一点，小边界框预测偏一点更不能忍受。作者不想对大的边界框和小的边界框中的绝对误差进行同等赋权。

为了缓和这个问题，作者用了一个比较取巧的办法，就是将边界框的 w 和 h 取平方根代替原本的 w 和 h。YOLO 实际上预测的是的宽度（w）和高度（h）的平方根，而不是宽度和高度。

因为坐标误差比分类误差大，为了增加对坐标误差为了提高边界框的准确性，作者将损失乘以 λ_{coord}（默认值：5）。

3.2.3. 置信度预测损失

如果在 box 中检测到物体，则置信度预测损失为：公式 3

公式 3：置信度预测损失公式-有物体

$$\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{\text{obj}}\left(C_i-\hat{C}_i\right)^2$$

如果在 box 中没有检测到物体，则置信度预测损失为：公式 4

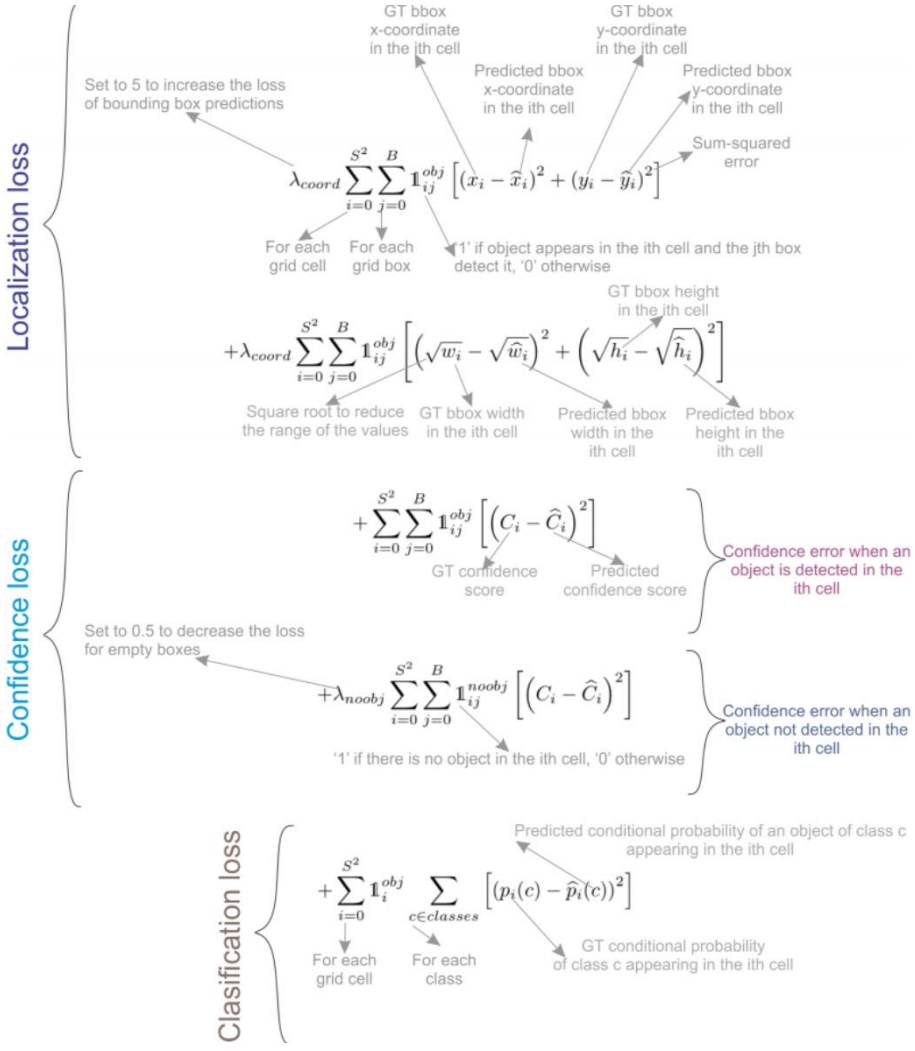
公式 4：置信度预测损失公式-无物体

$$\lambda_{\text{noobj}}\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{\text{noobj}}\left(C_i-\hat{C}_i\right)^2$$

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λ_{noobj}=0.5。

最终，将以上所有公式相加，我们获得了完整的损失函数（公式 5）。

公式 5：完整损失函数公式以及注释



4. YOLOv1 优缺点分析

4.1. 优点分析

YOLOv1 检测速度非常快。标准版本的 YOLOv1 可以每秒处理 45 张图像；YOLOv1 的极速版本每秒可以处理 150 帧图像。这就意味着 YOLOv1 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，YOLOv1 速度快于其他方法。

YOLOv1 实时检测的平均精度是其他实时监测系统的两倍。并且迁移能力强，能运用到其他的新的领域（比如艺术品目标检测）。

4.2. 缺点分析

YOLOv1 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。

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。（因为对于小的边界框，微小误差的影响更大）

并且 YOLOv1 对不常见的角度的目标泛化性能偏弱。

5. 笔记中涉及的相关技术点

5.1. CNN 系列算法

CNN 是一种用于物体检测的深度卷积网络，在用户看来，它是一个单一的、端到端的统一网络。该网络可以准确快速地预测不同物体的位置。

5.1.1. CNN

卷积神经网络（Convolutional Neural Network, CNN） 是一种深度学习模型，特别适用于处理图像数据。它通过卷积层、池化层和全连接层提取图像的特征。

- **卷积层**：提取局部特征，通过滤波器（卷积核）进行图像卷积运算。
- **池化层**：通过下采样（如最大池化或平均池化）减少特征图的尺寸，降低计算复杂度。
- **全连接层**：将提取到的特征映射到输出类别上，通常用于图像分类任务。

5.1.2. R-CNN

区域卷积神经网络（Region-based Convolutional Neural Network, R-CNN） 是一种用于目标检测的模型。它通过选择性搜索算法生成候选区域，对每个区域进行卷积神经网络的特征提取，然后进行分类。其主要特点为：

- **选择性搜索**：生成可能包含目标的候选区域（Region Proposals）。
- **特征提取**：对每个候选区域使用 CNN 进行特征提取。
- **分类器**：使用支持向量机（SVM）对提取的特征进行分类。
- **回归器**：对每个候选区域的边界框进行回归以提高定位精度。

5.1.3. Faster R-CNN

快速区域卷积神经网络（Faster R-CNN） 是 R-CNN 的改进版本，进一步提高了检测速度和精度。它引入了区域建议网络（Region Proposal Network, RPN）来生成候选区域，使整个过程端到端训练。其主要特点为：

- **区域建议网络（RPN）**：RPN 与 CNN 共享卷积特征，通过滑动窗口机制生成候选区域。
- **端到端训练**：RPN 和后续的分类器、回归器在同一网络中联合训练，提高了效率。
- **高效性**：相比 R-CNN 和 Fast R-CNN，Faster R-CNN 显著减少了候选区域生成的时间，提高了检测速度。

5.1.4. 对比总结

- **CNN**：用于图像分类，主要由卷积层、池化层和全连接层组成。
- **R-CNN**：用于目标检测，通过选择性搜索生成候选区域，使用 CNN 提取特征并进行分类。
- **Faster R-CNN**：改进了 R-CNN，引入了 RPN，端到端训练，提高了检测速度和精度。

5.2. 双阶段（two-stage）

双阶段目标检测网络首先生成一系列候选区域，然后再对这些区域进行分类和边界框回归。这种方法的典型代表是 R-CNN 系列算法（包括 Fast R-CNN, Faster R-CNN 等）。其步骤为：

- **区域提议（Region Proposal）**：第一阶段主要是提出潜在的目标区域，通常使用区域提议网络(Region Proposal Network, RPN)或其他机制来预测图像中可能包含目标的区域。
- **分类与回归**：在第二阶段，对这些提议的区域进行更细致的分类，并精确调整边界框位置。

其优点为精度高，由于有一个专门的阶段来精确调整边界框和分类，这种方法通常能达到较高的检测精度。同时拥有好的局部化能力，较好地处理了对象的定位问题。

其缺点为速度较慢，由于需要两个阶段的处理，计算成本较高。且训练和推理流程更复杂。

5.3. 单阶段（One-Stage）

单阶段目标检测网络直接从输入图像预测各类目标的类别和位置，无需区域提议步骤。代表算法包括 YOLO（You Only Look Once）、SSD（Single Shot Multibox Detector）等。其步骤为：

- **直接预测**：网络在单一的前向传递中直接预测不同类别的得分和对应的边界框位置，通常通过在多个尺度上预测来增强其对不同大小目标的检测能力。

其优点为速度快，由于省去了区域提议阶段，单阶段检测器能在速度上显著优于双阶段检测器，适合实时应用。并且结构简单，训练和推理流程直接、简洁。

其缺点为精度较低，通常单阶段检测器的精确度低于双阶段检测器，特别是在小对象的检测上。并且定位能力较弱。在边界框的精确定位方面可能不如双阶段检测器准确。

双阶段和单阶段网络在目标检测领域各有千秋。双阶段方法以其高精度和优异的定位能力在需要精确检测的场景中更受青睐，而单阶段方法则因其高速度和简洁性在需要快速响应的应用（如视频监控、自动驾驶）中更为常见。选择哪种方法取决于具体应用的需求和可用资源。

5.4. 边界框（bounding box）

在 YOLO 目标检测系统中，边界框（bounding box）是一个关键的概念。边界框用于定义图像中被检测对象的位置和尺寸。YOLO 通过在单次前向传播中预测图像中的所有对象的边界框和类别概率，从而实现快速且高效的目标检测。

其具体策略已在章节 3 介绍，此处不再赘述。

5.5. 类别概率

在 YOLO（You Only Look Once）目标检测系统中，类别概率是用来描述每个预测边界框可能包含的对象类别的概率。这些概率表示了边界框中包含每个可能类别的信心水平。这是目标检测中一个重要的组成部分，因为它不仅需要准确地定位对象，还需要正确地识别对象的类别。

其具体策略已在章节 3 介绍，此处不再赘述。

5.6. 回归问题

在机器学习领域，回归问题是一类旨在预测连续数值结果的问题。与分类问题不同，分类问题的目标是预测离散的标签或类别，回归问题则关注如何预测一个或多个连续变量的值。回归模型的输出是连续的数值，如价格、温度、长度等。

回归分析通常旨在确定输入变量和输出变量之间的关系，尤其是输入变量如何影响输出结果的量化模型。

回归模型的训练通常涉及最小化预测输出与实际输出之间的误差，例如通过最小化均方误差（MSE）。

常见的回归方法包括：线性回归、多项式回归、逻辑回归等。

回归评估指标：均方误差（MSE）、均方根误差（RMSE）、平均绝对误差（MAE）、R^2（决定系数）等。这些指标度量的是预测值和实际值之间的差异大小。

5.7. 分类问题

在机器学习领域，分类问题预测离散的标签或类别。分类模型的输出是有限的类别集中的一个，这些类别通常是互斥的。比如说：判断电子邮件是否为垃圾邮件、诊断病人是否患有某种疾病，或者识别图像中的对象类别。

常见的分类方法包括：决策树、随机森林、支持向量机（SVM）、神经网络等。

分类评估指标：准确率、精确率、召回率、F1 分数、混淆矩阵、ROC-AUC 曲线等。这些指标评估的是分类正确性和类别预测的性能。

选择回归或分类取决于预测目标的性质：如果目标是预测一个具体的数量值，那么回归是合适的选择；如果目标是判定输入属于哪个类别，那么分类是更好的选择。理解这些差异有助于选择合适的模型和算法，以解决特定的数据科学问题。

5.8. 模型前向过程（Model Forward Process）

- 这是目标检测算法的核心计算阶段，涉及将输入图像通过网络模型进行一次前向传播。
- 在 YOLO 中，这个过程通常包括多个卷积层、激活函数、池化层等，用于提取图像特征。
- 网络结构可能包括 DarkNet、CSPNet、EfficientRep、ELAN 等，这些结构设计用于高效地提取图像特征并减少计算量。
- 前向传播的输出是关于图像中目标位置和类别的预测，通常包括边界框（bounding boxes）的坐标和类别概率。

5.9. 区域提议网络（Region Proposal Network, RPN）

区域提议网络（Region Proposal Network, RPN）是一种用于目标检测任务的神经网络结构，它在 Faster R-CNN 架构中首次被引入。RPN 的主要目的是自动化并加速候选区域（region proposals）的生成过程，这些候选区域随后用于精确地定位和分类图像中的对象。

RPN 通过学习来直接从全卷积网络的最后一层特征图中预测物体的边界和其"对象性"得分（即一个区域是否包含任何对象的概率）。这是通过以下步骤实现的：

5.9.1. 特征提取

RPN 是建立在卷积神经网络（CNN）的基础上，利用 CNN 提取的特征图作为输入。这些特征图通过前面的卷积层学习到图像的高级特征。

5.9.2. 锚点（Anchors）

在特征图上的每个位置，RPN 使用多种尺寸和比例的预定义框（称为锚点）。这些锚点覆盖不同的区域大小和形状，以适应可能的对象形状和尺寸。

5.9.3. 候选区域提议

对于每个锚点，RPN 同时预测两类输出：

- 边界框回归偏移量（Bounding box regressions）：调整锚点的位置和尺寸，使其更精确地匹配潜在的目标对象。
- 对象性得分（Objectness scores）：评估每个锚点内包含目标的可能性。

5.9.4. 训练 RPN

RPN 的训练使用正样本（包含对象的锚点）和负样本（不包含对象的锚点）。通过最小化分类误差（是否包含对象）和回归误差（边界框的准确性），来训练网络。

5.9.5. RPN 与 Faster R-CNN 的结合

在 Faster R-CNN 架构中，RPN 是一个关键组成部分，它使得区域提议过程与对象检测网络可以共享相同的卷积特征，提高了效率和速度：

- RPN 生成的候选区域首先经过一系列的过滤和非极大值抑制（NMS）步骤，以减少候选区域的数量和重叠。
- 然后，这些精炼的候选区域被送入 Faster R-CNN 的后续部分进行详细的对象分类和更精细的边界框回归。

RPN 通过在目标检测流程中自动化和加速候选区域的生成，显著提高了目标检测任务的速度和效率。这种方法通过深度学习直接从图像特征中提取有用的空间区域，与传统的基于手工特征和复杂启发式的方法相比，提供了一种更为直接和有效的解决方案。

5.10. 机器学习常见激活函数

在机器学习和深度学习中，激活函数在神经网络模型中扮演着至关重要的角色，它们帮助引入非线性因素，使得网络能够学习和表达复杂的数据模式。以下是一些最常见的激活函数：

5.10.1. 线性激活函数

线性激活函数是一种在机器学习模型中使用的简单激活函数，其输出直接等于输入，没有引入非线性。这种激活函数保持输入数据的线性特性不变。在数学上，线性激活函数可以表示为：

$f(x) = ax + b$

其中 a 和 b 是常数。在最简单的形式中，这些常数可以设定为 $a = 1$ 和 $b = 0$，从而得到一个纯粹的线性激活函数：

$f(x) = x$

线性激活函数不会引入非线性，这意味着无论神经网络有多少层，如果所有层都使用线性激活函数，整个网络仍然是线性的。这限制了网络处理复杂问题（如分类非线性可分数据）的能力。但是线性激活函数非常简单，易于计算，不会增加训练过程的计算负担。

在解决回归问题时，尤其是输出为连续值的情况，线性激活函数是非常适合的。例如，在预测房价或股票价格等问题中，模型输出层通常会使用线性激活函数。

其局限性为：线性激活函数不适合处理复杂的数据模式，如分类和聚类非线性可分的数据集。并且使用线性激活函数的网络无法执行像分类和识别在内的更多复杂任务，因为这些任务通常依赖于输入数据的高度非线性映射。

5.10.2. Sigmoid 函数

- **公式**：$\sigma(x) = \frac{1}{1+e^{-x}}$
- **特点**：Sigmoid 函数输出一个介于 0 和 1 之间的值，常用于二分类问题中。它可以将任意值映射到(0, 1)区间，非常适合用作输出层的激活函数，用来预测概率。
- **缺点**：Sigmoid 函数在输入值很大或很小时会出现梯度消失问题，使得网络难以继续学习。

5.10.3. ReLU 函数（Rectified Linear Unit）

- **公式**：$\mathrm{f}(x) = \max(0,x)$
- **特点**：ReLU 函数在输入大于 0 时直接输出该值，否则输出 0。由于其计算简单，训练时的收敛速度比 Sigmoid 和 Tanh 函数快，已成为深度学习中最常用的激活函数之一。
- **缺点**：ReLU 在输入小于 0 时完全不激活，这可能导致“死神经元”问题，即部分神经元永远不会被激活，影响模型的学习能力。

5.10.4. Softmax 函数

Softmax 函数是一种在机器学习中常用的激活函数，特别是在处理多类分类问题时。它将一个含有任意实数的向量转换成另一个含有实数的向量，而转换后的向量元素的值域为(0,1)之间，并且各元素值的总和为 1。因此，softmax 函数的输出可以被视为一个概率分布，用来表示每个类别的概率。

定义：给定一个实数向量 $\mathbf{z}$（通常是分类器的原始输出，即逻辑回归模型或神经网络的最后一层非激活输出），softmax 函数定义为：

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

其中：

- $\sigma(\mathbf{z})_j$是经过 softmax 转换后的第 j 个元素的值。
- z_j是向量$\mathbf{z}$中的第 j 个元素。
- K 是向量$\mathbf{z}$的元素总数，也是类别的总数。
- e 是自然对数的底数，约等于 2.71828

工作原理：

- **指数操作**：Softmax 首先对每一个输入 z_j进行指数运算，确保每一个输出都是非负的。

- **归一化**：然后，每个指数化后的e^{z_j}会被除以所有指数化结果的总和。这一步操作使得所有输出值的和等于 1，从而可以将其解释为概率。

应用场景：

- 多类分类：Softmax 函数通常用于多类逻辑回归模型和神经网络的最后一层，使得模型输出可以表示为概率分布。这样，每个类别的输出值表示了输入属于该类别的概率。
- 神经网络：在深度学习中，当任务涉及到多个类别的判定时（如图像分类、文本分类等），通常在网络的输出层使用 softmax 函数。
- 概率解释：由于 softmax 输出的是一个概率分布，它可以用于任何需要概率解释的场景，比如自然语言处理中的语言模型、强化学习中的策略输出等。

特点：

- **软最大化**：Softmax 是"软性"的最大化方法。虽然它倾向于增强最大值并抑制较小的值，但与硬最大化（取向量中的最大值）不同，softmax 保留了原始向量中的所有成分的一些信息。
- **溢出问题**：在实际应用中，直接对 z_j使用指数函数可能导致数值上的溢出，这可以通过从所有输入值中减去最大值来进行数值稳定的处理。

总结来说，softmax 函数通过将原始输出转化为概率分布，不仅使模型的输出更加直观，还方便了后续的概率计算和决策制定，是多类分类问题中不可或缺的工具。

5.10.5. Tanh 函数（双曲正切函数）

公式：$\tanh(x) = \frac{2}{1+e^{-2x}} - 1$

- **特点**：Tanh 函数输出介于-1 和 1 之间的值，它是 Sigmoid 函数的变体，输出值被规范化到了-1 到 1 的范围内，这使得其数据分布相对均匀，有时能提高学习效率。
- **缺点**：与 Sigmoid 函数类似，Tanh 函数也会在输入值绝对值较大时出现梯度消失的问题。

5.10.6. Leaky ReLU 函数

公式：$f(x) = \max(\alpha x, x)$，其中α是一个很小的常数。

特点：Leaky ReLU 是 ReLU 的一个改进版本，它允许小于 0 的值有一个非零的梯度（如 $\alpha = 0.01$ ），从而解决了 ReLU 的死神经元问题。

优点：保持了 ReLU 的非线性特性和计算简便性，同时改善了信息在训练过程中的流失问题。

5.10.7. 结论

选择合适的激活函数对神经网络的性能有重大影响。不同的激活函数具有不同的数学特性和适用场景，合理选择激活函数可以提高网络的学习效率和预测性能。在实际应用中，常常需要根据具体问题和网络结构通过实验来确定最合适的激活函数。

5.11. 卷积层

在神经网络中，卷积层（Convolutional Layer）是一种特别设计用于处理具有已知网格结构的数据（如图像）的层。卷积层通过应用一种称为卷积的数学运算来提取输入数据的特征，这使得卷积神经网络（CNN）非常适合于图像处理、视频分析、自然语言处理等任务。

5.11.1. 卷积操作

卷积层使用一组可学习的过滤器（或称为核），每个过滤器负责从输入数据中提取特定类型的特征。例如，在图像中，一个过滤器可能用于检测边缘，而另一个可能用于捕捉纹理信息。

每个过滤器在输入数据上滑动（通常称为滑动窗口或内核窗口），并在每个位置上应用卷积运算，计算过滤器与其覆盖的输入数据之间的点积。

5.11.2. 特征映射（Feature Maps）

通过将过滤器应用于输入数据的所有可能位置，卷积层生成了一个特征映射（或激活图），该图表达了过滤器检测到的特征在输入数据中的空间分布。

如果输入数据具有多个通道（如 RGB 图像的三个颜色通道），卷积通常是跨所有输入通道进行的，并为每个过滤器产生一个二维特征映射。

5.11.3. 步幅和填充

- **步幅（Stride）**：定义了过滤器移动的步长。较大的步幅会减少特征映射的空间尺寸。
- **填充（Padding）**：为了控制特征映射的空间尺寸，可以在输入数据的边界添加额外的零填充。

5.11.4. 非线性激活

卷积后通常会应用一个非线性激活函数（参考上文：机器学习常见激活函数）于特征映射，以引入非线性，这使得网络能够捕捉更复杂的特征。

5.11.5. 应用和优势

- **参数共享**：在卷积层中，每个过滤器的参数在整个输入上共享，这显著减少了模型的参数数量，降低了过拟合的风险，并提高了模型的泛化能力。
- **局部连接**：卷积层的每个神经元只与输入数据的一个局部区域连接，这符合自然图像的局部性原理，即图像的小邻域内的像素之间通常相关性更强。
- **多层次特征提取**：在深度卷积神经网络中，通过堆叠多个卷积层，网络能够从简单的边缘和纹理特征逐步学习到复杂的对象特征，实现从低级到高级的特征表达。

卷积层因其高效的特征提取能力和适应多种数据类型的灵活性，成为了现代深度学习模型中不可或缺的一部分，特别是在视觉相关的任务中发挥着核心作用。

5.12. 全连接层（FC）

神经网络中，全连接层（Fully Connected Layer，通常缩写为 FC）是一种架构组件，其中网络的每个神经元都与前一层的所有神经元连接。全连接层通常用于神经网络的深层部分，负责综合之前层（如卷积层或池化层）提取的特征，并最终输出一个可以用于分类或回归的结果。

5.12.1. 工作原理

在全连接层中，每个神经元都与前一层的每个神经元通过权重连接。这意味着，如果前一层有 n 个神经元，而全连接层有 m 个神经元，则会有 n×m 个权重需要学习。

数学上，全连接层可以表示为矩阵乘法。如果把前一层的输出表示为向量 x，权重矩阵表示为 W，偏置向量为 b，则全连接层的输出 y 可以通过公式$y = Wx + b$计算得到。

全连接层的输出通常会通过一个激活函数进行非线性变换，以引入非线性因素，使网络能够学习复杂的函数。

5.12.2. 功能和应用

- **特征整合**：全连接层能够整合前面层（如卷积层）提取的局部特征，进行全局信息的综合与分析。
- **决策制定**：在分类任务中，全连接层常用于根据整合的特征做出最终的分类决策。在典型的 CNN 架构中，最后几个全连接层的输出可以被用来预测类别标签。
- **过拟合问题**：由于全连接层参数众多，它们更容易导致过拟合。为了缓解这个问题，常常在全连接层后使用如 Dropout 这样的正则化技术。
- **转换为其他任务**：虽然在图像处理任务中全连接层经常被用于输出分类结果，但它们也可以修改用于其他任务，如回归任务（预测连续值）或者作为其他复杂网络结构的一部分。

全连接层是神经网络中至关重要的一部分，它通过全面的方式连接网络中的信息，使得神经网络能够进行复杂的决策和预测。在实际使用中，设计合理的全连接层结构和采取适当的正则化措施是提高网络性能和泛化能力的关键。

5.13. 池化层

池化层（Pooling Layer），也称为下采样（Subsampling）或池化操作，是卷积神经网络中常见的一种结构组件，主要用于减小特征图（feature maps）的维度，从而减少计算量、内存消耗和防止过拟合。池化层通过对输入特征图中的邻近数据点进行聚合操作来实现这一功能。

5.13.1. 基本类型

- **最大池化（Max Pooling）**：最大池化是最常用的池化形式，它的操作是在输入特征图的指定区域（如 2x2 窗口）内取最大值。这种方法能够有效捕捉特征的局部极值，是非常有效的保留图像纹理信息的方式。

- **平均池化（Average Pooling）**：平均池化则是计算输入特征图的指定区域内的平均值。它提供了一种平滑特征的方法，有助于抑制噪声，但可能不如最大池化在保留特征的明确性上有效。

5.13.2. 工作原理

- **窗口大小**：指定一个窗口（通常是 2x2 或 3x3 大小），这个窗口会在输入的特征图上滑动。
- **步长（Stride）**：步长定义了窗口滑动的间隔。
- **操作**：在窗口覆盖的每个区域内，根据池化类型（最大值或平均值）进行操作，并将结果构成新的输出特征图。
- **边界处理（Padding）**：与卷积层类似，池化操作也可以选择是否对输入特征图进行边缘填充（padding），以调整输出大小。

5.13.3. 功能和应用

- **降维**：通过减少特征图的空间尺寸，池化层显著减少了后续层的参数数量和计算复杂度，提高了计算效率。
- **抗过拟合**:通过减少学习参数和模型复杂度,池化层有助于控制过拟合,提高模型泛化能力。
- **保持空间不变性**：池化操作可以提取区域特征，使特征对小的位置变化保持不变性，这对于处理自然图像中常见的小位移和形变尤为重要。
- **增强特征**：最大池化通过提取局部最强信号来强调重要的特征，而平均池化则有助于提取背景特征。

池化层是卷积神经网络不可或缺的一部分，它通过简单的下采样操作实现了对特征的压缩和抽象，既减少了计算负担，又增强了模型对输入变化的适应能力。在设计 CNN 时，合理地使用池化层是优化网络结构和提升性能的关键步骤。

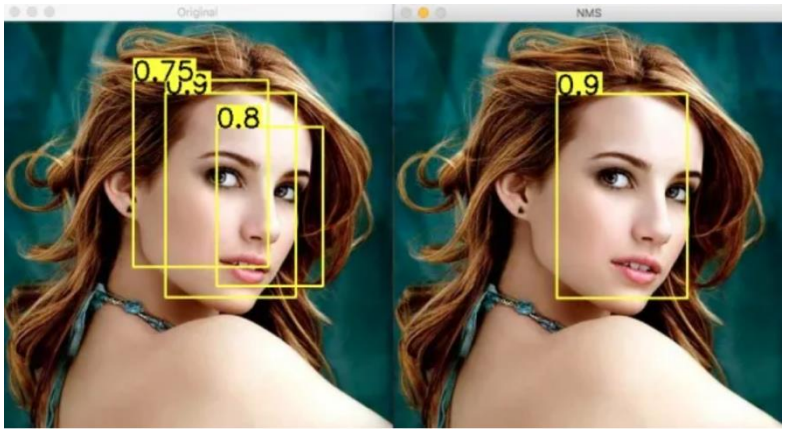
5.14. 非极大值抑制（NMS）

5.14.1. 什么是非极大值抑制

非极大值抑制，简称为 NMS 算法，英文为 Non-Maximum Suppression。其思想是搜素局部最大值，抑制非极大值。NMS 算法在不同应用中的具体实现不太一样，但思想是一样的。非极大值抑制，在计算机视觉任务中得到了广泛的应用，例如边缘检测、人脸检测、目标检测（DPM，YOLO，SSD，Faster R-CNN）等。

5.14.2. 为什么要用非极大值抑制

以目标检测为例：目标检测的过程中在同一目标的位置上会产生大量的候选框，这些候选框相互之间可能会有重叠，此时我们需要利用非极大值抑制找到最佳的目标边界框，消除冗余的边界框。Demo 如下图：

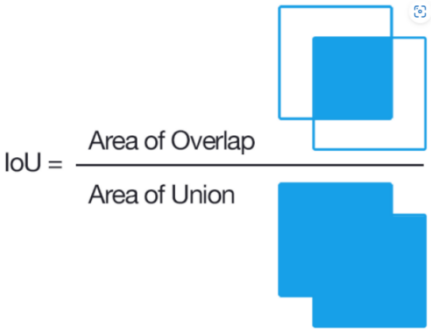


左图是人脸检测的候选框结果，每个边界框有一个置信度得分(confidence score)，如果不使用非极大值抑制，就会有多个候选框出现。右图是使用非极大值抑制之后的结果，符合我们人脸检测的预期结果。

5.14.3. 如何使用非极大值抑制

前提：目标边界框列表及其对应的置信度得分列表，设定阈值，阈值用来删除重叠较大的边界框。

IoU：intersection-over-union，即两个边界框的交集部分除以它们的并集。



如上图，很简单，IoU 相当于两个区域重叠的部分除以两个区域的集合部分得出的结果。

一般来说，这个 score ＞ 0.5 就可以被认为一个不错的结果了。

非极大值抑制的流程如下：

- 根据置信度得分进行排序

- 选择置信度最高的比边界框添加到最终输出列表中，将其从边界框列表中删除
- 计算所有边界框的面积
- 计算置信度最高的边界框与其它候选框的 IoU。
- 删除 IoU 大于阈值的边界框
- 重复上述过程，直至边界框列表为空。