目录

[1.传统目标检测方法 1](#_Toc5180)

[1.1 Viola-Jones (人脸检测) 2](#_Toc29588)

[1.2 HOG+SVM (行人检测、Opencv实现) 2](#_Toc31956)

[1.3 DPM (物体检测) 3](#_Toc12264)

[2.One stage 4](#_Toc7200)

[2.1 Yolo体系变化 4](#_Toc12619)

[2.2 YOLO网络结构是怎样的？ 5](#_Toc26054)

[2.3 YOLO的输入、输出、损失函数分别是什么？ 5](#_Toc10258)

[2.4 YOLO怎样预测？ 5](#_Toc20195)

[2.5 YOLOv3不使用Softmax对每个框进行分类的原因？ 6](#_Toc10883)

[2.6 YOLOv3的架构 6](#_Toc7132)

[2.7 YOLOv4的架构 7](#_Toc7638)

[BackBone训练策略 8](#_Toc11719)

[检测头训练策略 8](#_Toc13444)

[检测头推理策略 9](#_Toc21520)

[2.8 YOLOv5的架构 9](#_Toc5076)

[2.8.1 yolo5原理 9](#_Toc3137)

[2.8.2 yolov5引入了CSP结构，介绍一下它的原理和作用？ 10](#_Toc769)

[2.9 为什么SSD对小目标检测效果不好? 10](#_Toc10521)

[2.10 Anchor 10](#_Toc22647)

[2.10.1 为什么引入Anocher？ 10](#_Toc25434)

[2.10.2 Anocher为什么要使用不同尺寸和长宽比？ 10](#_Toc15853)

[2.10.3 Anochor Box的尺寸该怎么选择？ 11](#_Toc10733)

[3.Two stage 11](#_Toc27247)

[Faster R-CNN 11](#_Toc29781)

[Faster RCNN的loss有哪些？分别讲下。 12](#_Toc10641)

[Mask RCNN 12](#_Toc12677)

[ROI 13](#_Toc354)

[ROI Pooling 13](#_Toc32402)

[目标检测回归损失函数有哪些？ 13](#_Toc5340)

[One Stage Vs Two Stage 14](#_Toc27616)

[4.RPN 14](#_Toc3806)

[5.MTCNN 15](#_Toc3250)

[6.算法的比较 16](#_Toc506)

[6.1 优缺点 16](#_Toc11338)

[6.2 性能评价指标 16](#_Toc19347)

[7.人脸检测相关问题 18](#_Toc7420)

[7.1 目前主要有人脸检测方法分类？ 18](#_Toc15656)

[7.2 如何检测图片中不同大小的人脸？ 18](#_Toc30942)

[7.3 如何设定算法检测最小人脸尺寸? 19](#_Toc9407)

[7.4 如何定位人脸的位置？ 19](#_Toc10495)

[7.5 如何通过一个人脸的多个框确定最终人脸框位置？ 20](#_Toc21958)

[7.6 如何理解端到端？ 20](#_Toc28490)

[7.7 目标检测技巧有哪些？ 20](#_Toc7657)

[8.分类、目标检测、语义分割、实例分割的区别 21](#_Toc20601)

[9.做目标检测的话，数据集用什么？ 21](#_Toc11693)

[9.1 用自己的数据集做目标检测，在采集数据时应注意哪些问题？ 21](#_Toc26322)

1.传统目标检测方法

## 1.1 Viola-Jones (人脸检测)

## 1.2 HOG+SVM (行人检测、Opencv实现)

## 1.3 DPM (物体检测)

**NMS (非极大值抑制算法)**  
  
1 利用得分，给检测框排序:  
选出最大得分的检测框A1，  
将与A1的IOU重叠率高的检测框，进行删除,  
2 其余的剩下的检测再次排序，选择最大的A2，  
将与A2的IOU重叠率高的检测框，再次进行删除,  
3 …迭代多次

# 2.One stage

One-stage核心组件： CNN网络 + 回归网络  
  
主要包含SSD，YOLOv1、YOLOv2、YOLOv3系列。  
主干网络：CNN； 核心组件：回归网络

## 2.1 Yolo体系变化

**解释**：**cmBN**: Conv + Mish + BatchNorm

## 2.2 YOLO网络结构是怎样的？

强烈推荐阅读：[Yolo发展史](https://blog.csdn.net/g11d111/article/details/108845799)

YOLO网络借鉴了GoogLeNet分类网络结构，不同的是YOLO使用1x1卷积层和3x3卷积层替代inception module。如下图所示，整个检测网络包括24个卷积层和2个全连接层。其中，卷积层用来提取图像特征，全连接层用来预测图像位置和类别概率值。

## 2.3 YOLO的输入、输出、损失函数分别是什么？

YOLO将输入图像分成7x7的网格，最后输出是7x7xk的张量。YOLO网络最后接了两个全连接层，全连接层要求输入是固定大小的，所以YOLO要求输入图像有固定大小，论文中作者设计的输入尺寸是448x448。

## 2.4 YOLO怎样预测？

采用 NMS 算法从输出结果中提取最有可能的对象和其对应的边界框。  
**手写NMS**:  
[**https://zhuanlan.zhihu.com/p/75348108**](https://zhuanlan.zhihu.com/p/75348108)  
NMS步骤如下：  
1.设置一个Score的阈值，一个IOU的阈值；  
2.对于每类对象，遍历属于该类的所有候选框，  
①过滤掉Score低于Score阈值的候选框；  
②找到剩下的候选框中最大Score对应的候选框，添加到输出列表；  
③进一步计算剩下的候选框与②中输出列表中每个候选框的IOU，若该IOU大于设置的IOU阈值，将该候选框过滤掉，否则加入输出列表中；  
④最后输出列表中的候选框即为图片中该类对象预测的所有边界框  
3.返回步骤2继续处理下一类对象。

## 2.5 YOLOv3不使用Softmax对每个框进行分类的原因？

Softmax使得每个框分配一个类别（score最大的一个），而对于Open Images这种数据集，目标可能有重叠的类别标签，因此Softmax不适用于多标签分类。  
Softmax可被独立的多个logistic分类器替代，且准确率不会下降。

## 2.6 YOLOv3的架构

YOLOv3在之前Darknet-19的基础上引入了残差块，并进一步加深了网络，改进后的网络有53个卷积层，取名为Darknet-53。

强烈推荐阅读： [yolo3深度解析](https://blog.csdn.net/leviopku/article/details/82660381)

图片引用：[YOLOv3网络结构图](https://zhuanlan.zhihu.com/p/162043754)

## 2.7 YOLOv4的架构

[架构解析](https://blog.csdn.net/x454045816/article/details/109759989)  
  
**图片若侵权：可联系删除**

CSPDarknet53  
解决了其他大型CNN框架Backbone中网络优化的梯度信息重复问题，将梯度的变化从头到尾地集成到特征图中，因此减少了模型的参数量和**FLOPS**(每秒浮点运算次数)数值，既保证了推理速度和准确率，又减小了模型尺寸。

### BackBone训练策略

1.采用了**Mosaic**数据增强  
(参考了**CutMix**数据增强; 区别:Mosaic是一种将4张训练图像合并成一张进行训练的数据增强方法(而不是CutMix中的2张)。  
每个小批包含一个大的变化图像(4倍)，减少了估计均值和方差的时需要大mini-batch的要求，降低了训练成本。  
2.**DropBlock**正则化（引入：克服Dropout随机丢弃特征）：块的相邻相关区域中丢弃特征。  
3.类标签平滑

### 检测头训练策略

1.**CIoU**-loss： CIOU只是在DIOU基础上增加了一项 av  
2.**CmBN**策略： 只在每个Batch内部使用CBN的方法  
3.**SAT**：  
第一阶段：NN改变原始图像而不是网络权值。通过这种方式，NN对其自身进行一种**对抗式**的攻击，改变原始图像，制造图像上没有目标的假象；  
第二阶段：训练NN对修改后的图像进行正常的目标检测。  
消除网格敏感度  
4.单目标使用**多Anchor**  
5.**余弦**模拟退火

### 检测头推理策略

Mish, SPP, PAN,SAM和DIoU-NMS  
为了增大感受野，使用了**SPP**-block(对任意尺寸的特征图直接进行固定尺寸的池化-> 固定数量的特征)，使用PAN代替FPN进行参数聚合以适用于不同level的目标检测。

SPP：用来解决不同尺寸的特征图如何进入全连接层的  
PAN: 代替FPN进行参数聚合以适用于不同level的目标检测,YOLOv4算法将融合的方法由**addition**改为**Concatenation**

DarkNet-53相对DarkNet-19的有如下几点改进：  
1.加深网络层，精度提升，但是速度有所下降，  
2.进入了残差网络resNet模块，防止梯度下降；  
3.用卷积strid==2代替了池化，防止信息丢失  
4.使用SPP实现多尺寸输入，同尺寸输出。

**名词解释**：  
CSPNet (跨阶段局部网络)  
SPP-Net（空间金字塔池化网络）:Spatial Pyramid Pooling Networks  
PAN(路径聚合网络):Path Aggregation Network  
SAT(自对抗训练): Self-Adversarial Training

## 2.8 YOLOv5的架构

### 2.8.1 yolo5原理

原理可以分为四部分：输入端、backbone、Neck、输出端；  
输入端：针对小目标的检测，沿用了v4的**mosaic**增强，当然这个也是v5作者在他复现的v3上的原创，对不同的图片进行随机缩放、裁剪、排布后进行拼接；二是自适应锚框计算，在v3、v4中，初始化锚框是通过对coco数据集的进行聚类得到，v5中将锚框的计算加入了训练的代码中，每次训练时，自适应的计算不同训练集中的最佳锚框值；  
backbone：沿用了V4的**CSPDarkNet53**结构，但是在图片输入前加入了Focus切片操作，CSP结构实际上就是基于**Densnet**的思想，复制基础层的特征映射图，通过dense block发送到下一个阶段，从而将基础层的特征映射图分离出来。这样可以有效缓解梯度消失问题，支持特征传播，鼓励网络重用特征，从而减少网络参数数量。在V5中，提供了**四种不同大小的网络结构**：s、m、l、x，通过**depth**（深度）和**width**（宽度）两个参数控制。  
Neck：采用了**SPP**+**PAN**多尺度特征融合，PAN是一种自下而上的特征金字塔结构，是在FPN的基础上进行的改进，相对于FPN有着更好的特征融合效果。  
输出端：沿用了V3的head，使用**GIOU**损失进行边框回归，输出还是三个部分：**置信度**、**边框信息**、**分类信息**。

### 2.8.2 yolov5引入了CSP结构，介绍一下它的原理和作用？

CSP结构是一种思想，它和ResNet、DenseNet类似，可以看作是DenseNet的升级版，它将feature map拆成两个部分：  
一部分进行**卷积操作**；另一部分和上一部分卷积操作的结果进行**concate**。主要解决了三个问题：  
1.增强CNN的学习能力，能够在轻量化的同时保持着准确性；2. 降低计算成本；3. 降低内存开销。  
CSPNet改进了**密集块**和**过渡层**的信息流，优化了**梯度反向传播**的路径，**提升**了网络的学习能力，同时在处理**速度**和**内存**方面提升了不少。

## 2.9 为什么SSD对小目标检测效果不好?

小目标对应的anchor比较少，其对应的feature map上的pixel难以得到训练，这也是为什么SSD在augmentation之后精确度上涨（因为crop之后小目标就变为大目标）要检测小目标需要足够大的feature map来提供精确特征，同时也需要足够的语义信息来与背景作区分。

## 2.10 Anchor

定义: 提前在图像上预设好的不同大小，不同长宽比的框;同一位置设置多个不同尺度先验框

### 2.10.1 为什么引入Anocher？

使得模型更容易学习。使用anchor boxes之后，YOLOv2的召回率大大提升，所以在Yolo之后的版本中，均保留了先验框。

### 2.10.2 Anocher为什么要使用不同尺寸和长宽比？

为了得到更大的交并比

### 2.10.3 Anochor Box的尺寸该怎么选择？

在SSD、Faster-RCNN中，设计了9个不同大小和宽高比的anchor；  
anchor box的选择主要有三种方式：  
1.人为经验选取；2.k-means聚类；3.作为超参数进行学习  
YOLOv2中建议使用K-means聚类来代替人工设计，通过对训练集的bbox进行聚类，自动生成一组更加适合数据集的anchor，可以使网络的检测效果更好。  
虽然设置更多的先验框，IOU能获得一定的提升，但模型的复杂度随之增加，YOLOv2的作者选择了K=5个先验框。

# 3.Two stage

概念：先由算法生成一系列作为样本的候选框，再通过卷积神经网络进行样本分类。  
主要是R-CNN，SPP系列，Fast R-CNN，Faster R-CNN等；  
目标检测方法概述：  
主要通过一个CNN来完成目标检测过程，其提取的是CNN卷积特征，在训练网络时，其主要训练两个部分，第一步是训练RPN网络，第二步是训练目标区域检测的网络。网络的准确度高、速度相对One-stage慢。

## Faster R-CNN

Fast R-CNN依赖于外部候选区域方法，如选择性搜索。在测试中，Fast R-CNN需要2.3秒来进行预测，其中2秒用于生成2000个ROI。Faster R-CNN采用与Fast R-CNN相同的设计，只是它用内部深层网络代替了候选区域方法。新的候选区域网络（RPN）在生成ROI时效率更高，并且以每幅图像10毫秒的速度运行。

### Faster RCNN的loss有哪些？分别讲下。

第一部分为分类损失，第二部分为回归损失

## Mask RCNN

创新点：1. Backbone：ResNeXt-101 + FPN  
2. RoI Align替换RoI Pooling  
是一个实例分割算法，主要是在目标检测的基础上再进行分割。算法主要是Faster R-CNN+FCN，更具体一点就是ResNeXt + RPN + RoI Align + Fast R-CNN + FCN。

Mask R-CNN算法步骤：

1. 输入一幅你想处理的图片，然后进行对应的预处理操作，或者预处理后的图片；
2. 将其输入到一个预训练好的神经网络中（ResNeXt等）获得对应的feature map；
3. 对这个feature map中的每一点设定预定个的RoI，从而获得多个候选RoI；
4. 将这些候选的RoI送入RPN网络进行二值分类（前景或背景）和BB回归，过滤掉一部分候选的RoI；
5. 对这些剩下的RoI进行RoI Align操作；
6. 对这些RoI进行分类（N类别分类）、BB回归和MASK生成（在每一个RoI里面进行FCN操作）。

## ROI

RoI是Region of Interest，一般是指图像上的区域框，但这里指的是由Selective Search提取的候选框。往往经过RPN后输出的不止一个矩形框，所以这里我们是对多个RoI进行Pooling。

## ROI Pooling

目的是对非均匀尺寸的输入执行最大池化以获得固定尺寸的特征图。  
我们获得的anchor尺寸是不同的，能直接对其全连接，应该将区域提案划分为相等大小的部分（其数量与输出的维度相同）  
；然后进行max pooling.  
ROI Polling Vs ROI Align  
ROIPool存在两次量化误差，首先是将候选框边界量化为整数点坐标值，其次是将量化后的边界区域平均分割成 k x k 个单元，对每一个单元的边界进行量化。ROIAlign通过双线性插值避免了量化操作，保存了原始ROI的空间分布，有效避免了误差的产生；对于检测图片中大目标物体时，两种方案的差别不大，而如果是图片中有较多小目标物体需要检测，则优先选择ROIAlign，更精准一些。

## 目标检测回归损失函数有哪些？

演进路线是：  
Smooth L1 Loss --> IoU Loss --> GIoU Loss --> DIoU Loss --> CIoU Loss  
**GIoU loss** 仍然存在收敛速度慢、回归不准等问题  
目标检测中IOU是如何计算的？ 检测结果与 Ground Truth 的交集比上它们的并集。  
  
IOU 为 0 时，两个框不重叠，没有交集。  
IOU 为 1 时，两个框完全重叠。  
IOU 取值为 0 ～ 1 之间的值时，代表了两个框的重叠程度，数值越高，重叠程度越高。

# One Stage Vs Two Stage

# 4.RPN

Roi: 抠图+resize(固定同样尺寸大小)  
RPN网络结构？  
生成anchors -> softmax分类器提取positvie anchors -> bbox reg回归positive anchors -> Proposal Layer生成proposals  
RPN 执行两种不同类型的预测：二进制分类和边框回归调整。为了训练，我们把所有的锚 anchor box 分成两类。一类是「前景」，它与真实目标重叠并且其 IoU值大于 0.5；另一类是「背景」，它不与任何真实目标重叠或与真实目标的 IoU 值 小于 0.1。

# 5.MTCNN

流程：   
图片经过Pnet，会得到feature map，通过分类、NMS筛选掉大部分假的候选；  
然后剩余候选去原图crop图片输入Rnet，再对Rnet的输出筛选掉False、NMS去掉众多的候选；  
剩余候选再去原图crop出图片再输入到Onet，这个时候就能够输出准确的bbox、landmark坐标了。

1.由原始图片和PNet生成预测的bounding boxes。  
2.输入原始图片和PNet生成的bounding box，通过RNet，生成校正后的bounding box。  
3.输入元素图片和RNet生成的bounding box，通过ONet，生成校正后的bounding box和人脸面部轮廓关键点。

# 6.算法的比较

## 6.1 优缺点

## 6.2 性能评价指标

AP是在单个类别下的，mAP是AP值在所有类别下的均值；  
mAP 针对的就是**类识别准确率**，但是在目标检测任务中还有一个**边框回归任务**，框的准确率一般用交并比来**IoU** 衡量：

# 7.人脸检测相关问题

## 7.1 目前主要有人脸检测方法分类？

主要包含两个区域：传统人脸检测算法和基于深度学习的人脸检测算法。  
传统人脸检测算法主要可以分为4类： 基于知识、模型、特征和外观的人脸检测方法；  
基于级联CNN的人脸检测（cascade cnn）、 基于多任务CNN的人脸检测（MTCNN）、Facebox等，很大程度上提高了人脸检测的鲁棒性。  
当然通用目标检测算法像Faster-rcnn、yolo、ssd等也有用在人脸检测领域，也可以实现比较不错的结果，但是和专门人脸检测算法比还是有差别。

## 7.2 如何检测图片中不同大小的人脸？

传统人脸检测算法的策略：  
（1）缩放图片的大小（2）缩放滑动窗的大小

基于深度学习的人脸检测算法的策略：  
（1）缩放图片大小。（也可以通过缩放滑动窗的方式，基于深度学习的滑动窗人脸检测方式效率会很慢存在多次重复卷积，所以要采用全卷积神经网络（FCN），用FCN将不能用滑动窗的方法。）  
（2）通过anchor box的方法（通过特征图预测原图的anchor box区域）。

## 7.3 如何设定算法检测最小人脸尺寸?

主要是看滑动窗的最小窗口和anchorbox的最小窗口。  
（1）滑动窗的方法  
假设通过12×12的滑动窗，不对原图做缩放的话，就可以检测原图中12×12的最小人脸。但是往往通常给定最小人脸a=40、或者a=80，以这么大的输入训练CNN进行人脸检测不太现实，速度会很慢，并且下一次需求最小人脸a=30\*30又要去重新训练，通常还会是12×12的输入，为满足最小人脸框a，只需要在检测的时候对原图进行缩放即可：w=w×12/a。  
（2）anchorbox的方法  
原理类似，这里主要看anchorbox的最小box，可以通过缩放输入图片实现最小人脸的设定。

## 7.4 如何定位人脸的位置？

1）滑动窗的方式：基于分类器识别为人脸的框的位置确定最终的人脸;  
2）FCN的方式  
通过特征图映射到原图的方式确定最终识别为人脸的位置，特征图映射到原图人脸框是要看特征图相比较于原图有多少次缩放（缩放主要查看卷积的步长和池化层）  
3）通过anchor box的方式：  
通过特征图映射到图的窗口，通过特征图映射到原图到多个框的方式确定最终识别为人脸的位置。  
  
滑动窗的方式

## 7.5 如何通过一个人脸的多个框确定最终人脸框位置？

通过NMS得到最终的人脸位置：  
NMS改进版本有很多，最原始的NMS就是判断两个框的交集，如果交集大于设定的阈值，将删除其中一个框，那么两个框应该怎么选择删除哪一个呢？ 因为模型输出有概率值，一般会优选选择概率小的框删除。

## 7.6 如何理解端到端？

就是输入一张图像，直接可以给出检测结果，中间不需要你进行任何操作，很方便。非端到端就是不能一步搞定的，需要分步进行。RCNN就是典型的非端到端的例子，一张图过来，首先需要利用SS等方法提取Region Proposals，然后送入到神经网络提取特征，最后还要利用SVM分类，不能一步到位。

## 7.7 目标检测技巧有哪些？

1.数据增强：随机翻转、随机裁剪、添加噪声等也  
2.多尺度训练/测试  
通过输入更大、更多尺寸的图片进行训练，能够在一定程度上提高检测模型对物体大小的鲁棒性，仅在测试阶段引入多尺度，也可享受大尺寸和多尺寸带来的增益。  
3.全局语境  
把整张图片作为一个RoI，对其进行RoI Pooling并将得到的feature vector拼接于每个RoI的feature vector上，作为一种辅助信息传入之后的R-CNN子网络。  
4.预测框微调/投票法  
微调法最初是在SS算法得到的Region Proposal基础上用检测头部进行多次迭代得到一系列box，在ResNet的工作中，作者将输入R-CNN子网络的Region Proposal和R-CNN子网络得到的预测框共同进行NMS（见下面小节）后处理，  
最后，把跟NMS筛选所得预测框的IoU超过一定阈值的预测框进行按其分数加权的平均，得到最后的预测结果。投票法可以理解为以顶尖筛选出一流，再用一流的结果进行加权投票决策。  
5.随机权值平均（SWA）  
只需快速集合集成的一小部分算力，就可以接近其表现。SWA 可以用在任意架构和数据集上，都会有不错的表现  
6.OHEM(在线难例挖掘)  
两阶段检测模型中，提出的RoI Proposal在输入R-CNN子网络前，我们有机会对正负样本（背景类和前景类）的比例进行调整。通常，背景类的RoI Proposal个数要远远多于前景类，Fast R-CNN的处理方式是随机对两种样本进行上采样和下采样，以使每一batch的正负样本比例保持在1:3，这一做法缓解了类别比例不均衡的问题，是两阶段方法相比单阶段方法具有优势的地方，也被后来的大多数工作沿用。  
7.Soft NMS(软化非极大抑制)  
用于去除重合度（IoU）较高的预测框，只保留预测分数最高的预测框作为检测输出  
8.RoI对齐  
采用双线性插值的方法将RoI的表示精细化，并带来了较为明显的性能提升

# 8.分类、目标检测、语义分割、实例分割的区别

<https://www.cnblogs.com/zxj9487/p/11154316.html>

# 9.做目标检测的话，数据集用什么？

COCO上训练好的模型fine tune。  
PASCALVOC数据集: 有VOC2007和VOC2012两个数据集,类别仅20个,被看成目标检测方向的一个基准数据集。  
COCO数据集:包含80个类别  
ImageNet数据集

## 9.1 用自己的数据集做目标检测，在采集数据时应注意哪些问题？

1.首先可以找一些公开的数据集比如VOC数据集，先跑通实验，能训练和测试成功。确保数据和算法是没有问题的。  
2.其次，制作你的数据集，看是否可以训练成功，然后再去仔细分析测试的结果，去调优。  
3.阅读代码和实现原理，然后去通过调整代码去对算法做一些改进，然后看精度是否有提升。