车辆检测设计文档

1. 实验环境

Python 3.6.0

Opencv 3.4.2

Keras 2.2.4

Tensorflow 1.12.0

Matplotlib 3.0.2

Pillow 5.3.0

1. 实验目的

给定图片，要求能够输出图片中车辆的位置并画出检测框；给定视频，要求能够输

带有车辆检测框的视频流。

1. 实验背景

目标检测技术发展迅速，从运用传统机器学习方法检测目标到如今广泛采用深度学习方法，可以说是变化巨大。如今比较流行的算法可以分为两类：一类是基于Region Proposal的R-CNN系算法（R-CNN，Fast R-CNN, Faster R-CNN）它的特点是准确度高，但是速度慢；而另一类是Yolo，SSD这类one-stage算法，其仅仅使用一个CNN网络直接预测不同目标的类别与位置。它的特点: 速度快，但准确率稍低。以下是近年来目标检测的发展过程：



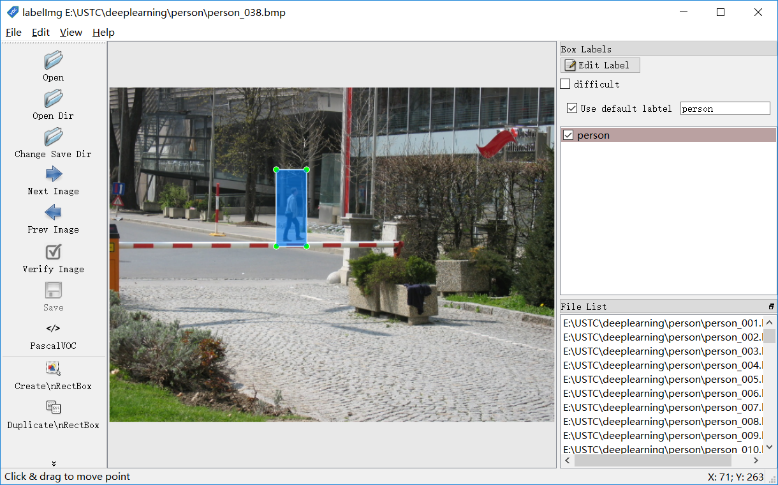
经过调研，我们决定从传统的SVM方法出发对图片和视频检测，再运用现在流行的YOLO3检测图片和视频。

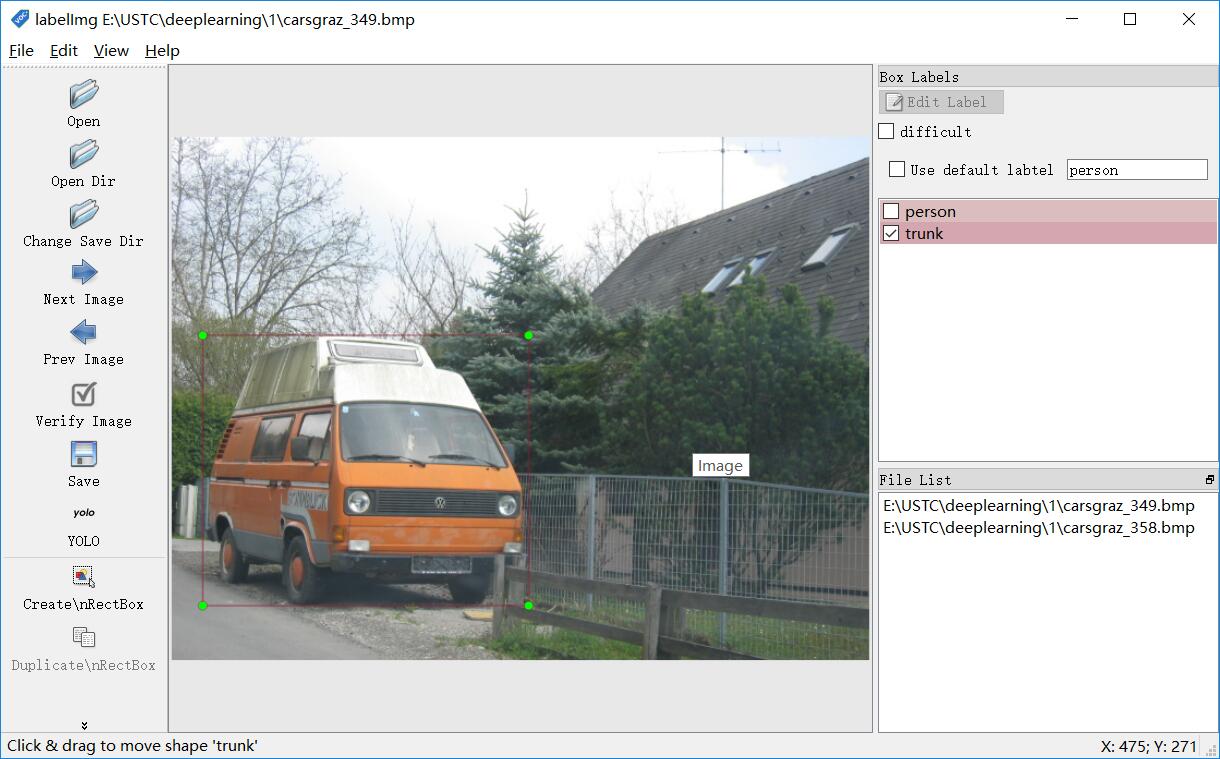
1. 数据标记
2. 选择数据集

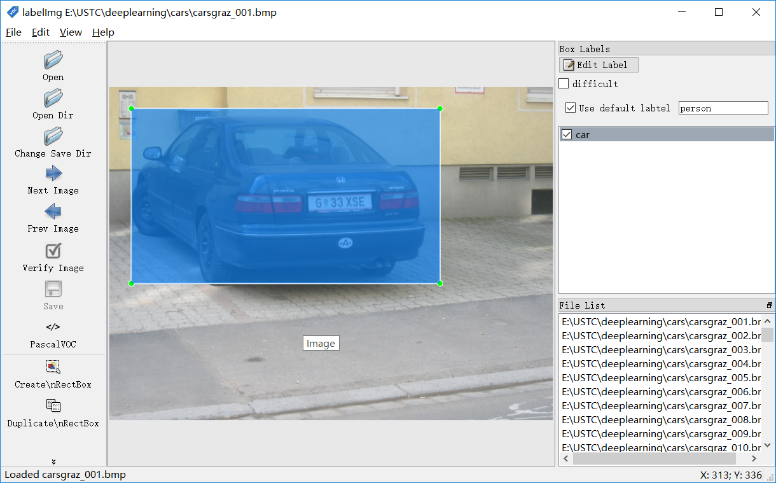
经过筛选，选择了以下几个开源数据集：Owncollection , cars , person。Owncollection数据集图片格式为64\*64，分为vehicles和non-vehicles，每种里面又分为far,left,right,middleclose四种数据集；cars数据集是对车辆图片的一种补充，其图片格式为640\*480；person数据集用来检测行人，其图片格式为640\*480 .以上三个数据集就是本次数据的主要来源，最终前后一共标记了2451张。

1. 进行标记

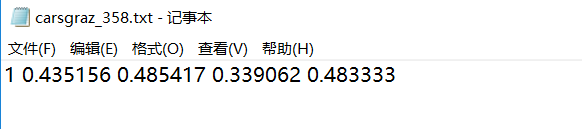
借助labelImg软件进行标记，如图所示，借助软件选择好打开文件夹和保存文件夹便可以开始进行数据标记了，作为一个队友的svm模型，只需要识别出car和非car，即0和1即可，所以只需要标记出car和非car两种类型；作为另一个队友使用的基于yolo3的模型，可以标记出多种种类，如car，person, trunk 。定义种类需要在data文件夹中在predefined\_classes.txt文件中输入需要的类型。接下来便可以开始数据标记工作了。如下图所示：







最终若选择yolo的数据模式，一个图片可以得到五个数据，分别为数据类型以及四个参数，如下图所示:



1. 划分验证集

将数据去拿不标记好后便可以划分数据集了，初步将数据集划分为四类，分别是测试集，训练集，验证集，训练和验证集即test.txt，train.txt，val.txt，trainval.txt。这样就将数据标记的工作做好了。

1. SVM实践
2. 实验数据

训练数据为64x64x3的RGB图片，包含车辆与非车辆图片两类，车辆图片8792张，非车辆图片8968张。

1. HOG特征
   1. 做法

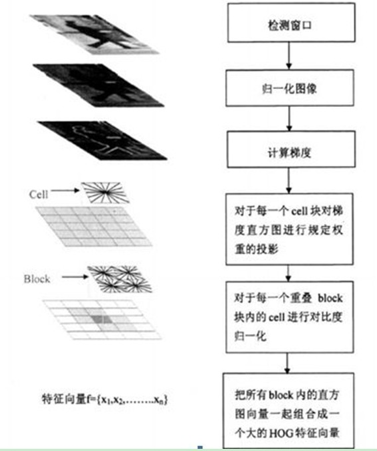
首先将图像分成小的连通区域，我们把它叫细胞单元。然后采集细胞单元中各像素点的梯度的或边缘的方向直方图。最后把这些直方图组合起来就可以构成特征描述器。

* 1. 提高性能

把这些局部直方图在图像的更大的范围内（我们把它叫区间或block）进行对比度归一化（contrast-normalized）所采用的方法是：先计算各直方图在这个区间（block）中的密度，然后根据这个密度对区间中的各个细胞单元做归一化。通过这个归一化后，能对光照变化和阴影获得更好的效果。

* 1. 工作流程

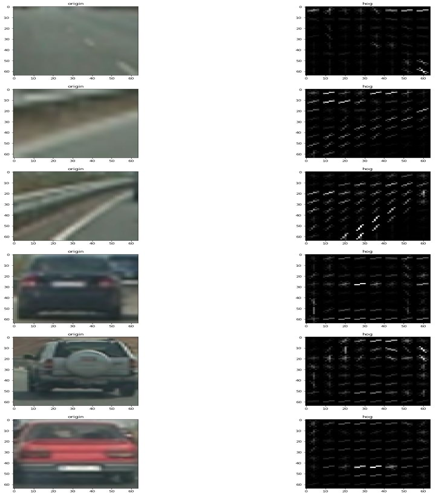
将图像划分成小cells（例如6\*6像素/cell）；每几个cell组成一个block（例如3\*3个cell/block），一个block内所有cell的特征descriptor串联起来便得到该block的HOG特征descriptor。



* 1. HOG优点

首先，由于HOG是在图像的局部方格单元上操作，所以它对图像几何的和光学的形变都能保持很好的不变性。其次，在粗的空域抽样、精细的方向抽样以及较强的局部光学归一化等条件下，细微的目标动作改变并不影响检测效果。

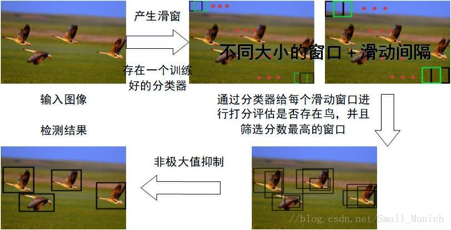
2.5 提取特征对比



1. 训练SVM分类器，分类车辆与非车辆

对数据集进行8/2划分，采用LinearSVC( )分类器。最终训练的分类器在测试数据集得到98.0%准确率。

1. 应用滑动窗口实现车辆检测



* 1. 主要思路

首先对输入图像进行不同窗口大小的滑窗进行从左往右、从上到下的滑动。每次滑动时候对当前窗口执行分类器(分类器是事先训练好的)。如果当前窗口得到较高的分类概率，则认为检测到了物体。对每个不同窗口大小的滑窗都进行检测后，会得到不同窗口检测到的物体标记，这些窗口大小会存在重复较高的部分，最后采用非极大值抑制(Non-Maximum Suppression, NMS)的方法进行筛选。最终，经过NMS筛选后获得检测到的物体。

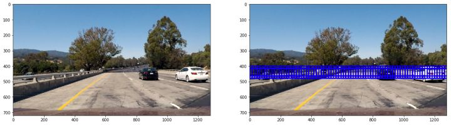
* 1. 总结

滑窗法简单易于理解，但是不同窗口大小进行图像全局搜索导致效率低下，而且设计窗口大小时候还需要考虑物体的长宽比。所以，对于实时性要求较高的分类器，不推荐使用滑窗法。

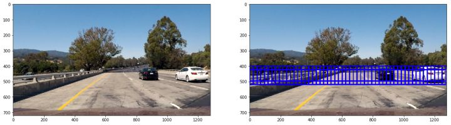
4.3 结果

本次车辆由于距离远近不同会在视频呈现的不一样的大小且出现的位置也会各异，这里使用4类不同大小的滑动窗口对图片中的车辆进行搜索：

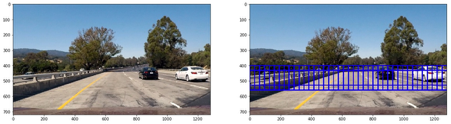
第一类大小为64x64,重叠率(overlap)为0.75，用来检测距离较远的车辆：



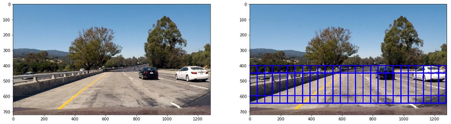
第二类大小为96x96，重叠率(overlap)为0.75，用来检测中距离车辆：



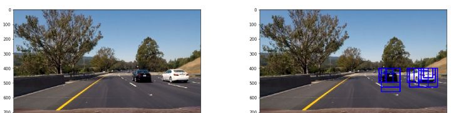
第三类大小为128x128,重叠率(overlap)为0.75，用来检测近距离车辆:



第四类大小为224x224,重叠率(overlap)为0.75，用来检测极近距离车辆:



将以上四类窗口应用在测试图片得到：



由于使用多个大小不一滑动窗口，且窗口存在重叠，单个车辆图像会被多个窗口捕捉检测。可以看到存在多窗口重合情况。



1. 应用热力图过滤错误检测

记录一张图片上所有positive detections，使用记录的positive detections形成一个检测热图：

def add\_heat(heatmap, bbox\_list):

# Iterate through list of bboxes

for box in bbox\_list:

# Add += 1 for all pixels inside each bbox

# Assuming each "box" takes the form ((x1, y1), (x2, y2))

heatmap[box[0][1]:box[1][1], box[0][0]:box[1][0]] += 1



然后对热图进行阈值过滤,过滤错误检测：

def apply\_threshold(heatmap, threshold):

# Zero out pixels below the threshold

heatmap[heatmap <= threshold] = 0

# Return thresholded map

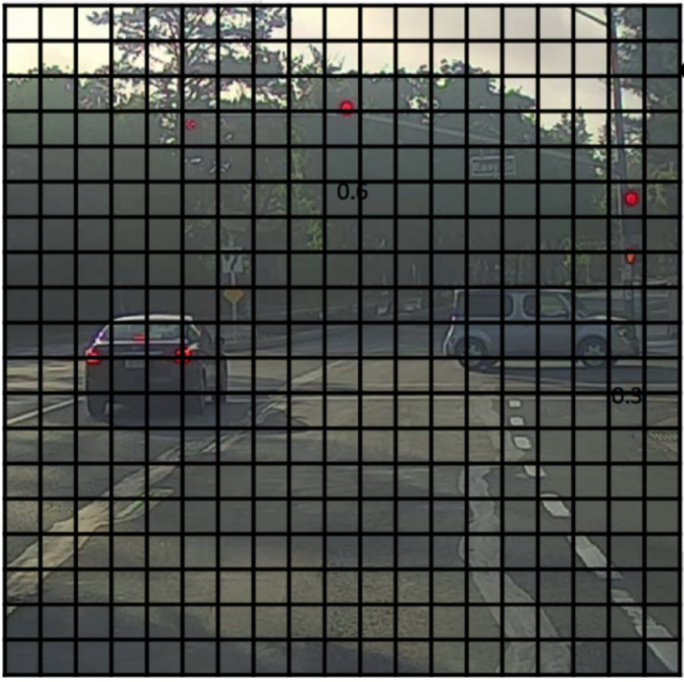
return heatmap

1. YOLOv3实践
2. 概述

YOLO网络利用整张图作为网络的输入，直接在输出层回归bounding box的位置和bounding box所属的类别。

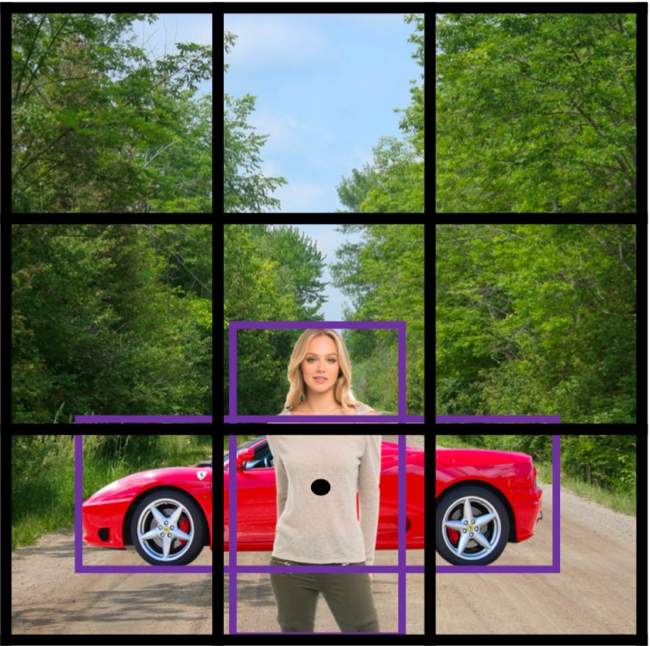
1. 网格单元(Grid cell)

将单张图片划分为sxs个网格单元，如果某个object的中心落在这个网格中，那这个网格就负责预测这个object。一般来说，s可取3，5，7，19等。



1. 锚框(Anchor box)

如下面的这张图，在一个网格单元中同时出现了两个形状不同的物体，这就是锚框的作用，为了检测同一个网格单元中的不同物体。每个网格要预测B个anchor box，每个anchor box除了要回归自身的位置之外，还要预测一个confidence值。如果这个网格中有物体，那么confidence应接近1；如果没有物体，confidence应为0。



那么对于1张608x608x3的输入图片，当网格单元是19x19，锚框个数是5时，它的输出向量应该是19 x 19 x 5 x (1 + 4 + c)，c代表类别数。

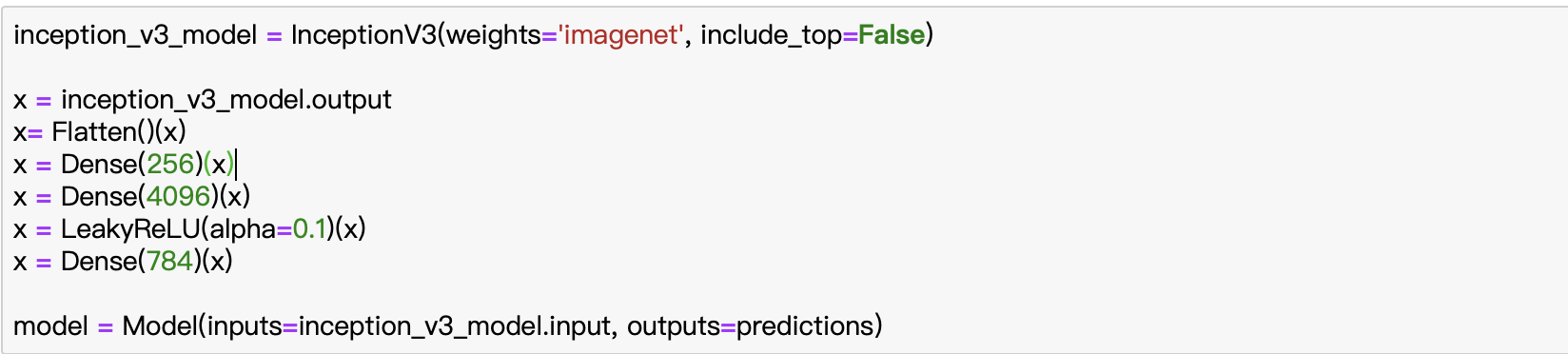
1. 训练方案1: 从0开始实现yolo网络

网络由特征提取层和特征预测层组成，特征提取层用来提取图片的抽象特征，而特征预测层是利用前一层提取得到的特征作分类。



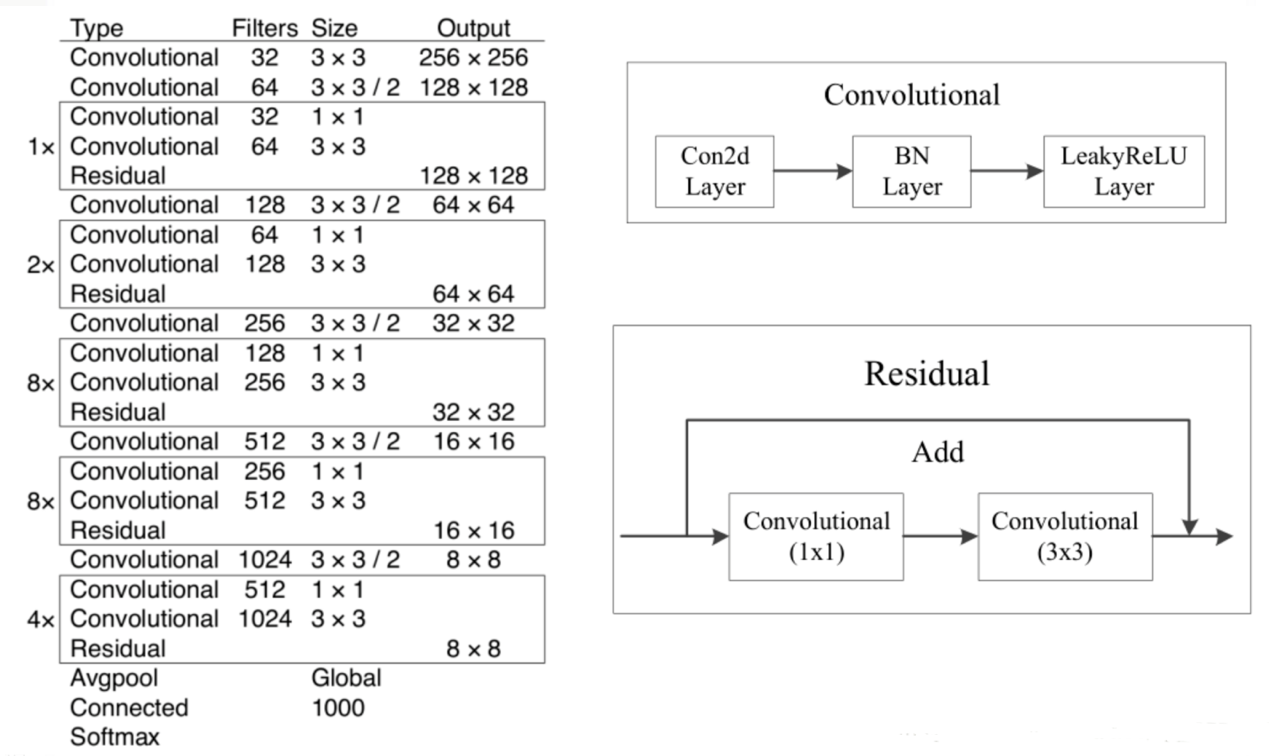
1. 训练方案2: 迁移学习

将已经在imagenet数据集上训练好的InceptionV3网络用作特征提取器，然后加上自己的类别分类器，实现整个网络结构。



1. 训练方案3: Darknet-53

Darknet-53是最新的yolo网络结构，由卷积块，残差块和全连接层组成。我们直接加载了论文中的最佳参数。

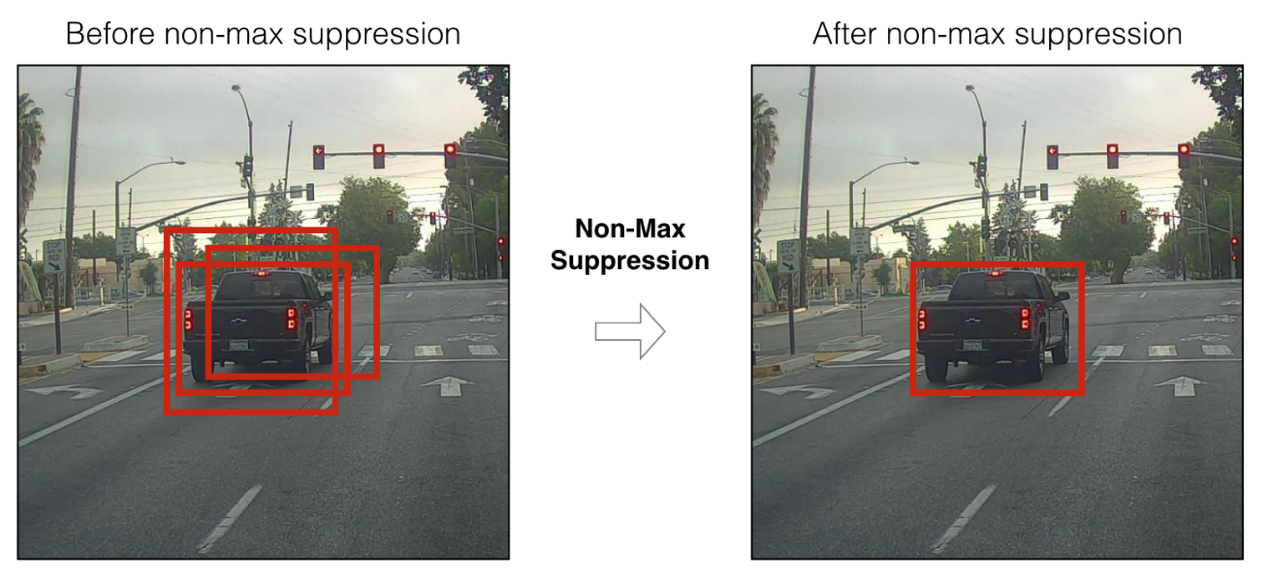


7. 预测算法: 非最大值抑制

a. 选择具有最高得分的锚框；

b. 计算该锚框与其他锚框的IoU，并移除大于iou\_threshold的锚框；

c. 重复以上两个步骤直到所有锚框被选择。



1. 结果展示
2. 图片

输入 输出

1. 视频

答辩时已经演示过。