# 充电宝遮挡检测

宁尧、张利、张丽斌、张远

1实现原理

## 1.1 主流目标检测算法

充电宝遮挡检测使用目标检测算法，识别目标并给出其位置；目前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），它们是two-stage的，需要先使用启发式方法（selective search）或者CNN网络（RPN）产生Region Proposal，然后再在Region Proposal上做分类与回归。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。第一类方法是准确度高一些，但是速度慢，但是第二类算法是速度快，但是准确性要低一些。各类目标检测算法性能比较如下图1所示；

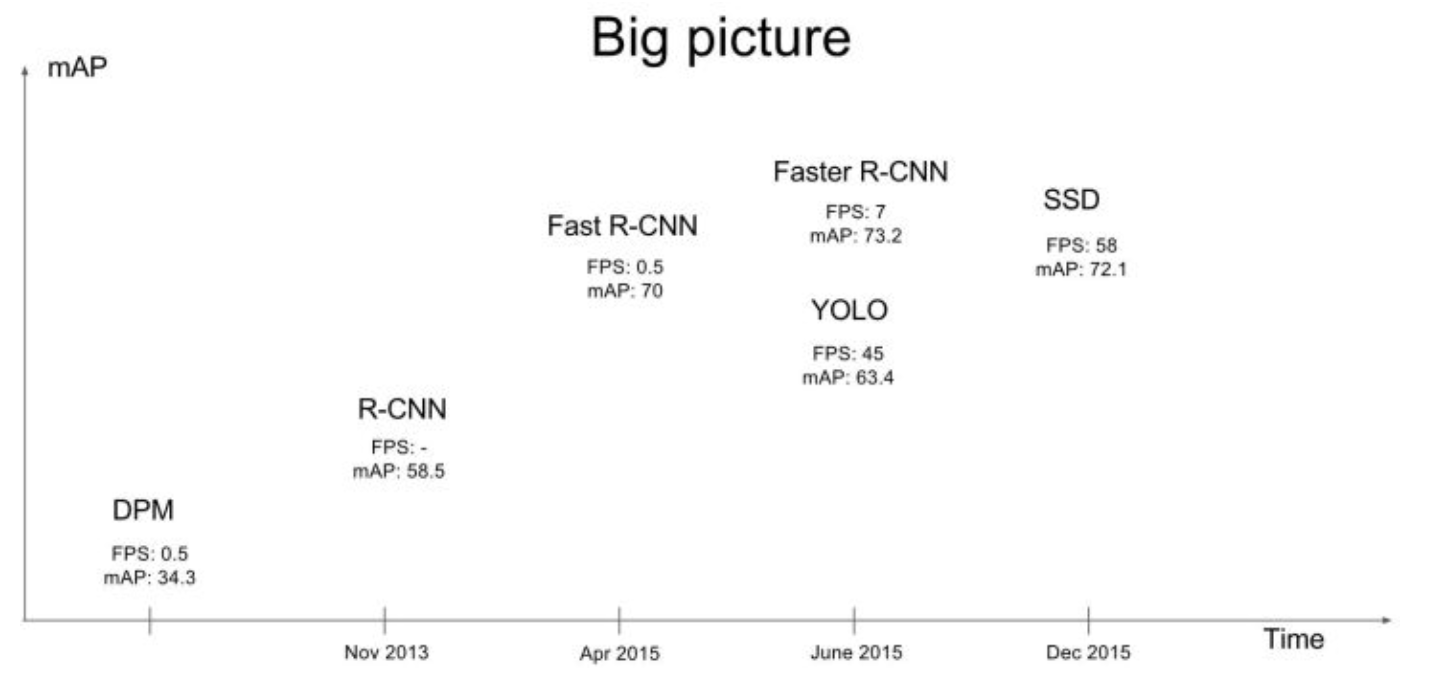


图1 目标检测算法性能比较

## 1.2 YOLO算法

### 1.2.1 原理简述

本实验检测算法为Yolo算法，其全称是You Only Look Once: Unified, Real-Time Object Detection，Yolo是只需要一次CNN运算，Unified指的是这是一个统一的框架，提供end-to-end的预测，而Real-Time体现是Yolo算法速度快；整个系统如图2所示，首先将输入图片resize到448x448，然后送入CNN网络，最后处理网络预测结果得到检测的目标。相比R-CNN算法，其是一个统一的框架，其速度更快，而且Yolo的训练过程也是end-to-end的；

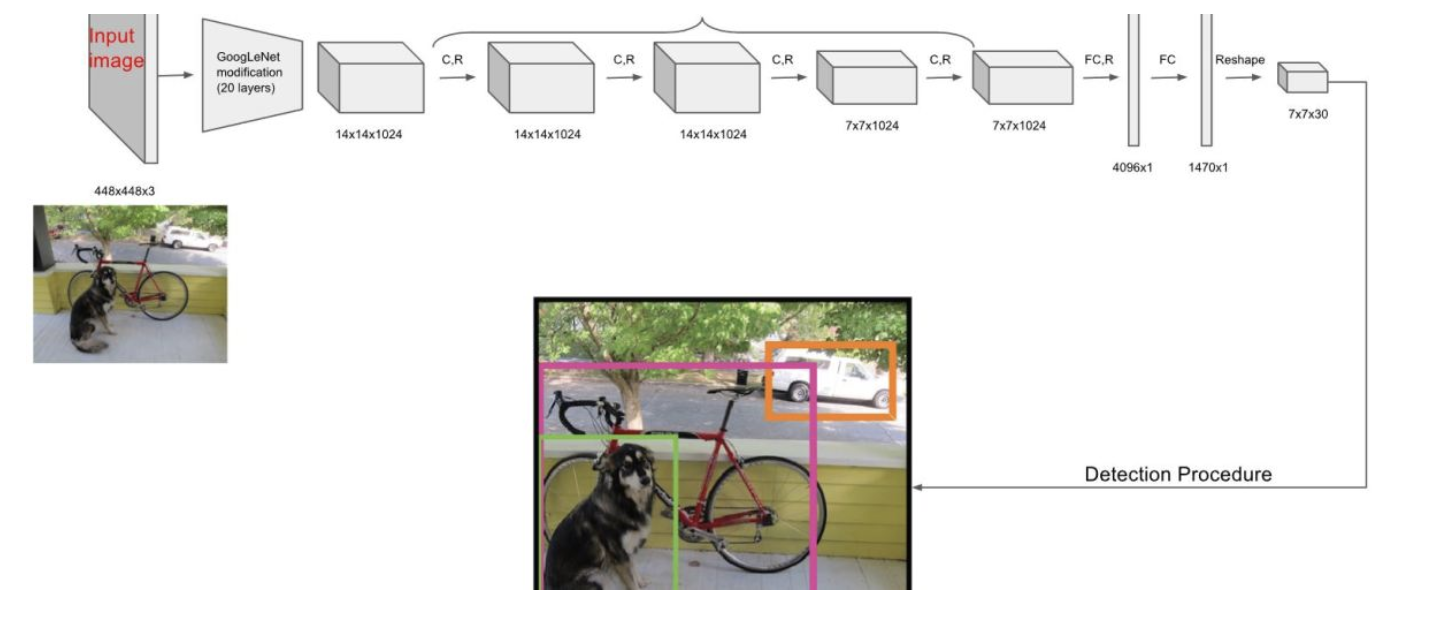


图2

Yolo的CNN网络将输入的图片分割成S \* S网格，然后每个单元格负责去检测那些中心点落在该格子内的目标，如图3所示，可以看到狗这个目标的中心落在左下角一个单元格内，那么该单元格负责预测这个狗。每个单元格会预测B个边界框（bounding box）以及边界框的置信度（confidence score）。所谓置信度其实包含两个方面：一是这个边界框含有目标的可能性大小，二是这个边界框的准确度。

前者记为Pr(Object), 当该边界框是背景时（即不包含目标），此时Pr(Object)=0; 而当该边界框包含目标时, Pr(Object)=1. 边界框的准确度可以用预测框与实际框（ground truth）的IOU（intersection over union，交并比）来表征，记为。因此，置信度可以定义为confidence = Pr(Object) \* ；

边界框的大小和位置可以4个值来表征：（x,y,w,h）,其中（x,y）是边界框的中心坐标，w和h是边界框的宽度和高度，每个边界框的预测值实际上包含5个元素（x,y,w,h,c），其中前4个表征边界框的大小与位置，而最后一个值是置信度。

图片包含 小狗, 建筑物, 室内

描述已自动生成

图3 网格划分

### 1.2.2 网络设计

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，包含24个卷积层和2个全连接层，如图4所示。对于卷积层，主要使用1x1卷积来做channle reduction，然后紧跟3x3卷积。对于卷积层和全连接层，采用Leaky ReLU激活函数max( x,0.1 \* x)。但是最后一层却采用线性激活函数。

图片包含 文字

描述已自动生成

图4

### 1.2.3网络训练

在训练之前，先在ImageNet上进行了预训练，其预训练的分类模型采用图8中前20个卷积层，然后添加一个average-pool层和全连接层。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。

Yolo算法将目标检测看成回归问题，所以采用的是均方差损失函数，损失函数定义如下：其中第一项是边界框中心坐标的误差项，指的是第i个单元格存在目标，且该单元格中的第 j个边界框负责预测该目标。第二项是边界框的高与宽的误差项。第三项是包含目标的边界框的置信度误差项。第四项是不包含目标的边界框的置信度误差项。而最后一项是包含目标的单元格的分类误差项， 指的是第i个单元格存在目标。

图片包含 文字

描述已自动生成

图5

### 1.2.4 网络预测

Yolo算法在的预测过程中使用了非极大值抑制算法（non maximum suppression, NMS），这个算法不单单是针对Yolo算法的，而是所有的检测算法中都会用到。NMS算法主要解决的是一个目标被多次检测的问题，首先从所有的检测框中找到置信度最大的那个框，然后挨个计算其与剩余框的IOU，如果其值大于一定阈值（重合度过高），那么就将该框剔除；然后对剩余的检测框重复上述过程，直到处理完所有的检测框。

### 1.2.5优缺点

（1）优点，Yolo采用一个CNN网络来实现检测，是单管道策略，其训练与预测都是end-to-end，所以Yolo算法比较简洁且速度快。第二点由于Yolo是对整张图片做卷积，所以其在检测目标有更大的视野，它不容易对背景误判。其实我觉得全连接层也是对这个有贡献的，因为全连接起到了attention的作用。另外，Yolo的泛化能力强，在做迁移时，模型鲁棒性高。

（2）缺点，Yolo各个单元格仅仅预测两个边界框，而且属于一个类别。对于小物体，Yolo的表现会不如人意。这方面的改进可以看SSD，其采用多尺度单元格。也可以看Faster R-CNN，其采用了anchor boxes。Yolo对于在物体的宽高比方面泛化率低，就是无法定位不寻常比例的物体。当然Yolo的定位不准确也是很大的问题。

# 2环境准备与运行

代码地址

github: <https://github.com/realdali/nzzz>

环境要求

系统：windows、mac

python 3.7及以上

安装

```

$ git clone git@github.com:realdali/nzzz.git

$ cd nzzz/

$ sudo pip3 install -r requirements.txt

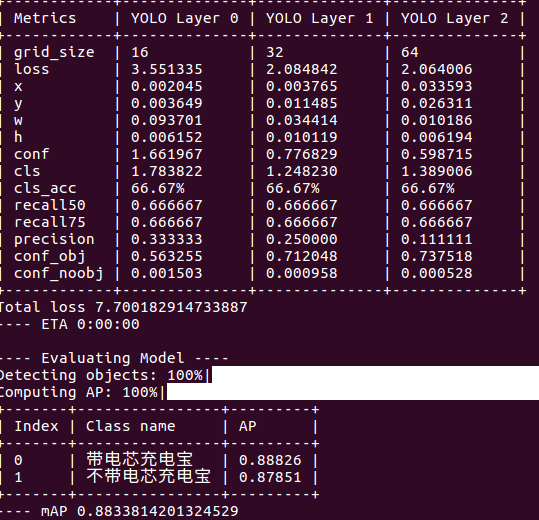
```

## 2.1训练

导入训练数据进行训练

$ python3 train.py --model\_def config/yolov3-custom.cfg --data\_config config/custom.data

得到如下训练结果



## 2.2 验证

在项目根目录执行以下命令

$ python3 test.py --valid\_path data/valid.txt --images\_path data/Image --labels\_path data/Annotation