1、 目标识别算法为 YOLOv3

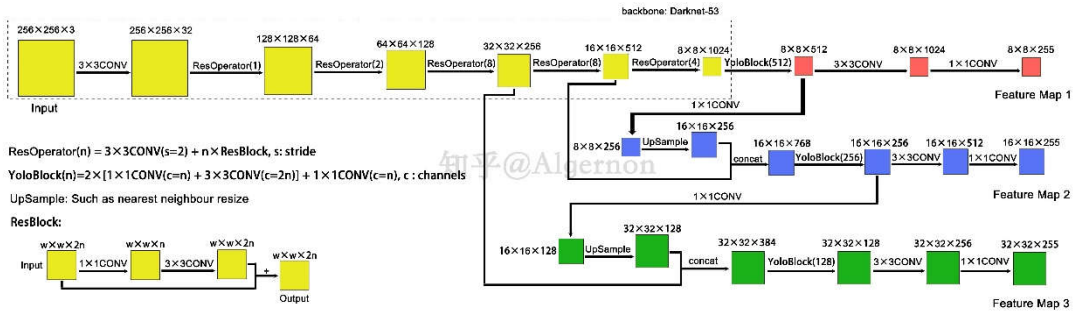
（一）特征图

Yolov3 使用 Darknet-53（黄色部分）作为整个网络的分类骨干部分，且整个网络全部由卷积层组成。Yolov3 总共输出 3 个特征图，第一个特征图下采样 32 倍，第二个特征图下采样 16 倍，第三个下采样 8 倍。

输入图像经过 Darknet-53（无全连接层），再经过 Yoloblock 生成的特征图被当作两用，第一用为经过 3*3 卷积层、1*1 卷积之后生成特征图一（8 倍）

第二用为经过 1*1 卷积层加上采样层，与 Darnet-53 网络的中间层输出结果进行拼接，产生特征图二。（16 倍）

特征图三经过和特征图二同样的循环之后产生。（32 倍）



（二）先验框

Yolov3 沿用了 Yolov2 中回归检测框的宽、高基于先验框的变化的技巧，并且使用 k-means 对数据集中的标签框进行聚类，得到类别中心点的 9 个框，作为先验框。

（三）检测框解码

根据特征图和先验框解码检测框的 x、y、w、h。（x，y）为中心点坐标，w、h 先验框的宽、高。

$$b_x = \sigma(t_x) + c_x$$
$$b_y = \sigma(t_y) + c_y$$
$$b_w = p_w e^{t_w}$$
$$b_h = p_h e^{t_h}$$

$\sigma(t_x)$、$\sigma(t_y)$是基于矩形框中心点左上角格点坐标的偏移量，p_w、p_h是先验框的宽、高。

2、 前后端通讯

（一） 前端

安卓采用 forest 轻量级 HTTP 客户端框架。

先创建一个 interface，并在 interface 中创建相应接口，调用请求。本 App 主要采用两个 POST 请求，第一个 POST 请求上传一张需要检测的图片，并采用 json 格式返回服务端中识别后的图片路径等信息；之后发送第二个 POST 请求，参数为需要下载的图片路径，从服务端中下载识别成功的图片。

```
@Post(url = "http://47.95.148.117:8090/upload", timeout = 120000)
String upload(@DataFile("file") String filePath, OnProgress onProgress);

@Post(url = "http://47.95.148.117:8090/download", timeout = 120000)
@DownloadFile(dir = "${0}", filename = "${1}")
File downloadFile(String dir, String filename, OnProgress onProgress, @JSONBody("downloadPath") String path);
```

（二） 后端

后端即服务器采用 nodejs 来搭建通讯，确定通讯端口为 8090，通过调用 listen 在服务端一直监听 8090 端口是否有接受到请求，如果接受到请求则调用相应脚本（shell 脚本）执行烟雾目标检测程序并返回结果。

```
//设置服务端监听端口为8090，成功后回调在控制台上打印提示
let server = app.listen(8090,()=>{
    console.log('The server is listening on port : 8090')
})
```

```
source /root/anaconda3/etc/profile.d/conda.sh
conda activate pytorch
path="/root/Android-Smoke-app/smokePredict"
io_path="/root/Android-Smoke-app/img_out"
cd $path
command="python predict.py predict ${1} ${io_path}/img_out ${io_path}/smoke_out "
result=`$command`
echo $result
conda deactivate
```