关于数字图像处理的学习与实践报告

报告人：王婧人

一、Opencv 综述

OpenCV 全称 Open Source Computer Vision Library,是由 Intel 公司资助的开源计算机视觉库。它能够实现图像处理和计算机视觉方面的很多通用算法。计算机视觉与机器学习密不可分，OpenCV 也内置了很多较为常用的机器学习算法，近年来在入侵检测、特定目标跟踪、目标检测、人脸检测、人脸识别、人脸跟踪、安防、航拍等领域，OpenCV 都发挥着巨大的作用。其作用的范围已不仅仅是图像处理领域，在视频图像的分析中也发挥着很大的作用。

二、实践

1. 图片边缘轮廓查找

通过对宫颈癌细胞的处理，找到宫颈癌细胞的边缘和轮廓，完成了对 opencv 的学习实践。对宫颈癌细胞处理中我尝试了两个方法，一个是对细胞核进行处理的区域生长法，另一个是对整个细胞进行处理的分水岭算法：

1. 区域生长法

区域生长是根据预先定义的生长准则将像素子区域组合为更大区域的过程,区域生长的基本思想是将具有相似性质的象素集合起来构成区域。首先对每个需要分割的区域找出一个种子象素作为生长的起点，然后将种子象素周围邻域中与种子有相同或相似性质的象素合

并到种子象素所在的区域中。而新的象素继续做种子向四周生长，直到再没有满足条件的象素可以包括进来，一个区域就生长而成了。[1]

这里采用的是自动种子点的区域生长算法，通过阈值分割法和区域生长算法的结合，来实现自动种子的区域生长。图 1 是关于该算法的步骤。

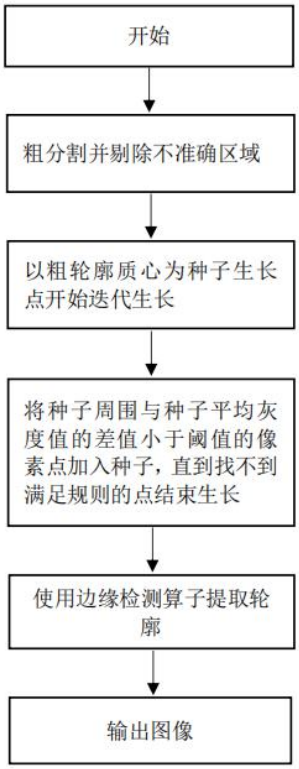


图 1：自动种子点区域生长算法步骤图

通过对操作数据的可视化，得到了关于宫颈癌细胞的细胞核的边缘分割图(图 3、图 4)。黑色的是由区域生长法得到的细胞核边缘，蓝色的则是裁剪的边框，同时为了去除无意义边框，这里设置了一个面积阈值，将面积小于 500 像素的边框舍弃。下面是关于自动种子区域生长算法的效果图。

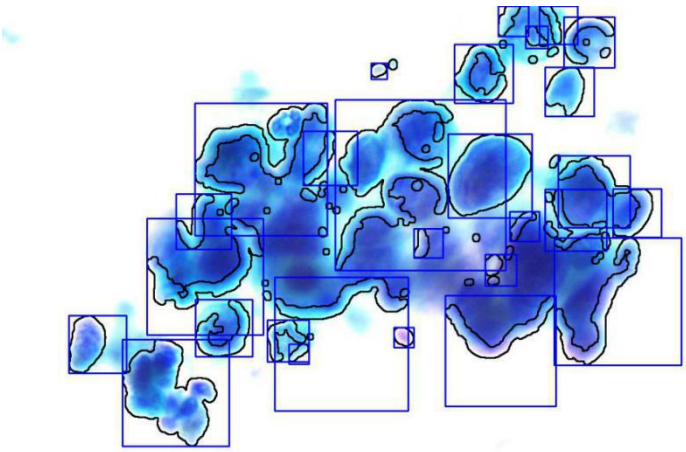


图 2

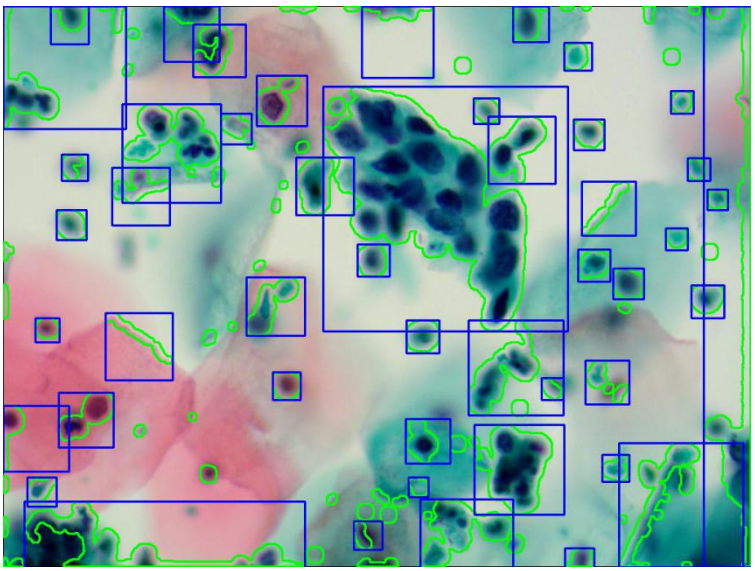


图 3

通过对比可以看出，该算法对于细胞核分散的样本效果较好，但是不可避免的是无法对多个合并在一起的细胞核进行分割，并且不能很好的分割色差较小的细胞质与细胞核。同时该算法的速度十分缓慢，需要大量的时间来运行该算法，效率不高。

2.分水岭算法

该算法的核心思想是将图像中灰度值的差异看作地势的差异，图像中灰度值较低的像素点对应于地势中的较低点，每个局部极小值及其影响区为集水盆，分隔的边界形成分水岭。在灰度值较高的山脊上建立堤坝，防止和其他区域融合，这样当水位达到最高山脊时，所建立的一系列堤坝成为了隔开各个区域的分水岭。应用到图像分割中，分水岭变换是指将原图像转换成一个标记图像，其中所有属于同一集水盆的点均被赋予同一个标记，并用一个特殊的标记来标识分水岭上的点[2] 。

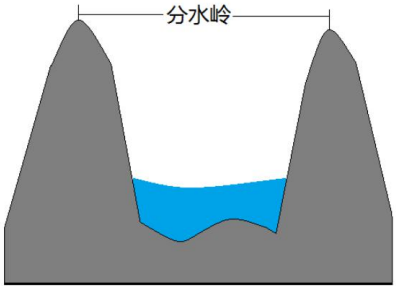


图 4

效果图如下(图 5，图 6、图 7)：

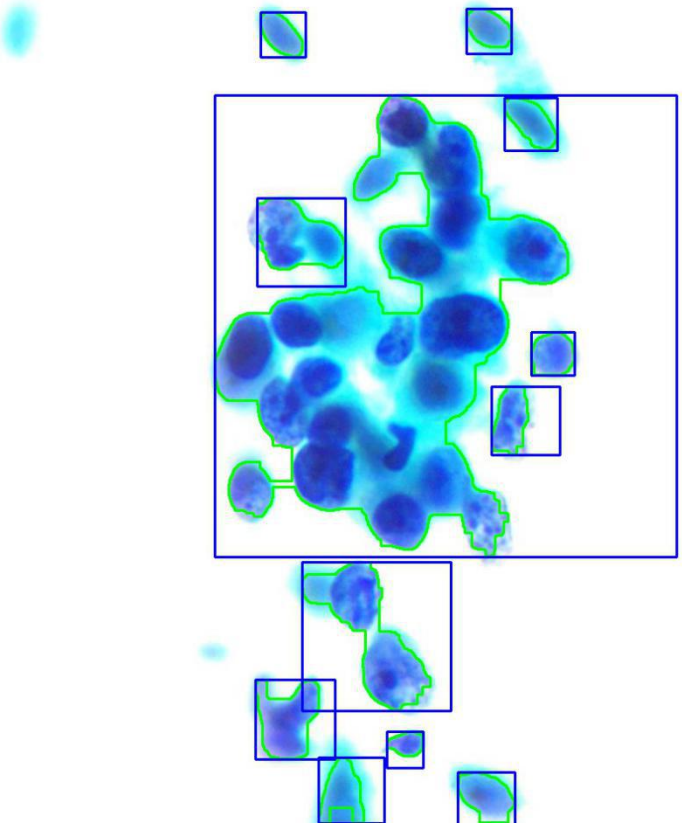


图 5

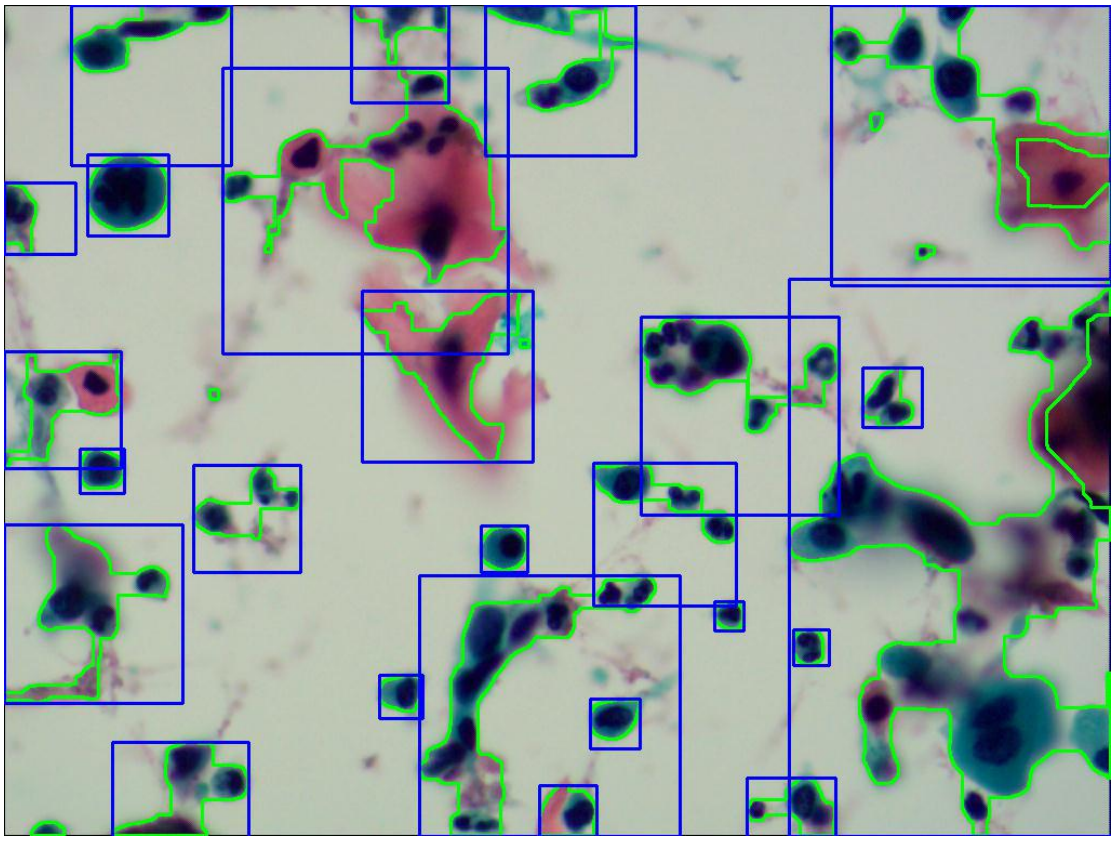


图 6

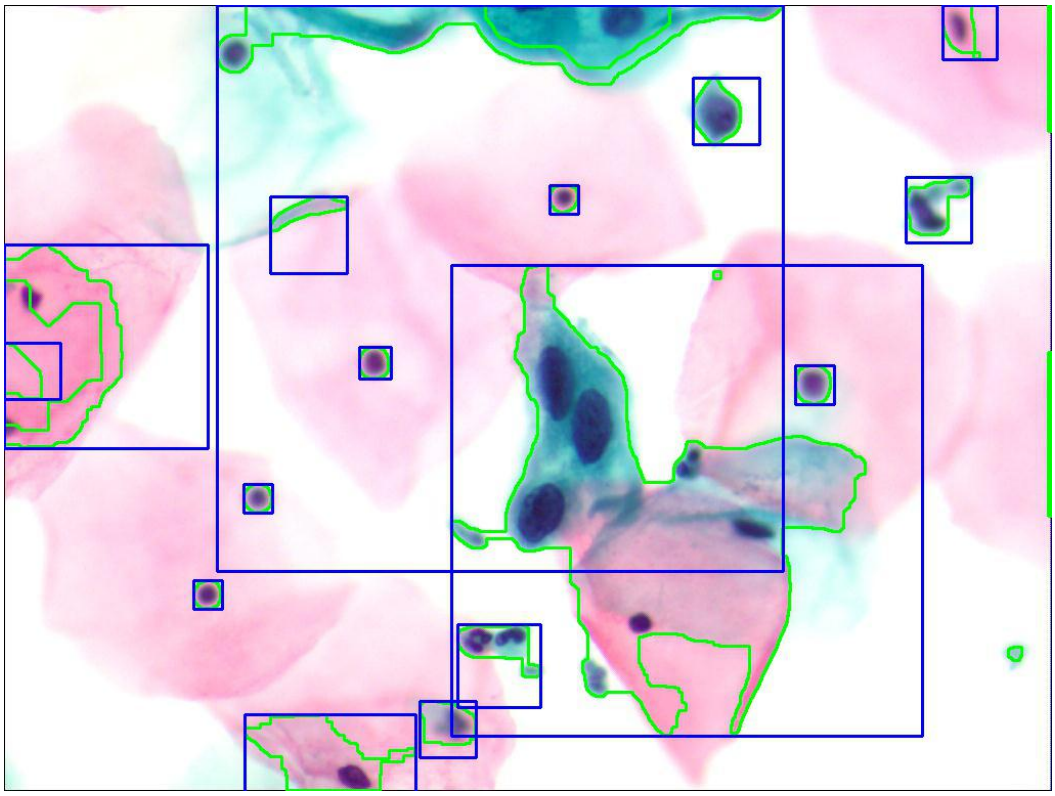


图 7

该算法的运行速度很快，也可以很好的将不重叠细胞体区域分离出来，但是对于重叠在一起的细胞体，该算法没有办法把他们分割出来。同时对于颜色过浅或者模糊的细胞体，往往也不能识别出细胞体，而只能识别出细胞核。

2. 校园卡卡号识别实践

通过结合二值化，图像匹配等方法，对湘潭大学校园卡卡号进行识别。

图 8

3. 图片的扫描

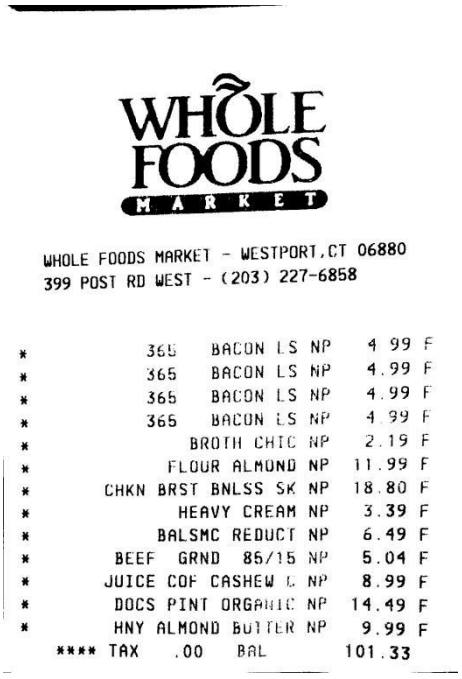


图 9

4. 全景图像拼接

图 10

5. 停车场车位识别

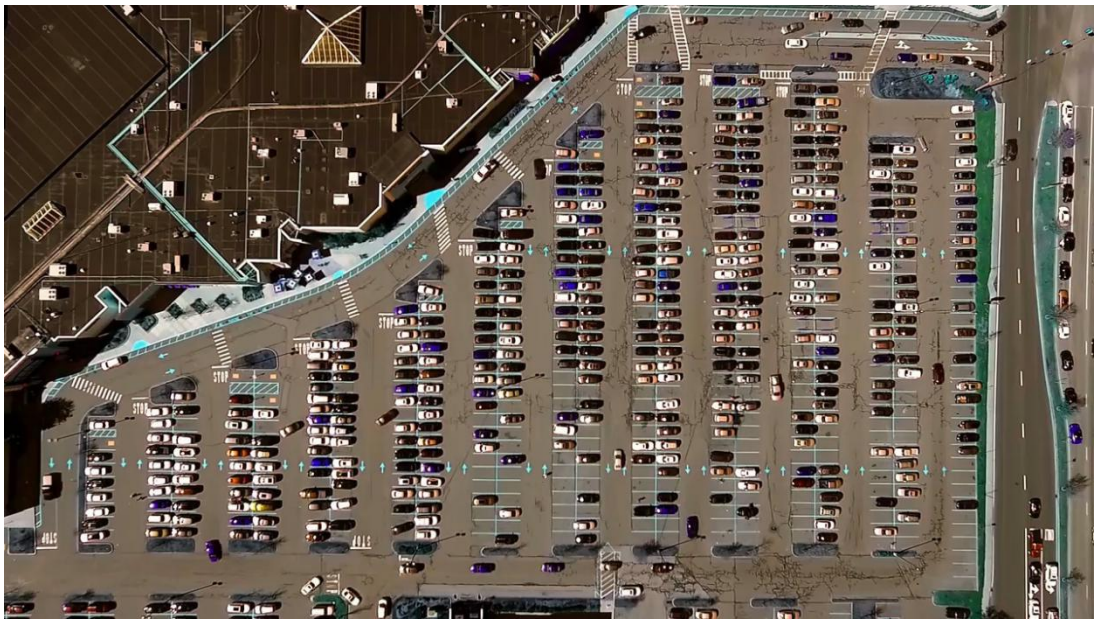


图 11

对停车场图片进行预处理，手动标记出边缘点，可以剔除非工作区域的影响。

图 12

图 13

利用地面的白线来确定每个车位，通过每列的长度与固定的车位宽度来找到每个车位。

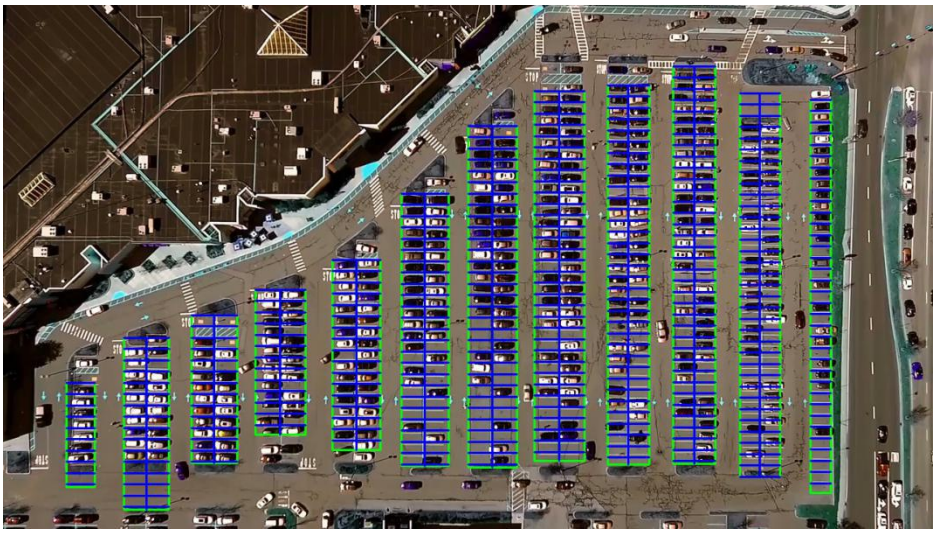


图 14

将每个车位作为数据放入训练好的神经网络，来判断该车位是否有车。用绿色标记出没有车的空车位。

图 13

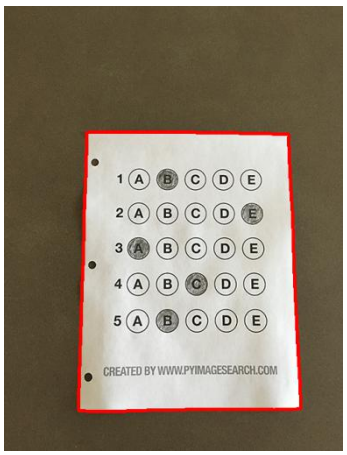
问题：1.该神经网络是利用数量为 384 和 164 的训练集与测试集训练出的 vgg16 模型，虽运用了数据增强，但是数量依旧较少，由上图也可看出，他在停车场图片中表现较差，还有许多位置的空车位未识别出来。2.车位位置的识别并不一定准确，有些不是车位的位置也被划

入车位区域。在车位的划分中出现了大量的手动调整，该代码只较好的适用于该停车场，对其他停车场车位识别效果较差。

我的目标：由于我对于 tensorflow 的学习还停留在较浅的区域，对于该网络的用 tensorflow 进行复现存在一定的困难，在我充分对神经网络进行学习后，再对该模型进行更好的优化。

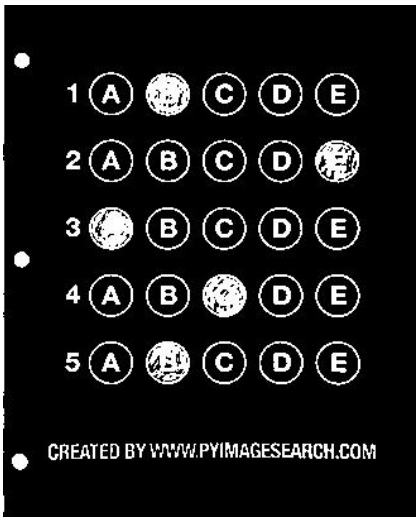
6. 答题卡识别

基础操作：高斯滤波+边缘检验+轮廓检验



透视变换（坐标，w/h，M）

Otsu 二值化处理，并找到所有轮廓



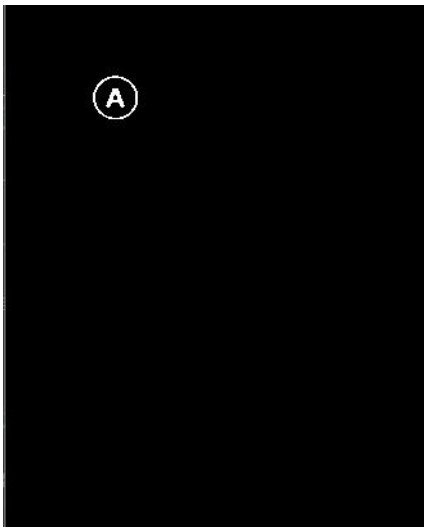
遍历轮廓并筛选：得到最小外接矩形，并根据 w/h 比例进行筛选，得到所有圆圈轮廓

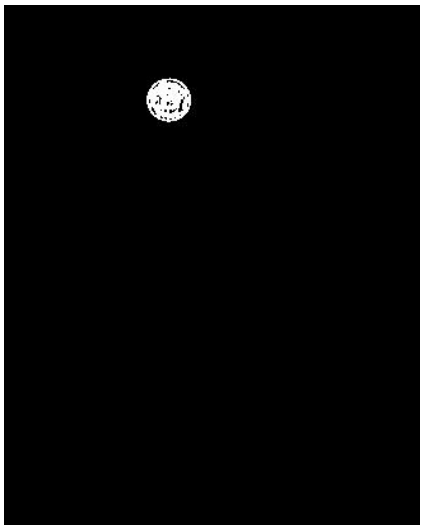
对轮廓排序：从上到下按题号分组

获得所选答案

（1）对轮廓排序：从左到右按选项排序，每个选项都有一个固定的位置

（2）通过计算非零点个数来判断该选项是否被选择（掩码+二值图）





（3）设定阈值，找出 0-4 五个位置上哪些选项被填充

获取正确答案，对比并判断正误，计算正确率

7. 光流估计

7.1 稀疏光流估计

图 14

复现代码存在的问题：只能捕捉第一帧就存在的特则点，对于视频后面出现的人，捕捉不到行动轨迹。同时，对于丢失的目标，该光流估计也很难捕捉。针对这个问题，每 5 帧对特征点进行一个刷新结合网络上查阅的资料，得到优化过后的光流估计。

图 15

7.2 稠密光流估计

图 16

8. 目标追踪

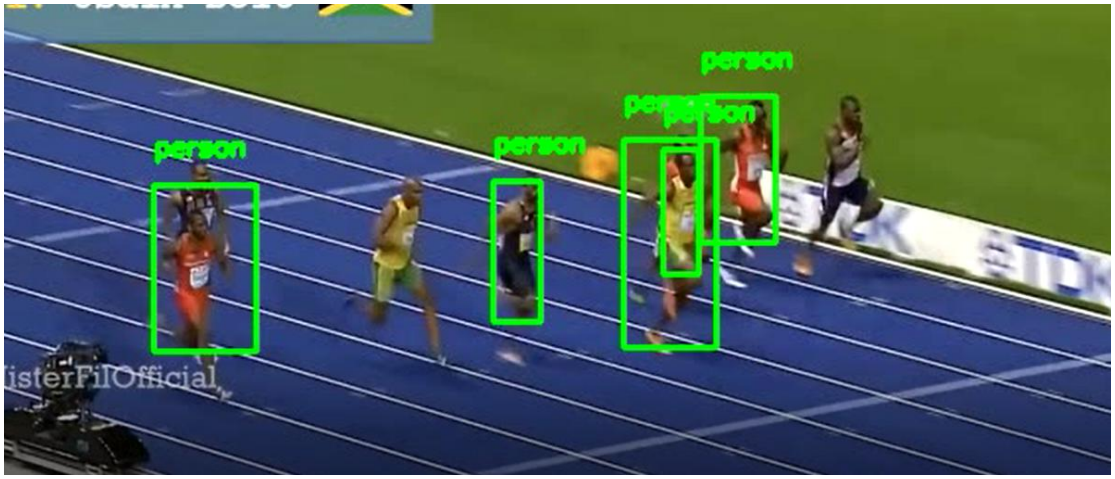


图 17

问题：自动追踪的效果并不好，存在目标遗漏以及追踪框无法跟踪的问题。

9. 人脸识别

图 18

通过 dlib 库进行两步操作

1 检测人脸

detector = dlib.get_frontal_face_detector()#获取人脸检测器

通过使用 detector 找到图中的人脸，再对每个人脸进行分别遍历。

2 识别五官

这里使用 dlib68 点特征检测数据，把照片中的人脸数据提取出来

predictor = dlib.shape_predictor(predictor_path)获取识别器

可以去官网上了解每个五官对应的特征点群来进行取用。

10. 疲劳检验

图 19

通过五官的识别找到人脸眼睛的位置，通过对眼睛的长宽比构成的变式来对图中人眨眼进行检测。