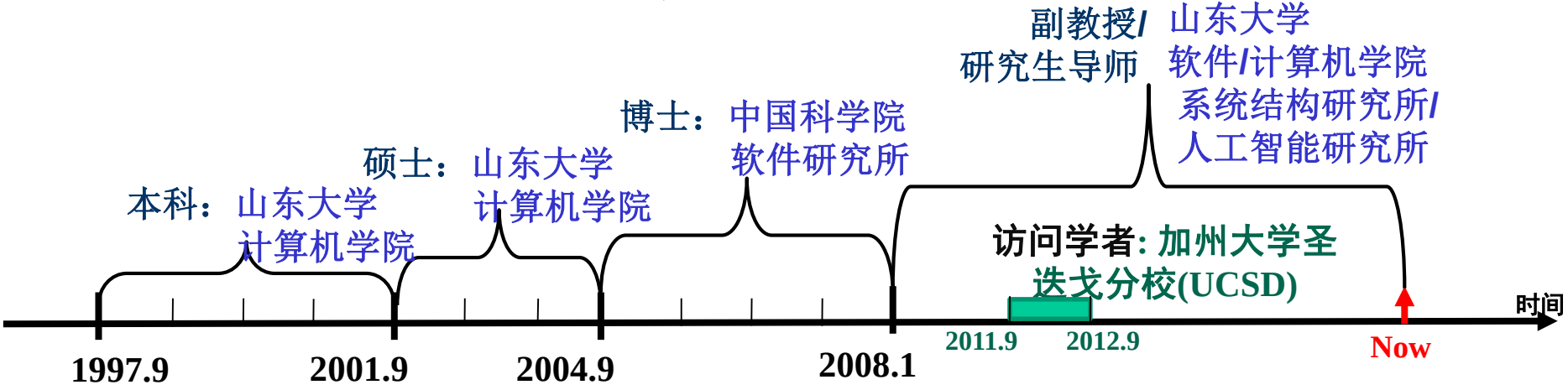
工业人工智能技术

基于OpenCV的级联分类器检测

李新

山东大学 软件学院

李新 个人简历



- 研究方向
  - 目标检测与跟踪
  - 无人机智能巡检
  - 大数据处理

邮箱：lx@sdu.edu.cn

电话：138-531-23559

OpenCV级联分类器

级联分类器CascadeClassifier：opencv下objdetect模块中用来做目标检测的级联分类器的一个类，它可以帮助我们检测例如车牌、眼睛、人脸等物体。它的大概原理就是判别某个物体是否属于某个分类。

以人脸为例，我们可以把眼睛、鼻子、眉毛、嘴巴等属性定义成一个分类器，如果检测到一个模型符合定义人脸的所有属性，那么就认为它是一个人脸

基于Haar特征的级联分类器

- 使用基于Haar特征的级联分类器的对象检测是Paul Viola和Michael Jones于2001年在其论文“Rapid Object Detection using a Boosted Cascade of Simple Features”提出的一种对象检测方法。这是一种基于机器学习的方法，其中从许多正负图像中训练级联函数，然后用于检测其他图像中的对象。
- Haar特征是一种反映图像的灰度变化的，像素分模块求差值的一种特征，包括：边缘特征、线性特征、中心特征和对角线特征

OpenCV自带训练好的人脸检测模型

Opencv自带训练好的人脸检测模型，存储在sources/data/haarcascades文件夹和sources/data/lbpcascades文件夹下，包括：

- 人脸检测器（默认）：haarcascade_frontalface_default.xml
- 人脸检测器（快速Harr）：haarcascade_frontalface_alt2.xml
- 人脸检测器（侧视）：haarcascade_profileface.xml
- 眼部检测器（左眼）：haarcascade_lefteye_2splits.xml
- 眼部检测器（右眼）：haarcascade_righteye_2splits.xml
- 嘴部检测器：haarcascade_mcs_mouth.xml
- 鼻子检测器：haarcascade_mcs_nose.xml
- 身体检测器：haarcascade_fullbody.xml
- 人脸检测器（快速LBP）：lbpcascade_frontalface.xml

1. CascadeClassifier()函数

```
#装入预训练模型
faceCascade =
cv2.CascadeClassifier(cv2.data.haarcascades+"haarcascade_frontalface_default.xml")
```

2. detectMultiScale()函数

detectMultiScale(image, scaleFactor, minNeighbors)进行检测

- **image:待处理的图像**
- **scaleFactor：检测框的最小尺寸**
- **minNeighbors：相当于检测的阈值，过小会出现误检现象，即把一些其他元素误判成人脸，过大可能会检测不到**
- **目标函数输出的是检测到目标图片中的每个人脸的x、y坐标值和宽度、高度**

样本图片人脸检测


Original img
WINNERS UEFA CHAMPIONS LEAGUE 2017/18
皇家马德里
Real Madrid CF
CHAMPIONS LEAGUE


img
WINNERS UEFA CHAMPIONS LEAGUE 2017/18
皇家马德里
Real Madrid CF
CHAMPIONS LEAGUE

实验代码

```
import cv2
#原图
img = cv2 .imread ("team1.jpg")
cv2.imshow("Original img",img)
#使用预训练摸型创建cascade分类器
faceCascade = cv2.CascadeClassifier(cv2.data.haarcascades
+"haarcascade_frontalface_default.xml")

#识别，将结果存储到faces变量中
faces = faceCascade.detectMultiScale(img, 1.1, 8)
for (x,y,w,h) in faces:
  #将结果绘制到原图中
    cv2.rectangle(img, (x,y),(x+w , y+h),(0,0,255),2)
cv2.imshow("img",img)
cv2.waitKey(0)
```

样本图片车辆检测

实验代码

```
import numpy as np
import cv2
import time

car_cascade = 'cascades/haarcascade_car.xml'
car_classifier = cv2.CascadeClassifier(car_cascade)
capture = cv2.VideoCapture('files/cars.avi')
cv2.namedWindow('Cars', cv2.WINDOW_NORMAL)

while capture.isOpened():
    response, frame = capture.read()
    if response:
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        cars = car_classifier.detectMultiScale(gray, 1.2, 3)
        for (x, y, w, h) in cars:
            cv2.rectangle(frame, (x,y), (x+w, y+h), (0,0,0), 3)
            cv2.imshow('Cars', frame)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    else:
        break

capture.release()
cv2.destroyAllWindows()
```

级联训练器特点

优点：

- 检测速度快，可以实现实时
- 对光线不敏感，强光弱光都可以检测到
- 远近都可识别出来，框的大小也是自适应的

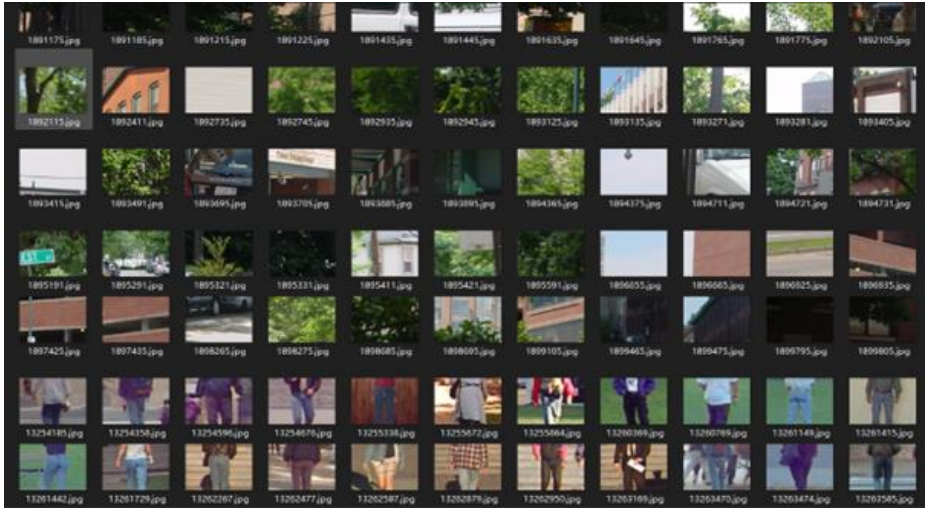
缺点：

- 侧脸识别不出来
- 如果人脸有部分被遮挡，也识别不出来

级联分类器训练——第一步准备样本

- 准备好自己需要检测物的正样本图像
- 准备好自己需要检测物的负样本图像

正样本为需要识别的车辆



负样本是除了车辆以外的物体，如树木、行人、路牌等

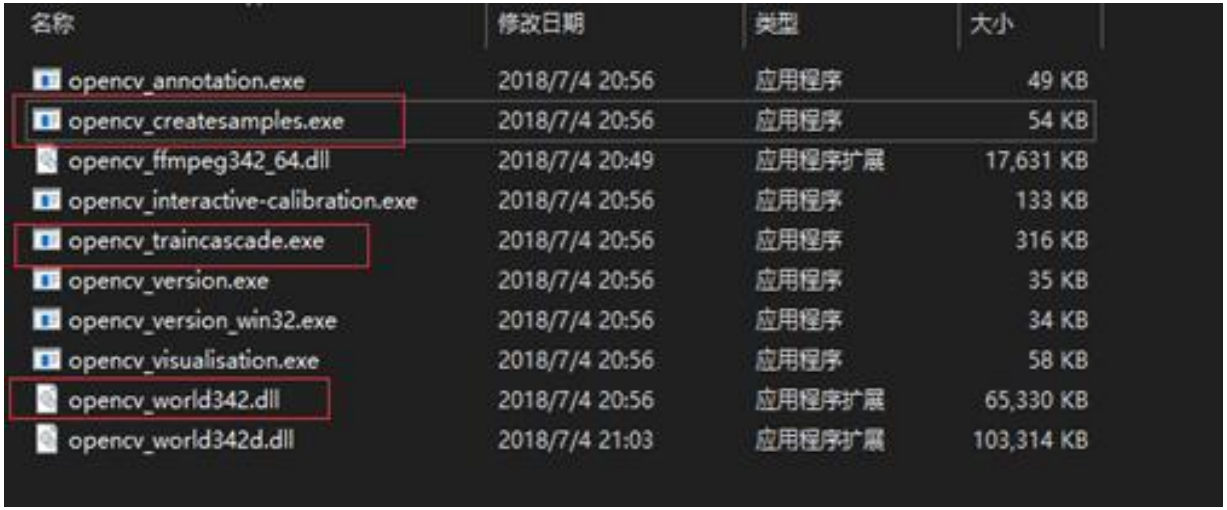
级联分类器训练——第二步找到训练程序

下载好OpenCV win10系统安装包，从安装包中的opencv\build\x64\vc15\bin 找到

opencv_createsamples.exe

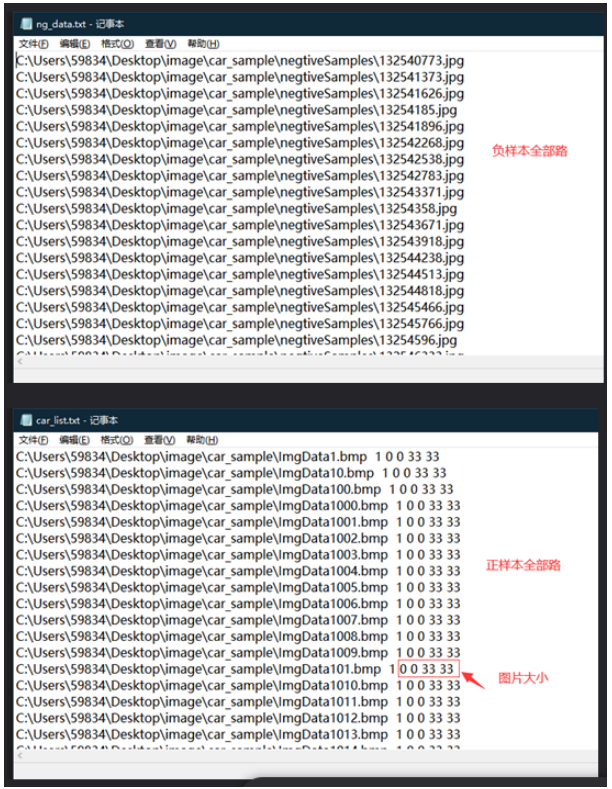
opencv_traincascade.exe

opencv_world342.dll



级联分类器训练——第三步创建样本目录

1.创建正负样本的图像路径的txt文件



级联分类器训练——第四步训练正样本

通过命令行执行命令进行样本采集生成

opencv_createsamples.exe -info car_list.txt -vec car_samples.vec -num 80 -w 33 -h 33

- **info字段填写正样本描述文件；**
- **vec用于保存制作的正样本；**
- **num制定正样本的数目；**
- **w和-h分别指定正样本的宽和高**

```
C:\Users\59834\Desktop\image\car_sample>opencv_createsamples.exe -info car_list.txt -vec car_samples.vec  -num 80 -w 33
 -h 33
Info file name: car_list.txt
Img file name: (NULL)
Vec file name: car_samples.vec
BG  file name: (NULL)
Num: 80
BG color: 0
BG threshold: 80
Invert: FALSE
Max intensity deviation: 40
Max x angle: 1.1
Max y angle: 1.1
Max z angle: 0.5
Show samples: FALSE
Width: 33
Height: 33
Max Scale: -1
RNG Seed: 12345
Create training samples from images collection...
Done. Created 80 samples
```

级联分类器训练——第五步训练负样本

2.通过命令行执行命令进行样本采集生成

opencv_traincascade.exe -data data -vec car_samples.vec -bg ng_data.txt -numPos 80 -numNeg 240 numStages 7 -w 33 -h 33 -minHitRate 0.995 -maxFalseAlarmRate 0.45 -mode ALL

- **data**：指定保存训练结果的文件夹；
- **vec**:指定正样本集；
- **bg**:指定负样本的描述文件夹；
- **numPos**：指定每一级参与训练的正样本的数目（要小于正样本总数）；
- **numNeg**:指定每一级参与训练的负样本的数目（可以大于负样本图片的总数）；
- **numStage**:训练的级数；
- **w**:正样本的宽；**h**:正样本的高；
- **minHitRate**:每一级需要达到的命中率（一般取值0.95-0.995）；
- **maxFalseAlarmRate**:每一级所允许的最大误检率；
- **mode**:使用Haar-like特征时使用，可选BASIC、CORE或者ALL；
- **featureType**:可选HAAR或LBP，默认为HAAR

```
C:\Users\59834\Desktop\image\car_sample>opencv_traincascade.exe -data data -vec car_samples.vec  -bg ng_data.txt  -numPo
s 80 -numNeg 240 -numStages 7 -w 33 -h 33
PARAMETERS:
cascadeDirName: data
vecFileName: car_samples.vec
bgFileName: ng_data.txt
numPos: 80
numNeg: 240
numStages: 7
precalcValBufSize[Mb] : 1024
precalcIdxBufSize[Mb] : 1024
acceptanceRatioBreakValue : -1
stageType: BOOST
featureType: HAAR
sampleWidth: 33
sampleHeight: 33
boostType: GAB
minHitRate: 0.995
maxFalseAlarmRate: 0.5
weightTrimRate: 0.95
maxDepth: 1
maxWeakCount: 100
mode: BASIC
Number of unique features given windowSize [33,33] : 576640

===== TRAINING 0-stage =====
<BEGIN
POS count : consumed   80 : 80
NEG count : acceptanceRatio    240 : 1
Precalculation time: 1.733
+----+---------+---------+
|  N |    HR   |    FA   |
+----+---------+---------+
|   1|        1|        1|
+----+---------+---------+
|   2|        1|        1|
+----+---------+---------+
|   3|        1|0.0458333|
+----+---------+---------+
END>
Training until now has taken 0 days 0 hours 0 minutes 5 seconds.
```

最终训练结果

cascade.xml	2021/1/9 15:06	XML 文档	19 KB
params.xml	2021/1/9 14:51	XML 文档	1 KB
stage0.xml	2021/1/9 14:51	XML 文档	1 KB
stage1.xml	2021/1/9 14:52	XML 文档	1 KB
stage2.xml	2021/1/9 14:52	XML 文档	1 KB
stage3.xml	2021/1/9 14:55	XML 文档	2 KB
stage4.xml	2021/1/9 15:01	XML 文档	1 KB
stage5.xml	2021/1/9 15:01	XML 文档	2 KB
stage6.xml	2021/1/9 15:02	XML 文档	1 KB
stage7.xml	2021/1/9 15:02	XML 文档	1 KB
stage8.xml	2021/1/9 15:02	XML 文档	1 KB
stage9.xml	2021/1/9 15:03	XML 文档	1 KB
stage10.xml	2021/1/9 15:04	XML 文档	2 KB
stage11.xml	2021/1/9 15:06	XML 文档	1 KB
stage12.xml	2021/1/9 15:26	XML 文档	1 KB
stage13.xml	2021/1/9 15:33	XML 文档	2 KB

谢 谢！