**实战三：手把手教你实现物体识别**

**一、基于Haad+Adaboost实现人脸识别**

**1.原理介绍（参考下面的博客文章）**

<http://www.cnblogs.com/ello/archive/2012/04/28/2475419.html>

Haar分类器 = Haar-like特征+积分图方法+AdaBoost+级联

Haar分类器算法的要点如下：

a.使用Haar-like特征做检测

b.使用积分图(Integral Image)对Haar-like特征求值进行加速

c.使用AdaBoost算法训练区分人脸和非人脸的强分类器

d.使用筛选式级联把强分类器级联到一起，提高准确率

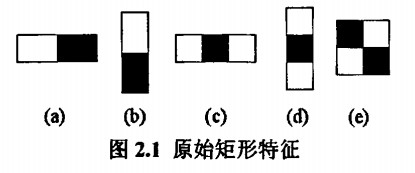
**（1）Haar-like特征是何方神圣呢？**

**a.什么是特征呢？**

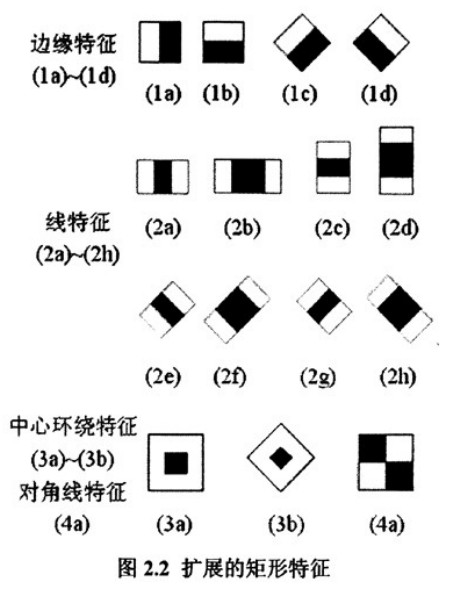
假设在人脸检测时我们需要有这么一个子窗口在待检测的图片窗口中不断的移位滑动，子窗口每到一个位置，就会计算出该区域的特征，然后用我们训练好的级联分类器对该特征进行筛选，一旦该特征通过了所有强分类器的筛选，则判定该区域为人脸。

**b.Haar-like特征的表示**

下面是Viola牛们提出的Haar-like特征。



下面是Lienhart等牛们提出的Haar-like特征。



这些所谓的特征不就是一堆堆带条纹的矩阵么，到底是干什么用的呢？

解释:将上面的任意一个矩形放到人脸区域上，然后，将白色区域的像素和减去黑色区域的像素和，得到的值暂且称之为人脸特征值，如果你把这个矩形放到一个非人脸区域，那么计算出的特征值和人脸特征值是不一样的，所以这些方块的目的就是把人脸特征量化，以区分人脸和费人脸。

为了增加区分度，可以对多个矩形特征计算得到一个区分度更大的特征值，那么什么样的矩形特征怎么样的组合到一块可以更好的区分出人脸和非人脸呢，这就是AdaBoost算法要做的事情了。

**（2）AdaBoost你给我如实道来！**

AdaBoost是一种具有一般性的分类器提升算法，它使用的分类器并不局限某一特定的算法。利用AdaBoost可以帮助我们选择更好的矩阵特征组合。

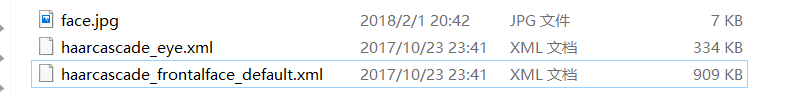
**（3）积分图是一个加速器**

积分图就是只遍历一次图像就可以求出图像中所有区域像素和的快速算法，大大的提高了图像特征值计算的效率。

**2.基于Haad+Adaboost实现人脸识别**

**（1）前期准备**

准备一张人脸照片face.jpg,并获取opencv自带的人脸检测器以及眼睛检测器模块。



**（2）整体步骤**

a.加载xml文件

b.加载jpg图片

c.jpg图片转灰度图

d.检测人脸

e.在图片中将人脸用矩形框起来

f.展示图片

**（3）程序源码**

import cv2

import numpy as np

# xml文件的引入

face\_xml = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')

eye\_xml = cv2.CascadeClassifier('haarcascade\_eye.xml')

# load jpg

img = cv2.imread('face.jpg')

cv2.imshow('src',img)

# haar gray

gray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)

# dect

# 1 人脸灰度图片数据 2 缩放系数 3 目标大小

faces = face\_xml.detectMultiScale(gray,1.3,5)

print('face=',len(faces))

# draw

for (x,y,w,h) in faces:

cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)

# 眼睛识别代码

roi\_gray = gray[y:y+h,x:x+w]

roi\_img = img[y:y+h,x:x+w]

eyes = eye\_xml.detectMultiScale(roi\_gray)

print('eyes=',len(eyes))

for (e\_x,e\_y,e\_w,e\_h) in eyes:

cv2.rectangle(roi\_img,(e\_x,e\_y),(e\_x+e\_w,e\_y+e\_h),(0,255,0),2)

cv2.imshow('dst',img)

cv2.waitKey(0)

**（4）结果展示**



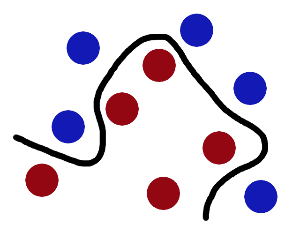
**二、基于SVM的男女识别**

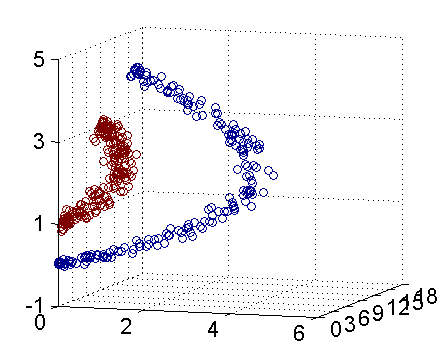
**1.原理**

还是先从原理入手，从网上看到的刘强西救爱人的故事：

<https://www.sohu.com/a/128747589_614807>

总之，上述故事的球称作[data]（数据源），棍子称作[classifier]（分类器），最大间隙trick称作[optimization]（最优化），拍桌子称作[kernelling](建立核函数)，那张纸称作[hyperplane](超平面)





**2.基于SVM的男女识别**

提供男女的身高和体重的数据，进行训练，然后再给出数据，预测男女。

有标签的训练称之为监督学习，样本分为负样本和正样本。

**（1）前期准备**

女生数据[155,48],[159,50],[164,53],[168,56],[172,60]

男生数据[152,53],[156,55],[160,56],[172,64],[176,65]

预测数据 女[167,55], 男[162,57]

标签 0女 1男

**（2）整体步骤**

a.准备data数据

b.准备label标签

c.训练SVM

d.数据的预测

**（3）源码**

import cv2

import numpy as np

import matplotlib.pyplot as plt

# 1 准备data

rand1 = np.array([[155,48],[159,50],[164,53],[168,56],[172,60]])

rand2 = np.array([[152,53],[156,55],[160,56],[172,64],[176,65]])

data = np.vstack((rand1,rand2))

data = np.array(data,dtype='float32')

# 2 label 标签

label = np.array([[0],[0],[0],[0],[0],[1],[1],[1],[1],[1]])

# 3 训练SVM

# m1 机器学习模块 SVM\_create()创建向量机

svm = cv2.ml.SVM\_create()

# 属性设置

svm.setType(cv2.ml.SVM\_C\_SVC) # svm type

svm.setKernel(cv2.ml.SVM\_LINEAR) # line 线性分类器

svm.setC(0.01)

# 训练

svm.train(data,cv2.ml.ROW\_SAMPLE,label)

# 4 完成数据的预测

pt\_data = np.vstack([[167,55],[162,57]]) # 0 女生 1 男生

pt\_data = np.array(pt\_data,dtype='float32')

# 开始预测

(par1,par2) = svm.predict(pt\_data)

print(par2)

**（4）结果展示**



**三、基于Hog+SVM实现小狮子的识别**

**1.Hog原理**

**参考文章**

<https://www.cnblogs.com/hustdc/p/6563891.html>

<https://blog.csdn.net/datase/article/details/71374471>

hog特征是histogram of gradient的缩写，我们观察图像时，信息更多来自目标边沿的突变。我们计算一块区域内的所有像素处的梯度信息，即突变的方向和大小，然后对360度进行划分，得到多个bin，统计该区域内所有像素所在的bin，就得到了一个histogram，就是说hog特征。

**2.基于Hog+SVM实现小狮子的识别**

**（1）前期准备**

样本的准备，正样本和负样本的准备，正样本是包含需要检测的物体，负样本是不包含需要检测的物体。

这里的样本数据提供一种方法，从视频中获取图片，直接给出代码。

# 视频分解成图片

# 1 load加载视频 2 读取info 3 解码 单帧视频parse 4 展示 imshow

import cv2

# 获取一个视频打开cap

cap = cv2.VideoCapture('1.mp4')

# 判断是否打开

isOpened = cap.isOpened

print(isOpened)

#帧率

fps = cap.get(cv2.CAP\_PROP\_FPS)

#宽度

width = int(cap.get(cv2.CAP\_PROP\_FRAME\_WIDTH))

#高度

height = int(cap.get(cv2.CAP\_PROP\_FRAME\_HEIGHT))

print(fps,width,height)

i = 0

while(isOpened):

if i == 100:

break

else:

i = i+1

(flag,frame) = cap.read() # 读取每一张 flag读取是否成功 frame内容

fileName = 'image'+ str(i) + '.jpg'

print(fileName)

if flag == True:

#写入图片

cv2.imwrite(fileName,frame,[cv2.IMWRITE\_JPEG\_QUALITY,100])

print('end!')

将视频生成的图片尺寸转换成64\*128的照片

# 540 \* 960 ==>64\*128

import cv2

for i in range(0,100):

fileName = 'imgs\\'+str(i+1)+'.jpg'

print(fileName)

img = cv2.imread(fileName)

imgInfo = img.shape

height = imgInfo[0]

width = imgInfo[1]

mode = imgInfo[2]

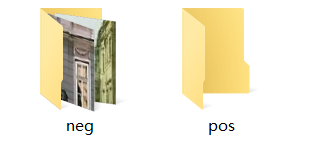
dstHeight = 128

dstWidth = 64

dst = cv2.resize(img,(dstWidth,dstHeight))

cv2.imwrite(fileName,dst)

具体的照片信息



**（2）总体步骤**

a.全局变量的赋值

b.hog对象的创建

c.SVM分类的创建

d.计算hog以及label标签

e.训练

f.检测  
**（3）代码**

# 1 准备样本 2 训练 3 预测

# 1 样本

# pos是正样本，包含需要检测的物体 neg是负样本，不包含需要检测的物体

# 图片大小是64\*128

# 如何获取样本？ 1 来源于网络 2 源自公司内部 3 自己收集

# 一个好的样本 远胜过一个复杂的神经网络

# 正样本：尽可能的多样 环境多样 干扰多样 820 负样本 1931 1:2或者1:3之间

# 2 训练

# 1.参数 2 hog 3.SVM 4.computer hog 5.label 6.train 7.predicr 8.draw

import cv2

import numpy as np

import matplotlib.pyplot as plt

#1 设置全局变量 在一个windows窗体中有105个block，每个block下有4个cell，每个cell下有9个bin，总共3780维

PosNum = 820 # 正样本的个数

NegNum = 1931 # 负样本的个数

winSize = (64,128) # 窗体大小

blockSize = (16,16) # 105个block

blockStride = (8,8) # block的步长

cellSize = (8,8)

nBin = 9

#2 hog对象的创建

hog = cv2.HOGDescriptor(winSize,blockSize,blockStride,cellSize,nBin)

#3 SVM分类器的创建

svm = cv2.ml.SVM\_create()

#4 5 计算hog label

featureNum = int(((128-16)/8+1)\*((64-16)/8+1)\*4\*9) # 3780

featureArray = np.zeros(((PosNum+NegNum),featureNum),np.float32)

labelArray = np.zeros(((PosNum+NegNum),1),np.int32)

# SVM 监督学习 样本和标签 SVM学习的是image的hog特征

for i in range(0,PosNum):

fileName = 'pos\\'+str(i+1)+'.jpg'

img = cv2.imread(fileName)

hist = hog.compute(img,(8,8)) # 当前hog的计算 3780维的数据

for j in range(0,featureNum):

featureArray[i,j] = hist[j] # hog特征的装载

labelArray[i,0] = 1 # 正样本标签为1

for i in range(0,NegNum):

fileName = 'neg\\'+str(i+1)+'.jpg'

hist = hog.compute(img,(8,8))

for j in range(0,featureNum):

featureArray[i+PosNum,j] = hist[j]

labelArray[i+PosNum,0] = -1 # 负样本标签为-1

# 设置SVM的属性

svm.setType(cv2.ml.SVM\_C\_SVC)

svm.setKernel(cv2.ml.SVM\_LINEAR)

svm.setC(0.01)

# 6 train

ret = svm.train(featureArray,cv2.ml.ROW\_SAMPLE,labelArray)

# 7 检测（创建myHog--->myDect参数得到-->来源于resultArray（公式得到） rho(训练得到)）

# rho

alpha = np.zeros((1),np.float32)

rho = svm.getDecisionFunction(0,alpha)

print(rho)

print(alpha)

# resultArray

alphaArray = np.zeros((1,1),np.float32)

supportVArray = np.zeros((1,featureNum),np.float32)

resultArray = np.zeros((1,featureNum),np.float32)

alphaArray[0,0] = alpha

resultArray = -1\*alphaArray\*supportVArray #计算公式

# mydect参数

myDetect = np.zeros((3781),np.float32)

for i in range(0,3780):

myDetect[i] = resultArray[0,i]

myDetect[3780] = rho[0]

# 构建好myHog

myHog = cv2.HOGDescriptor()

myHog.setSVMDetector(myDetect)

# 待检测图片的加载

imageSrc = cv2.imread('Test.jpg',1)

# 检测小狮子 (8,8)winds的滑动步长 1.05 缩放系数 (32,32)窗口大小

objs = myHog.detectMultiScale(imageSrc,0,(8,8),(32,32),1.05,2)

# 起始位置、宽和高 objs三维信息

x = int(objs[0][0][0])

y = int(objs[0][0][1])

w = int(objs[0][0][2])

h = int(objs[0][0][3])

# 目标的绘制 图片 起始位置 终止位置 颜色

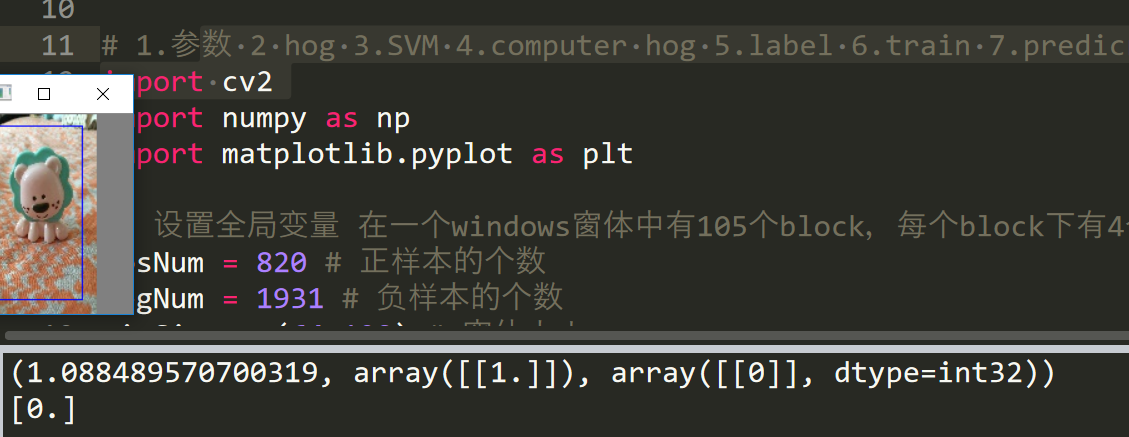
cv2.rectangle(imageSrc,(x,y),(x+w,y+h),(255,0,0))

# 目标的展示

cv2.imshow('dst',imageSrc)

cv2.waitKey(0)

（4）结果展示



**四、项目地址**

链接：https://pan.baidu.com/s/14O4NbkxZ3G9AWMOtyX9xXg

提取码：24hb