人脸识别程序优化与训练

一、引言

随着互联网的发展，慕课与线上教学越来越成为老师们的教学选择，但是在进行线上教学与慕课时，老师往往不能时时的关注学生的上课状态，导致教学质量与线下教学相比有所下降。人工智能的发展与人脸表情识别的程序的出现，可以帮助老师在课堂上时时检测学生们的学习状态，为老师根据学生们学习状态调整教学方式提供了有力的技术支持。

本次实验在研究了基于 haar 特征图的人脸识别程序的原理之后，进行了课堂环境下的人脸识别，并通过调整人脸识别程序的五个基本参数使得人脸识别率达到了极限，但是此时人脸识别程序依然存在错误识别以及识别率较低的现象，特别是在大教室，多人脸的情况，识别情况非常复杂，为此，实验组进行了人脸识别训练程序的探究，希望通过训练课堂环境下人脸样本以求得到较高的人脸识别率来满足教师对学生课堂状态把握的需求。

二、级联分类器结构探究

级联分类器是基于 haar 特征图人脸识别程序的重要组成结构，人脸识别程序的训练是由级联分类器以及 Adaboost 算法共同完成，理解级联分类器的基本结构有助于之后的人脸识别程序的训练。

级联分类器基本结构组成为：强分类器，弱分类器与积分图像，首先级联分类器是由多个强分类器串联构成(如图 1)，

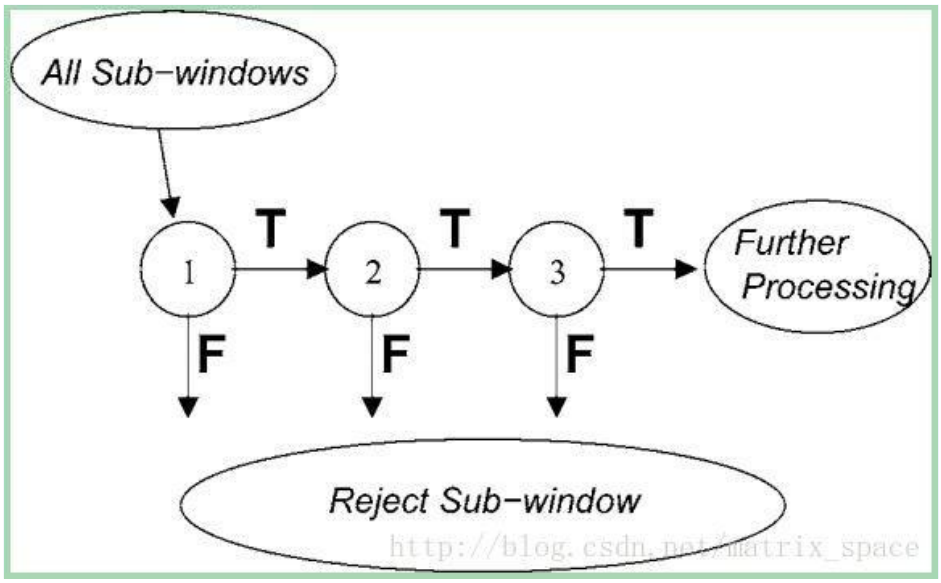

图 1：级联分类器与强分类器关系图

图 1 展示的是级联分类器与强分类器的关系图，在进行人脸识别时。只有全部的强分类器通过人脸检测，才可最终判定该区域为人脸区域。将强分类器串联起来的意义在于可以分摊误差，使得每个强分类器可以在较高的误差的前提下，整个级联分类器的整体误差则显得很低。强分类器是由三个或多个弱分类器线性组合而成，每个弱分类器的线性组合系数由 Adaboost 算法和通过率共同决定，因训练样本的不同而不同，而弱分类器是由积分图像与 haar 特征图构

成，每个弱分类器至少代表一个积分图像，积分图像的定义为该点左上方区域内所有点的像素和,如图 2 所示：

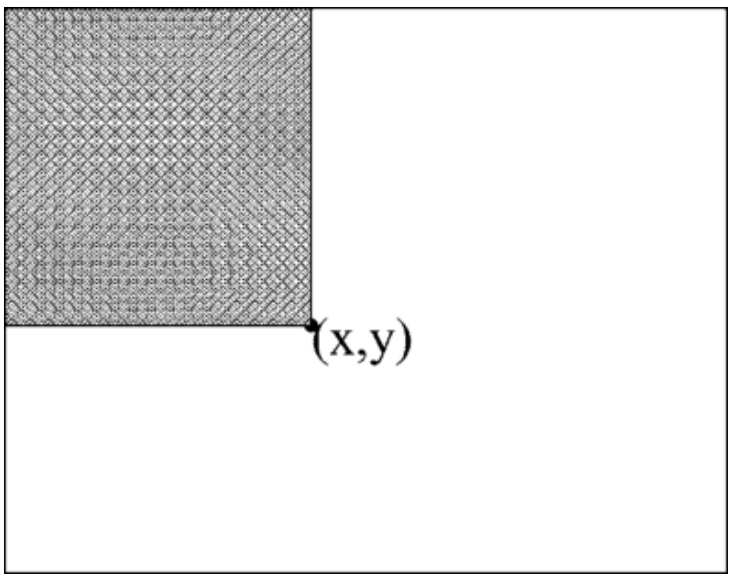

图 2：积分图像展示图

图 2 中点(x,y)的积分图像值为图中积分区域内像素和，通过这样的定义可以得到所描述的每一个矩形区域内的像素和，得到的像素和与 haar 特征图进行比对，在一定范围内符合 haar 特征图的区域为人脸的高概率区域，这样该区域可以通过不同的 haar 特征图的比对，最终确定该区域是否为人脸区域。经过研究得到，实验中所使用的级联分类器级数为 38 级，其中前七级还有的特征数分别为 2->10->25->25->50->50->50，后面 31 级的特征数由训练样本与规定的通过率相关。

三、人脸识别程序参数意义

在人脸识别程序中共有四个重要参数，分别为最大可检测对象(MaxSize)，最小可检测对象(MinSize)，多尺寸缩放因子(ScaleFactor)，检测阈值(minneighbors)，这四个参数分别代表的含义为：

1.MinSize(最小可检测对象的大小):

最小可检测对象的大小，指定为两个元素向量 [height weight]。对于包含对象的最小大小区域，以像素为单位设置此属性。该值必须大于或等于用于训练模型的图像大小。当处理图像之前知道最小对象大小时，使用此属性可以减少计算时间。如果不指定此属性的值，探测器会将其设置为用于训练分类模型的图像的大小。

2. MaxSize(最大可检测对象的大小):

最大可检测对象的大小，指定为两个元素向量 [height weight]。指定要检测的最大对象的大小（以像素为单位)。当处理图像之前知道最大对象大小时，使用此属性可以减少计算时间。如果不为此属性指定值，探测器会将其指定为 size(IMG)。

3.ScaleFactor(用于多尺度对象检测的缩放):

用于多尺度对象检测的缩放，指定为大于 1.0001 的值。指定用于递增缩放检测

的因子在 MinSize 和 MaxSize 之间缩放。

4. minneighbors（检测阈值）：

检测阈值，指定为整数。阈值定义在对象周围有多个检测的区域中声明最终检测所需的条件。合并满足阈值的共置检测组，

以在目标对象周围生成一个边界框。通过要求在多尺度检测阶段多次检测目标对象，提高此阈值可能有助于抑制错误检测。将此属性设置为 0，将返回所有检测，而不执行阈值或合并操作。此属性是可调整的。

通过对以上四个参数进行调整并运行人脸识别程序进行人脸识别率的对比，经过整理得到下表：

表 1：ScaleFactor 为 1.1 时 minneighbors 对识别率的影响

minneighbors	人脸识别数	错误识别数	有效识别数	低头数	识别率
3	24	7	17	5	0.773
4	18	4	14	5	0.636
5	17	3	14	5	0.636
6	16	2	14	5	0.636

表 2：minneighbors 为 5 时 ScaleFactor 对识别率的影响

scaleFactor	人脸识别数	错误识别数	有效识别数	低头数	识别率
1.1	24	7	17	5	0.773
1.2	13	1	12	5	0.545
1.3	12	0	12	5	0.545

通过表 1 表 2 数据的研究发现，随着多尺寸缩放因子的增加，人脸识别数减小，人脸识别率也逐渐减小；随着检测阈值的增加，人脸识别数减小，人脸识别率也随之减小。将数据进行整理，做出多尺度缩放因子随人脸识别率，错误识别率的关系曲线如下：

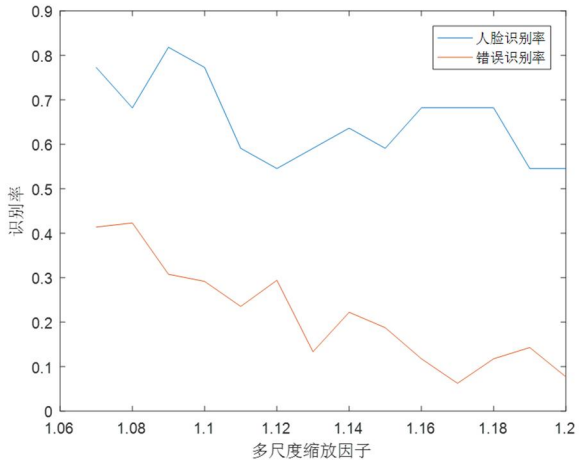

图 3：多尺度缩放因子与人脸识别率，错误发生率的关系曲线

图 3 中蓝色曲线是多尺度缩放因子随人脸识别率的变化曲线，橙色曲线表示多尺度缩放因子随错误率的变化曲线，从图中可以看到随着多尺度缩放因子的增加，人脸识别率与错误率程波浪式下降，通过不断的实验，最终得到最佳的参数为：$ScaleFactor=1.09$，$minneighbors=3$，此时人脸识别率为 81.8%，错误识别数为 10。这样的识别率与错误识别数无法满足教师上课需求，因此训练课堂环境下人脸识别程序就显得尤为重要。

四、人脸识别训练程序初步

在研究了基于 haar 特征图的人脸识别程序对课堂环境下人脸识别存在较大误差，较小识别率的问题后，实验组决定进行训练课堂环境下的样本来提高人脸识别程序的识别率，降低错误率，以达到可以指导教师教学的水平。已知的人脸识别程序训练需要以下步骤：首先进行人脸检测程序来检测未经训练前的人脸识别率，同时获得正样本人脸图片与负样本非人脸图片，第二步人脸数据收集程序，将第一步检测到的人脸以大头贴的形式存储在一个文件夹中，且所得到的图片为黑白图片，下图为人脸数据收集程序的训练结果：

图 4：人脸数据收集程序结果图

图 4 中收集到人脸黑白大头贴将作为进行人脸比对的重要训练样本，第三步为人脸训练程序，将第二步得到的人脸大头贴进行训练最终得到一个 yml 文件，第四步为人脸识别程序，是将得到的 yml 文件放在程序开头进行人脸识别，最终可以得到每一张人脸的识别相似度与识别率。

图 5：人脸识别程序运行结果图

图 5 展示的是人脸检测程序经过训练得到的人脸识别程序比对图，图中绿色的框代表程序认为是人脸框的区域，绿色框中的下方有人脸比对的相似度，相似度最高为 55%，后经过增大人脸数据收集程序的训练样本(由 3912 张图片增至 6000 张)来进行比对，得到结果图中人脸相似度仅提高了 1%，所以若要提高人脸的比对率则需要修改人脸数据收集程序使之可以在训练样本中得到更多的人脸大头贴，进而满足的相似度比对的需要。

五、实验总结

1.基于 haar 特征的人脸检测程序在课堂环境下的人脸检测率最高达到 81.8%，此时所设定的参数：多尺度缩放因子为 1.09，检测阈值设为 3，此时人脸识别人数可达到 18 人，错误识别数为 10 个。

2.虽然调整人脸检测程序的参数可以提高人脸检测率，但是错误率也会提高，因此若要进一步提高人脸检测率，降低错误率，可考虑调整训练样本，训练课堂环境下的人脸识别样本，将基于 haar 特征人脸识别程序得到错误识别作为训练的负样本进行训练，进而提高人脸识别程序的人脸识别率，降低错误发生率。

3.最终通过训练的人脸识别程序的人脸比对率并不高，可能的原因是训练样本的数量过低，人脸数据收集程序对人脸照片的收集效率不高，可考虑对人脸数据收集程序进行修改优化，或者提高训练样本的数量。

附录

所用软件为 python
1 人脸数据收集程序：

```
import cv2
import os
face_detector = cv2.CascadeClassifier(r'F:\Lib\site-packages\cv2\data\haarcascade_frontalface_default.xml')
face_id = input('\n enter user id:')
count = 0
# 循环读取图片
for filename in os.listdir(r"D:\物理学与信息技术学院\人工智能\image1"):
    print ( filename)
    # 读取图像
    img = cv2.imread(filename)
    if img is None:                    #判断读入的 img 是否为空，为空就继续下一轮循环
        continue
    # 转为灰度图片
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    # 检测人脸
    faces = face_detector.detectMultiScale(gray, 1.3, 5)
    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x+w, y+w), (255, 0, 0))
        count += 1
    # 保存图像
    cv2.imwrite("Facedata/User." + str(face_id) + '.' + str(count) + '.jpg', gray[y: y + h, x: x + w])
    cv2.imshow('image', img)
```

2.人脸数据训练程序

```
import numpy as np
from PIL import Image
import os
import cv2
# 人脸数据路径
path = 'Facedata'
recognizer = cv2.face.LBPHFaceRecognizer_create()
detector = cv2.CascadeClassifier(r'F:\Lib\site-packages\cv2\data\haarcascade_frontalface_default.xml')
def getImagesAndLabels(path):
    imagePaths = [os.path.join(path, f) for f in os.listdir(path)]  # join 函数的作用？
    faceSamples = []
    ids = []
    for imagePath in imagePaths:
        PIL_img = Image.open(imagePath).convert('L')    # convert it to grayscale
```

```
            img_numpy = np.array(PIL_img, 'uint8')
            id = int(os.path.split(imagePath)[-1].split(".")[1])
            faces = detector.detectMultiScale(img_numpy)
            for (x, y, w, h) in faces:
                faceSamples.append(img_numpy[y:y + h, x: x + w])
                ids.append(id)
    return faceSamples, ids
print('Training faces. It will take a few seconds. Wait ...')
faces, ids = getImagesAndLabels(path)
recognizer.train(faces, np.array(ids))
recognizer.write(r'face_trainer\trainer4.yml')
print("{0} faces trained. Exiting Program".format(len(np.unique(ids))))
```

4.人脸识别程序

```
import cv2
recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read('face_trainer/trainer4.yml')
cascadePath = r'F:\Lib\site-packages\cv2\data\haarcascade_frontalface_default.xml'
faceCascade = cv2.CascadeClassifier(cascadePath)
font = cv2.FONT_HERSHEY_SIMPLEX
idnum = 0
names = ['Allen', 'Bob','boy']
img = cv2.imread('869.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
faces = faceCascade.detectMultiScale(
            gray,
            scaleFactor=1.1,
            minNeighbors=3,
            minSize=(50, 50),maxSize=(145,145)
)

for (x, y, w, h) in faces:
    cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)
    idnum, confidence = recognizer.predict(gray[y:y+h, x:x+w])

    if confidence < 100:
        idnum = names[idnum]
        confidence = "{0}%".format(round(100 - confidence))
    else:
        idnum = "unknown"
        confidence = "{0}%".format(round(100 - confidence))

    cv2.putText(img, str(idnum), (x+5, y-5), font, 1, (0, 0, 255), 1)
    cv2.putText(img, str(confidence), (x+5, y+h-5), font, 1, (0, 0, 0), 1)
res=cv2.resize(img,(1500,1200),interpolation=cv2.INTER_CUBIC)
cv2.imshow('iker1',res)
cv2.imshow('img', img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```