基于 PCA 的 MATLAB 人脸识别及分析

李　想　　黄家辉

摘要

人脸识别，是基于人的脸部特征信息进行身份识别的一种生物识别技术。用摄像机或摄像头采集含有人脸的图像或视频流，并自动在图像中检测和跟踪人脸，进而对检测到的人脸进行脸部识别的一系列相关技术，通常也叫做人像识别、面部识别。本文利用 MATLAB 软件，结合 PCA（主成分分析）方法，进行对人脸的识别及分析。针对问题一，首先利用包含测试集的库数据集构造一个人脸模型，然后将测试集与训练集进行匹配，找到与之对应的训练集头像。在其中利用 PCA 将库数据集和测试集进行降维，提升了计算速度。针对问题二，首先利用神经网络方法检测人脸图像中是否有眼镜；然后，利用二值化和数学形态学方法定位眼镜边框针对问题三，基于问题二的方法，再利用图像中面部像素的分布规律修复眼镜边框遮挡位置的图像，从而达到去除眼镜的目的。[a.pdf]

关键词：**MATLAB　PCA**（主成分分析）　人脸识别

1　问题重述

人脸识别技术是基于人的脸部特征，对输入的人脸图像或者视频流. 首先判断其是否存在人脸, 如果存在人脸，则进一步的给出每个脸的位置、大小和各个主要面部器官的位置信息。并依据这些信息，进一步提取每个人脸中所蕴涵的身份特征，并将其与已知的人脸进行对比，从而识别每个人脸的身份。给定一些戴眼镜和不戴眼镜的照片，试建立合适的模型完成以下问题：

1、读取图像，选取适当的模型识别人类脸部特征，比如眼睛部位、嘴巴部位等；

2、将所给文件中戴眼镜和不戴眼镜的前 100 个（A001-A100，B001-B100）作为测试样本，选取适当的特征建立数学模型，判断该照片中的人物是否配带眼镜，并对剩余的样本进行自动识别，给出识别结果及识别正确率。

3、建立数学模型，尝试对附件 3 戴眼镜正面人物图像，去掉其眼镜的正面人物图像，并选择至少 5 个附件 1 中的图像进行试验。

2　模型的假设和符号的规定

2.1　模型的假设

1. 图片大小（像素）不一致时，均调整为像素 236Œ315。

3　问题分析

3.1　问题一的分析

附件所给图片像素不一致，故在按照矩阵的方式进行读取信息时，要将每张图片的大小进行调整，固定为像素 236Œ315。之后，每个人的脸都有其特殊性，所以在所给的数据中，利用欧式距离进行识别每张人的脸。

3.2　问题二的分析

眼镜检测问题目前研究较少，首先需要对输入的图像进行人脸检测，基于人脸检测结果再进一步检测图像中人脸是否佩戴有眼镜。对于眼镜的检测，可利用眼睛与周围像素的对比明显来判断是否有佩戴眼镜，并可利用神经网络来检测。

3.3　问题三的分析

在问题二检测到眼睛的基础上，确定眼镜的位置，并去除眼睛，修复边框遮挡处的图像信息，最终获得眼镜去除后的人脸图像。根据参考文献，可采用二值化处理和数学形态学方法定位眼镜边框。考虑到同一幅图像中光照环境固定，本文利用边框局部面部像素进行插值的方法修复眼镜边框遮挡位置的图像。

4　问题一模型的建立与求解

4.1　将所给附件一戴眼镜图片信息录入获取矩阵

以附件中图片的像素信息为基准，设置行列大小固定的矩阵，因为附件中的图片像素多为 236Œ315，故将大小不一致的图片，进行调整大小以便后续操作。提前进行的预分配数据可以加速数据读取，矩阵的行数是库数据的个数，列数是图片的维度。

4.2　利用 PCA 对所的数据进行处理

矩阵第一列与另一矩阵的差值小于 〇 是即认为两组矩阵所对应的两张图片可进行第一张矩阵对应的图片左侧可与第二个矩阵所对应的图片右侧进行拼接

2.1 特征脸的计算：

5　问题二模型的建立与求解

5.1　基于神经网络的离线训练和在线检测

考虑到眼镜边框与周围像素对比度明显，本文提取眼睛区域的边缘特征，并基于该特征提出一种基于神经网络的眼镜检测方法，基本流程分为离线训练和在线检测两部分

离线训练过程首先采集佩戴眼镜和无眼镜的人脸图像，形成有标签的人脸图像数据，然后对人脸图像进行边缘检测和特征提取，基于提取的数据特征训练神经网络，最终获得眼镜检测的神经网络模型。

在线检测过程为：对输入的人脸图像首先进行边缘检测和特征提取，然后利用已训练好的模型进行计算，最终判断输入的人脸图像是否佩戴眼镜。

5.1.1 特征提取

5.2 特征提取

观察人脸图像的边缘图像可获知两个事实：（1）通常眼睛区域位于人脸图像的上半部分；（2）眼镜边框的边缘信息较丰富。基于以上两点，本文方法在人脸图像的边缘图的基础上提取了 7 维特征，具体特征提取方法为：（1）在边缘图中提取眼睛先验区域。假设人脸图像的左上角坐标为（0，0），人脸图像的高度为 h ，宽度为 w ，则眼睛先验区域左上角位于图像的 (w/4h/4) 处，该区域的宽度为 w/2 ，高度为 h/4 。示意图如图 4（a）所示。（2）利用自适应阈值的方法对提取的眼睛先验区域边缘图进行二值化处理。（3）计算眼睛先验区域边缘点的比例，作为特征 1。（4）参照示意图 4（b）将眼睛先验区域划分为 6 个子区域。（5）计算每个子区域的边缘点比例，作为特征 2 7。至此共提取了能够表达眼睛周围区域边缘情况的 7 维特征，这些特征将用于模型的训练和眼镜检测的在线判别。

6 问题三模型的建立与求解

6.1 眼镜边框定位方法

确认输入的人脸图像中有眼镜后需要定位眼镜边框，然后再修复边框遮挡处的图像。本文主要采用二值化处理和数学形态学方法定位眼镜边框。具体方法为：（1）针对输入的有眼镜的原始人脸图像的边缘图像 E ，采用最大类间差的方法进行二值化处理，得到二值图像 IB 。（2）在二值化的基础上采用数学形态学的开运算消除部分未连接的较小图像边缘，然后利用连通域的方法去除孤立的边缘点。（3）随后根据边缘连通域的位置去除重心处于图像高度 1/2 以下的边缘，也即去除了嘴巴和鼻子区域的边缘，得到去噪图像 IC 。（4）采用 2 次迭代闭运算的方法连接剩余边缘。然后横向扫描图像，将边缘断裂长度低于 1/6 图像宽度的边缘进行连接，形成眼镜边框掩膜图像 IM ，完成眼镜边框定位。

6.2 眼镜边框图像修复方法

眼镜边框定位后，如何修复边框遮挡处的面部图像信息，使边框位置图像显示更加自然，是一个具有挑战性的问题。考虑到同一幅图像中光照环境固定，本文利用边框局部面部像素进行插值的方法修复眼镜边框遮挡位置的图像。该方法的主要特点是先基于输入的彩色图像建立肤色模型，然后利用肤色检测方法获取眼镜边框周围面部的像素点，再利用这些像素点对边框位置进行线性插值，最后对处理完的图像进行中值滤波，最终获得去除眼镜边框后的图像。具体流程框图如图 6 所示。

6.3 在线肤色建模

当前肤色检测方法研究已较为成熟，从肤色模型中是否使用参数方面考虑可划分为参数模型方法和非参数模型方法两类。参数模型方法主要包括高斯模型、混合高斯模型、椭圆模型等，非参数模型方法包括贝叶斯方法、SOM 方法、数据挖掘方法、神经网络方法等 [15-16] 。虽然不同人种、不同光照下肤色分布具有聚集性特点，但同一幅图像中同一人的肤色分布更为集中。因此，本文选取人脸图像中面部的部分像素，在线建立有参的高斯肤色模型，用于后续

面部像素点的检测。高斯肤色模型的建立方法参见文献 [15]，本文不再赘述。面部像素点的选取是本文建立高斯肤色模型的关键，此处选取左右眼镜底部边框位置下方的部分像素作为面部像素。具体面部肤色点选取过程如下：（1）针对眼镜边框掩模图像，将眼镜边框进行纵向投影，定位眼镜边框左右边界。（2）根据眼镜边框左右边界将边框掩模图像从中间划分为左右两部分，并分别对左右两部分进行横向投影，定位左右眼镜边框上下边界。（3）分别在左右眼镜边框下方 5 个像素外，靠左右眼镜中心线外侧取宽度为左右眼镜宽度的 1/3，高度为面部图像高度 1/6 的矩形框内的像素为面部像素，

6.4 眼镜去除效果

在对眼镜去除效果进行测试时，本文方法与基于 PCA 的方法进行了对比。基于 PCA 方法的实现参考了文献 [2]，由于训练集和诸如人脸检测、眼镜定位等步骤实现细节不同，本文 PCA 方法的结果并不能等同于文献 [2] 实验结果。本文实现的基于 PCA 的方法，训练时采用了 CAS-PEAL 人脸数据库中 100 幅正面无眼镜人脸图像，并且通过眼睛中心点进行人脸对齐和眼睛局部区域裁剪。实验过程中人脸检测步骤采用了 OpenCV 中实现的类 Haar 特征的方法，眼睛中心位置则人工进行标定。图 9 示出了本文方法与基于 PCA 方法处理的样图，其中 PCA 方法是基于灰度图像进行训练和去除眼镜的。从图 9 中可以看出本文方法对于黑色眼镜边框的图像处理效果明显好于基于 PCA 的方法，处理后的眼镜边框残留更少，图像更加自然。然而对于浅颜色边框的人脸图像本文处理效果较差，究其原因是眼镜边框颜色过浅时，难以准确定位边框位置。

7 模型的优缺点及改进方向

7.1 模型分析

模型数据较少，不具有普遍性

7.2 方向

找更多数据，进行数值模拟计算。

参考文献

8 问题一所用程序：

8.1 程序：

```
% 代码段
function [accuracy]=my_face_recognition( train_dir ,test_dir ,train_num ,test_
%该函数实现了利用PCA方法进行人脸识别的过程
%Input
%       train_dir: 库数据集的目录
%       test_dir: 测试数据集的目录
%       train_num: 选择的库数据集的个数
```

```
%          test_num：要测试的数据集的个数，要小于库数据集个数

if train_num<=test_num
    fprintf('库数据集要大于测试数据集！\n');
    return ;
end


%因为文件大小固定，所以在此我们设置矩阵的行列为定值
row=315;
column=236;
train_data=zeros(train_num,row*column);%预分配数据可以加速数据读取，矩阵的行
train_files=dir(train_dir);%获取库目录下的所有文件，获得的每一个文件都是一个
for i=1:train_num
    file_name=sprintf('%s\\%s',train_dir,train_files(i+2).name);%这里需要加
    img_data = imread(file_name);
    [m,n] = size(img_data);
    if m ~= row || n ~= column
        img_data = imresize(img_data,[row column]);
    end
    %[row column]=size(img_data);
    img_data=img_data(1:row*column);%将读取的数据转成一个行向量
    train_data(i,:)=img_data;%将该行向量添加到库集中
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%求平均脸，与人脸识别无关，只是一个测试
imgmean=mean(train_data);
size(imgmean);
mean_img=reshape(imgmean,row,column);
mean_img=uint8(mean_img);
imwrite(mean_img,'H:\1.bmp');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%{
if b==1
    for i=1:test_num
        train_data(i,:)=train_data(i,:)-imgmean;
    end
end
%}

%进行主成份分析，返回的结果为
%    COEFF：特征向量
%    latent：特征值，按由大到小的顺序排列
%当数据的维度大于数据个数时，通过在函数后面添加参数'econ'可以加速计算
```

```
%[COEFF,~,latent] = princomp(train_data,'econ');

[COEFF,~,latent] = pca(train_data);

%保存的维度（特征值）个数使图像保存的能量大于95%
dimension_left=0;
cum_percent=cumsum(latent)/sum(latent);
for i=1:length(cum_percent)
    if cum_percent(i)>=energy
        dimension_left=i;
        break;
    end
end
%fprintf('dimension left is %d\n',dimension_left);


%将库数据集进行降维
train_data_reduced=train_data*COEFF(:,1:dimension_left);

%读取测试数据集
test_data=zeros(train_num,row*column);%预分配数据可以加速数据读取
test_files=dir(test_dir);%获取库目录下的所有文件，获得的每一个文件都是一个结
for i=1:test_num
    file_name=sprintf('%s\\%s',test_dir,test_files(i+2).name);%这里需要加双

    img_data = imread(file_name);
    [m,n] = size(img_data);
    if m ~= row || n ~= column
        img_data = imresize(img_data,[row column]);
    end
    img_data=img_data(1:row*column);%将读取的数据转成一个行向量
    test_data(i,:)=img_data;%将该行向量添加到库集中
end

%{
if b==1
    for i=1:test_num
        test_data(i,:)=test_data(i,:)-imgmean;
    end
end
%}

%将测试数据集进行降维
test_data_reduced=test_data*COEFF(:,1:dimension_left);

accuracy=0;
```

```
for i=1:test_num
    %通过计算向量二阶范数的方法计算欧式距离
    min=norm(test_data_reduced(i,:)-train_data_reduced(1,:));
    position=1;
    for j=2:train_num
        distance=norm(test_data_reduced(i,:)-train_data_reduced(j,:));
        if min>distance
            min=distance;
            position=j;
        end
    end
    %fprintf('test_file:%s,train_file;%s\n',test_files(i+2).name,train_files
    if same_person(test_files(i+2).name,train_files(position+2).name)==1
        accuracy=accuracy+1;
    else
        %fprintf('test_file:%s,train_file;%s\n',test_files(i+2).name,train_
    end
end
accuracy=accuracy/test_num;
fprintf('Accuracy is %f,energy %f,dimension left %d\n',accuracy,energy,dime

%用来比较两个字符串的前五位是否相同
    function result=same_person(s1,s2)
        result=strncmp(s1,s2,100);
    end
end
```