

**模式识别大作业**

题 目 人脸特征识别

学 院 信息科学与工程学院

专 业 控制科学与工程

组 员 李楠希

指导教师 赵海涛

**完成日期： 2018年 10 月26日**

模式识别作业报告——人脸特征识别

Y30180656 李楠希

通过本次模式识别的课程学习以及课下查阅资料，我对主元分析法（PCA）以及SVM支持向量机分类有了一定的了解，此次，将其运用到人脸特征识别算法之中，希望对这些知识会有更好的理解与巩固。

本次试验，采用ORL人脸图库，这些数据为已经预处理的数据，格式为pgm，可以在matlab中使用imshow(imread('图片路径'))打开，数据是400张人脸图片，属于40个人，每个人10张，每张大小112\*92像素。将每个人的前五张图片作为训练集进行训练，将每个人的另外五张作为测试集，以判断能否正常识别，并匹配出相似度较高的照片。下面简单讲述本次试验的一些原理以及算法的详细步骤。

1. **人脸识别经典算法——特征脸**

特征脸技术是近代发展起来的用于人脸或者一般性刚体识别以及其它涉及到人脸处理的一种方法。其主要方法是首先将一批人脸图像转换成一个特征向量集，称为“Eigenfaces”，即“特征脸”，它们是最初训练图像集的基本组件。而识别的过程是把一副新的图像投影到特征脸子空间，并通过它的投影点在子空间的位置以及投影线的长度来进行判定和识别。

将图像变换到另一个空间后，同一个类别的图像会聚到一起，不同类别的图像会聚力比较远，在原像素空间中不同类别的图像在分布上很难用简单的线或者面切分，变换到另一个空间，就可以很好的把他们分开了。Eigenfaces选择的空间变换方法是PCA（主成分分析），利用PCA得到人脸分布的主要成分，具体实现是对训练集中所有人脸图像的协方差矩阵进行本征值分解，得到对应的特征向量，这些特征向量就是“特征脸”，如下图。每个特征向量或者特征脸相当于捕

**图1 特征脸举例**

捉或者描述人脸之间的一种变化或者特性。这就意味着每个人脸都可以表示为这些特征脸的线性组合。

有论文研究表明，一般情况下40个特征脸的应用已经可以使识别精度达到较高。具体的特征脸产生过程将会在算法里讲述。

1. **基本算法原理**

2.1 主元分析法

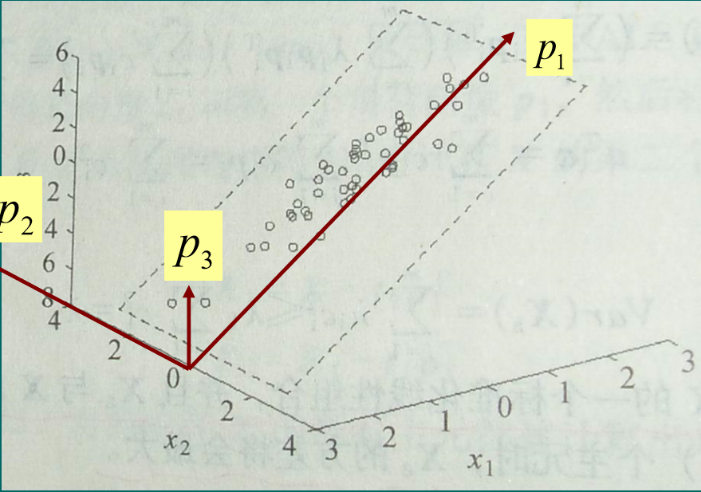
PCA方法是将高维过程数据投影到正交的低维子空间，并保留主要过程信息。而在几何上，把样本构成的坐标系，通过某种线性组合旋转到新的坐标空间，新的坐标轴代表了具有最大方差的方向。

假设代表一个包含了m个变量的测量样本，每个变量各有n个独立采样，构造出测量数据矩阵 ，其中每一列代表一个测量变量，每一行代表一个样本。

1. 对数据矩阵进行协方差分解，并选择主元的个数

X的协方差矩阵为 ，对其进行特征值分解，并且按照特征值的大小降序排序，如下：



其中，是一个对角阵，也是S的特征值矩阵，而且其对角线上的元素满足；是S的特征向量矩阵，维数为m x m，是的前A列，包含所有主元的信息，是余下的m-A列，包含非主元信息。通过得到的特征向量，可以得到对应的主元。由此，包含非主元信息的部分可以略去，即将用于X矩阵表示的维数进行降低。如果用两个主元来表示这些数据，可以用两条相互垂

**图2 主元分析法分解**

直的直线构成的平面来表示，其中一条直线是数据的第１个主元，代表数据最大

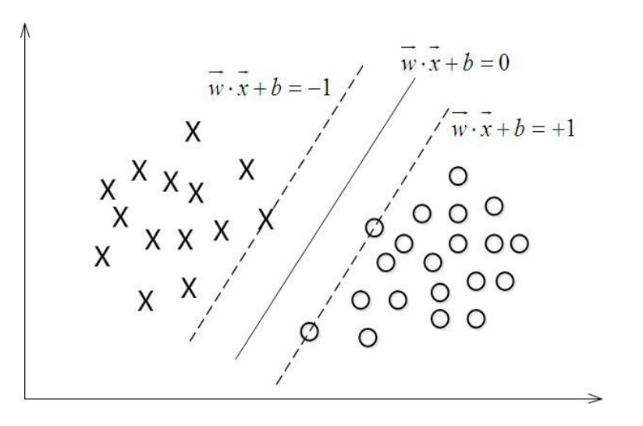
变化方向；另外一条直线是数据的第２个主元，代表数据的第二大变化方向。

2.2 SVM支持向量机分类器

支持向量机（简称SVM），通俗来讲，它是一种二类分类模型，其基本模型定义为特征空间上的间隔最大的线性分类器，其学习策略便是间隔最大化，最终可转化为一个凸二次规划问题的求解。

而想要理解SVM，首先必须清楚一个概念：线性分类器。即给定一些数据点，它们分别属于两个不同的类，现在便要找到一个线性分类器把这些数据分成两类。如果用x表示数据点，用y表示类别（y可以取1或者-1，分别代表两个不同的类），一个线性分类器的学习目标便是要在n维的数据空间中找到一个超平面（hyper plane），这个超平面的方程可以表示为下式：

 （2-1）

其中最简单的例子便是如下图所示，现在有一个二维平面，平面上有两种不同的数据，分别用圈和叉表示。由于这些数据是线性可分的，所以可以用一条直线将这两类数据分开，这条直线就相当于一个超平面，超平面一边的数据点所对应的y全是-1 ，另一边所对应的y全是1。这个超平面可以用分类函数2-1表示，当f(x) 等于0的时候，x便是位于超平面上的点，而f(x)大于0的点对应 y=1 的数据点，f(x)小于0的点对应y=-1的点。

**图3 二维数据分类**

接下来的问题是，如何确定这个超平面呢？从直观上而言，这个超平面应该是最适合分开两类数据的直线。而判定“最适合”的标准就是这条直线离直线两边的数据的间隔最大。所以，得寻找有着最大间隔的超平面。具体推导方法可参考相关资料，在此不再详述。另外，本次人脸识别不仅只是分出两类，而是需要分出40个类别，因而相对较为复杂，此次试验调用libsvm工具箱，libsvm 是台湾大学林智仁副教授等开发设计的一个简单、易于使用和快速有效的SVM模式识别与回归的软件包，它不仅提供了编译好的可在 Windows 系列系统的执行文件，还提供了源代码，方便改进、修改以及在其它操作系统上应用；该软件还有一个特点，就是对 SVM 所涉及的参数调节相对比较少，提供了很多的默认参数，利用这些默认参数就可以解决很多问题；并且提供了交互检验等问题，包括基于一对一算法的多类模式识别问题。

1. **算法过程实现**

本次试验采用MATLAB编程实现，在目标文件夹的人脸库中，每个人有10张图，我们将前5张作为训练集，就有200张图。用一个矩阵来表示这200张图，就形成了一个200\*10304的矩阵。后5张图用来当测试集，当flag为0时，找出训练集的标签realclass(200\*1的矩阵)与数据f\_matrix（200\*10304的矩阵），

当flag为1时，就是找出测试集来测试分类效果，得出准确率。SVM训练的目的就是利用训练集找到这么一个分类函数（模型），再在测试集中检测。具体分为六个函数，分别进行了不同的工作处理，下面将详细讲述，并给出部分代码。

3.1 算法主体框架

算法主体框架即用主函数来进行描述，其按照步骤依次是定义相关变量，训练集处理（读取训练数据，训练数据PCA特征提取，显示主成分脸，训练特征数据归一化，SVM样本训练），测试数据处理（读取测试数据，测试数据特征降维，测试数据归一化，SVM样本分类，人脸识别模块）。函数内容如下，主要是按照正常逻辑进行函数调用。

|  |
| --- |
| clc,clear  npersons=40;%选取40个人的脸  global imgrow;  global imgcol;  imgrow=112;  imgcol=92;  disp('读取训练数据...')  [f\_matrix,train\_label]=ReadFace(npersons,0);%读取训练数据  nfaces=size(f\_matrix,1);%样本人脸的数量  disp('.................................................')  %对训练集进行降维处理  disp('训练数据PCA特征提取...')  mA=mean(f\_matrix);  k=20;%降维至20维  [train\_pcaface,V]=fastPCA(f\_matrix,k,mA);%主成分分析法特征提取  disp('.................................................')  %显示主成分脸，即特征脸，低维的基  disp('显示主成分脸...')  visualize(V)%显示主分量脸 ,即特征脸  disp('.................................................')  %低维训练集归一化  disp('训练特征数据归一化...')  disp('.................................................')  lowvec=min(train\_pcaface);  upvec=max(train\_pcaface);  train\_scaledface = scaling( train\_pcaface,lowvec,upvec);  %SVM样本训练  disp('SVM样本训练...')  disp('.................................................')  model = svmtrain(train\_label,train\_scaledface,'-t 0');  %读取测试数据  disp('读取测试数据...')  disp('.................................................')  [test\_facedata,test\_facelabel]=ReadFace(npersons,1);  %测试数据降维  disp('测试数据特征降维...')  disp('.................................................')  %test\_pcatestface-测试数据低维空间的表示  m=size(test\_facedata,1);  for i=1:m  test\_facedata(i,:)=test\_facedata(i,:)-mA;  end  test\_pcatestface=test\_facedata\*V;    %测试数据归一化  disp('测试特征数据归一化...')  disp('.................................................')  scaled\_testface = scaling( test\_pcatestface,lowvec,upvec);  %利用训练集建立的模型，对测试集进行分类  disp('SVM样本分类...')  disp('.................................................')  [predict\_label,accuracy,decision\_values]=svmpredict(test\_facelabel,scaled\_testface,model);  %人脸识别模块  disp('人脸识别模块')  disp('.................................................')  recognition(mA,V,model) ; |

3.2 数据读取

数据读取为程序中的ReadFace函数，首先建立主要变量realclass作为数据标签，f\_matrix为5行112\*92列的矩阵，每一行为一张图片的灰度数据。读取图片数据主要依靠imread函数，将输入路径的图片的每个像素灰度值保存在矩阵当中。而此函数的使用首先依靠strcat函数与num2str将输入路径转换为字符串。为避免篇幅过长，在此仅列出主要函数，具体注释见源代码。

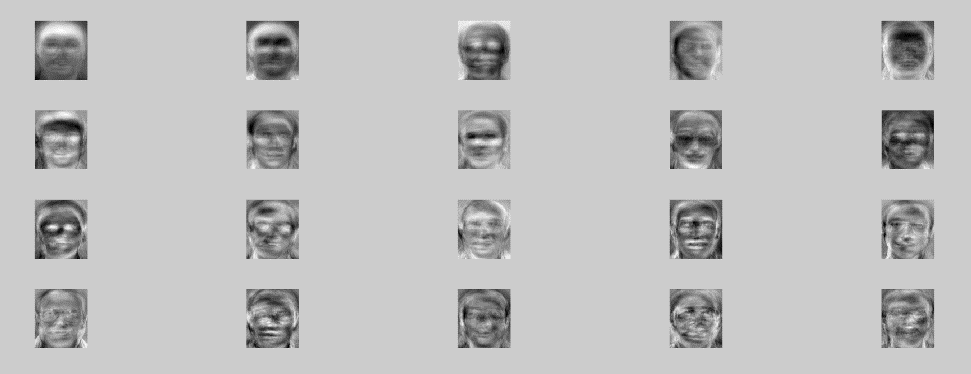
|  |
| --- |
| facepath=strcat('D:\matlab文件\renlianshibie\faces\s',num2str(i),'\');  cachepath=facepath;  for j=1:5  facepath=cachepath;  if flag==0  %函数strcat(a,b,...)说明:将输入字符a,b...连接成单个字符  facepath=strcat(facepath,num2str(j));  else  facepath=strcat(facepath,num2str(j+5));  end  realclass((i-1)\*5+j)=i;  facepath=strcat(facepath,'.pgm');  %读取输入路径的图片，将每个像素灰度值保存在输出的矩阵中  img=imread(facepath);  f\_matrix((i-1)\*5+j,:)=img(:)'; |

3.3 降维处理

使用主成分分析方法进行降维，注意首先要对原矩阵进行中心化处理，即将数据集的均值归零，只读数据的偏差部分。其步骤较为简单。同样与中心化相对应的是还需进行特征数规范化，即将同一个样本中的不同维度归一化，因为对于不同的属性，如果不归一化的话是不具有比较性的，两者不在一个量级上。具体是将图像数据中一个样本的不同维度的值，最小值和最大值规范到-1和1，其他值按比例规范到（-1,1）。

|  |
| --- |
| function [ pcaA,V] = fastPCA( A,k,mA)  m=size(A,1);  Z=(A-repmat(mA,m,1)); %中心化样本矩阵  T=Z\*Z';  [V1,D]=eigs(T,k);  V=Z'\*V1; %协方差矩阵的特征向量  %V = (Z'\*Z)./(size(Z,1)-1)  for i=1:k %特征向量单位化  l=norm(V(:,i));  V(:,i)=V(:,i)/l;  end  pcaA=Z\*V;  end  function [ scaledface] = scaling( faceMat,lowvec,upvec )  upnew=1;  lownew=-1;  [m,n]=size(faceMat);  scaledface=zeros(m,n);  for i=1:m  scaledface(i,:)=lownew+(faceMat(i,:)-lowvec)./(upvec-lowvec)\*(upnew-lownew);  end  end |

3.4 显示特征脸

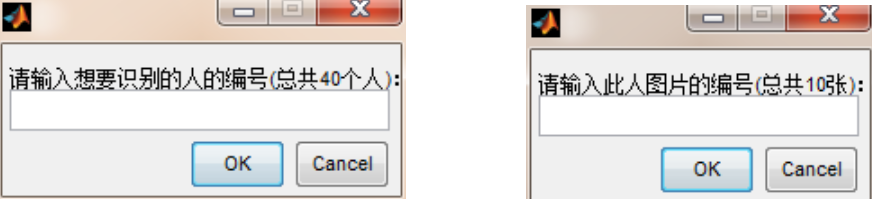
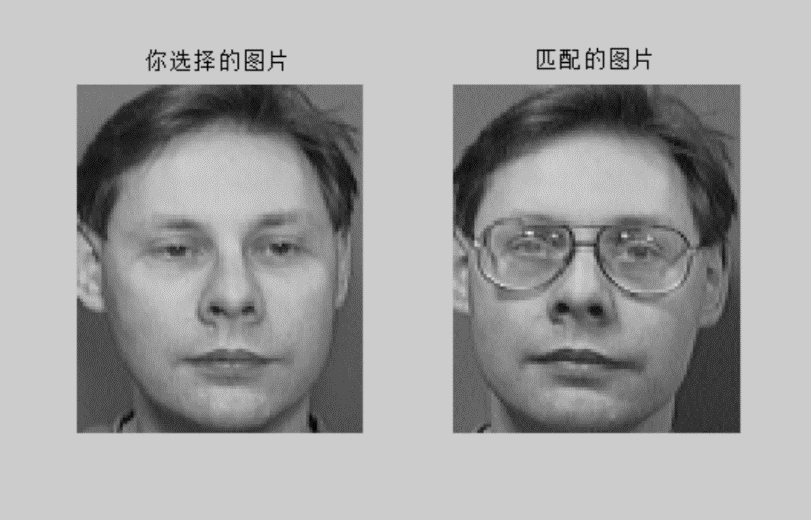
 将变换空间中的基向量（单位特征向量）显示出来，由于降低为了20维，因而在此维度上的基便是这20个特征脸，其他所有的经过了降维的脸都可以由这20张特征脸线性表示，所以我们要进行脸部识别时，把选择要识别的脸进行降维处理后，再右乘这些基（特征脸）所构成的矩阵，就能得到这些脸在低维中的线性表示，而这些表示就是识别的依据。其特征脸如下图：

**图4 特征脸**

|  |
| --- |
| function visualize( B )  global imgrow;  global imgcol;  figure  img=zeros(imgrow,imgcol);  for i=1:20  img(:)=B(:,i);  subplot(4,5,i);  imshow(img,[])  end  end |

3.5 脸部识别

建立函数为新的样本利用已经建立好的分类模型进行识别，在输入框中输入是别的图片，并对图片信息进行处理，利用svmpredict对测试样本进行分析，判断出属于哪个人，并显示出最为相似的图片。运行该程序时会弹出如下输入框。

输入序号后将会为输入的测试图片选择分类号，并显示出匹配的图片。

**图5 测试图片匹配**

|  |
| --- |
| function recognition(mA,V,model)  % mA-均值  % V-协方差矩阵特征向量  % model-通过SVM对训练集训练得出的已经建立好的模型  global imgrow;  global imgcol;  %弹出输入框，选择要识别的图片  select\_person\_num=str2double(cell2mat(inputdlg('请输入想要识别的人的编号(总共40个人)：')));  select\_img\_num=str2double(cell2mat(inputdlg('请输入此人图片的编号(总共10张)：')));  %对图片信息进行处理，化为1\*10304的行向量  disp('读取选择的图片...')  select\_facepath=strcat('F:\MATLAB人脸识别\Face\facedata\s',num2str(select\_person\_num),'\',num2str(select\_img\_num),'.pgm');  select\_img=imread(select\_facepath);  select\_matrix=zeros(1,imgrow\*imgcol);  select\_matrix(1,:)=select\_img(:)';  select\_matrix=(select\_matrix-mA)\*V;%PCA降维后的低维表示  %图形归一化  disp('规范化选择的图片...')  select\_matrix = scaling( select\_matrix,min(select\_matrix),max(select\_matrix));  %测试选择的图片，accuracy只有两个值，100%表示匹配正确，0%表示匹配错误  disp('测试选择的图片...')  [select\_predict\_label,accuracy,decision\_values]=svmpredict(select\_person\_num,select\_matrix,model);  %显示原有图片和匹配图片进行比较  disp('显示选择的图片...')  figure(2);  subplot(1,2,1);imshow(select\_img);title('你选择的图片');  subplot(1,2,2);  imshow(imread(strcat('F:\MATLAB人脸识别\Face\facedata\s',num2str(select\_predict\_label),'\',num2str(1),'.pgm')));  title('匹配的图片'); |

3.6 试验结果

通过对此程序多次运行可以发现，对输入照片的正确识别率为90.5%左右，通过粗略研究发现，将PCA替换为KPCA，或者用参数优化的方法调整参数可以略微提高准确率，另外便是重新编写SVM训练模型，但是对于分成几十类的SVM编写，对于本人来说略有难度，日后有空可以继续研究（暂且对python理解较浅，因而本次试验采用MATLAB进行编程）。

**附：文件说明**

本次附件一共包含有：

1 大作业报告；

2 最终的MATLAB实现程序源码。

3 导出了的预测数据集