

深圳大学实验报告

课程名称 机器学习

项目名称 PCA 算法

学 院 计算机与软件学院

专 业 软件工程腾班

指导教师 赖志辉

报 告 人 郑杨 学号 2020151002

实验时间 2022 年 3 月 5 日至 2022 年 3 月 8 日

实验报告提交时间 2022 年 3 月 8 日

教务处制

一、实验目的与要求

1. 实现 PCA 算法的人脸重构，即用 20,40,60,80,...,160 个投影来重构图像的效果。
2. 实现 PCA 算法的人脸识别，给出 10,20,30,...,160 维的人脸识别识别率。
3. 用 PCA 用来进行人脸图像降维，实现 3 个不同数据集多个子集的二维和三维空间实现数据的可视化。
4. 同时设计一个新的 PCA 算法，内容简要写在实验报告中，并与经典 PCA 比较。

二、实验内容与方法

1. PCA 原理简述，使用公式推理相关原理。
2. 编程实现 PCA
3. PCA 人脸重构，使用 matlab 实现并展示重构图像。
4. PCA 人脸识别，k 近邻分类，使用 matlab 实现并展示识别率。
5. 人脸图像降维的可视化，使用 matlab 实现并展示二维与三维空间。
6. 简述新的 PCA 算法，与经典 PCA 进行比较。

三、实验步骤与过程

1. PCA 原理简述

(1) 背景

我们知道，许多机器学习算法的复杂度和数据集的维数有着密切关系，甚至与维数呈指数级关系。在数据集维数很高的情况下，机器学习的资源耗费是不可接受的，因此我们必须对数据进行降维处理。降维意味着数据的丢失，但由于实际数据一般都会有或大或小的相关性，我们可以想办法在降维的同时使得损失的数据最少。

(2) 数学原理阐述

假设我们现在有 n 个 m 维数据（表示为列向量） $(a_1, a_2, a_3, \dots, a_n)$ ，我们想要通过某种方法把 m 维降低为 k 维。所谓的降维，实际上就是由高维向量向每个低维基上面投影，形成由 k 个新坐标组成的新向量。一般的，如果我们有 n 个 m 维向量，想要将他们变换到由 k 个 m 维向量所表示的新空间中，那么首先将 k 个基按行组成矩阵 A ，然后将向量按列组成矩阵 B ，那么 AB 中的第 i 列就是 A 中第 i 列降维的结果。如下：

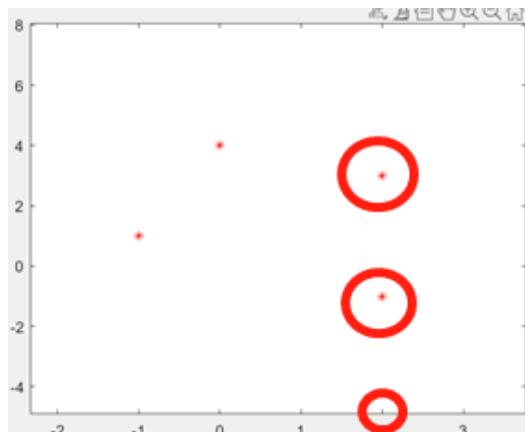
$$\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \end{pmatrix} (a_1 \ a_2 \ \dots \ a_n) = \begin{pmatrix} b_1 a_1 & b_1 a_2 & \dots & b_1 a_n \\ b_2 a_1 & b_2 a_2 & \dots & b_2 a_n \\ \vdots & \vdots & \ddots & \vdots \\ b_k a_1 & b_k a_2 & \dots & b_k a_n \end{pmatrix}$$

那么有了降维的数学表示之后呢，我们的目标就是找到一组最优的基，使得数据降低维度之后尽可能多的保留原始信息。为了不让整个过程过于抽象，我们看一个简单的例子。假设我们有 4 个 2 维数据，如下：

$$\begin{pmatrix} -1 & 2 & 0 & 2 \\ 1 & 3 & 4 & -1 \end{pmatrix}$$

如果我们想要把这些 2 维数据降为 1 维数据，那就相当于是在二维坐标系中找到一个方向，把这些平面上的点投影到这个方向上成为直线上的点。如下图，如果我们把这个

方向定为 X 轴正方向的话，那么图中有两个点的投影点会重叠，这样就损失了原有信息。



直观上的理解就是，我们选择的方向需要使得投影点尽可能的分散。数学形式上，可以表述为**最大化投影之后向量相同字段间的方差**。（这里的字段也就是向量的每一维度）由方差的定义可得，一个字段的方差就是每个元素与字段均值差的平方和的均值，如下：

$$Var(i) = \frac{1}{n} \sum_{j=1}^n (a_{ji} - \mu)^2$$

其中， a_{ji} 表示的是列向量 a_j 的第 i 个元素。

而为了更方便的计算，我们会把所有的向量做一个中心化处理，就是减掉他们的均值，于是上式中的 μ 就可以消去，变为：

$$Var(i) = \frac{1}{n} \sum_{j=1}^n a_{ji}^2$$

那么对于二维变为一维的问题而言，就是找到一个一维基，使得上述式子的值最大。接下来让我们来看看**更高维的问题**，比如说**三维降到二维**的问题。与之前类似，我们首先找到一个方差最大的方向，那么第二个方向又应该如何选择呢？当然我们希望第二个方向的方差也最大，但是如果单纯考虑方差的话，这两个方向的重复度会很高，显然这样选择的维度是没有用的。因此，我们希望任意两个方向之间不存在线性相关性。数学上，我们可以用协方差来表示两个字段的线性相关性，由于字段均值为 0，所以表示起来也非常的简洁：

$$Cov(i, j) = \frac{1}{n} \sum_{k=1}^n a_{ki} a_{kj}$$

为了让两个字段的协方差为 0，我们选择的两个基必须正交。那么我们的优化问题转化为：**选择 k 个单位正交基，使得原始数据变换到这组基上后，字段间协方差为 0，字段的方差尽可能大。**

我们再次借助数学工具，协方差矩阵。协方差矩阵的定义为：我们把 n 个 m 维数据排列成 $m \times n$ 的矩阵 X ，令 $C = \frac{1}{n} XX^T$ ，则 C 是一个对称矩阵，它的对角线是每个字段的

方差，第 i 行第 j 列是第 i 个字段和第 j 个字段的协方差，我们把矩阵 C 称为协方差矩

阵。那么我们的目标就是把这个协方差矩阵对角化，使得除了对角线外的其他元素都为 0。

所以说，我们降维后数据的协方差矩阵 D 必须是一个对角矩阵，我们来推导一下 D 和 C 的关系。假设我们选择的基构成的矩阵为 P ，降维后的数据矩阵为 Y ，那么有：

$$Y = PX$$

由协方差矩阵定义有：

$$D = \frac{1}{n} YY^T = \frac{1}{n} (PX)(PX^T) = \frac{1}{n} PXX^T P^T = P\left(\frac{1}{n} XX^T\right)P^T = PCP^T$$

那么我们就是想要找到一个矩阵 P 使得协方差矩阵 C 对角化。而在线性代数领域，已经存在以下结论：

一个 m 行 m 列的实对称矩阵一定可以找到 m 个单位正交特征向量，设这 m 个特征向量为 $e_1, e_2, e_3, \dots, e_m$ ，我们把他们按列排成一个矩阵 $E = (e_1 \ e_2 \ \dots \ e_m)$ ，则对于 C 存在以下等式：

$$E^T C E = \Lambda$$

其中， Λ 是一个对角矩阵，对角线上是 E 的每一个特征向量对应的特征值。

那么对比一下我们可以发现，我们要找的矩阵 P ，其实就是 E^T ！

(3) PCA 基本步骤

让我们回顾一下整个 PCA 的流程。

I. 首先把数据按列排列成一个矩阵 X

II. 然后算出协方差矩阵 $C = \frac{1}{n} XX^T$

III. 之后求出协方差矩阵 C 的特征值与特征向量，并按特征值从大到小排序

VI. 取出特征值前 k 大对应的特征向量组成降维矩阵 P

2. 编程实现 PCA

我把整个过程拆成了两个函数，Readdata 和 PCA，故名思意就是读取数据和执行 PCA 算法。Readdata 函数输入为文件地址、人物类别数、每一类作为训练集的图片数量和每一类的图片总数；输出为训练集矩阵、测试集矩阵、训练集标签和测试集标签等。PCA 函数输入为训练集矩阵和将要选取的维度数量；输出为用于降维的转换矩阵和训练集矩阵每一行的均值（列向量）。如下图所示：

```
% input: data address, count of classes,
%         count of datas for training, count of the datas in each class
% output: matrix of the training set(as column vector), matrix of the testing set,
%         labels of training set, labels of testing set,
%         row of original image, column of original image
function [X_train, X_test, Label_train, Label_test, row, col] = Read_data(address, class_cnt, train_cnt, eachclass_cnt)

% input: training set, first kth dimensions for choosing
% output: the matrix for reducing dimension, the mean vector of training set
function [P, Mean] = PCA(X_train, k_dimensions)
```

关于 ReadData 函数，就是打开文件读取，把图片矩阵转换为列向量加入训练集矩阵或者测试集矩阵中而已，不再赘述。对于 PCA 函数，代码实现于上一小节所叙述的步骤一致，由于篇幅问题也不再解释。这里详细讲述一下 PCA 求取协方差矩阵特征向量与特征值的加速方法。

由于求取方阵特征值与特征向量的时间复杂度为 $O(m^3)$ ， m 为数据维度。对于高维数据而言，速率极低。在数据数量 $n \ll m$ 的情况下，我们尝试把时间复杂度转化为 $O(n^3)$ 。

我们尝试计算 $C = \frac{1}{n} X^T X$ 的特征值与特征向量，设某一个特征向量为 u ，对应的特征值为 λ ，由特征值与特征向量的定义有：

$$X^T X u = \lambda u \Rightarrow X X^T X u = \lambda X u \Rightarrow X X^T (X u) = \lambda (X u)$$

就是说这样计算出来的特征值与原矩阵相同，而特征向量只需要左乘数据矩阵 X 就可以变换为原来的特征向量。那么可以利用这一特点进行加速。

那么有了这两个函数之后，我们就只需要在主程序中设置好参数，调用函数进行一系列实验即可。

```
% main function
clear
close all

address = 'C:\Users\19116\Desktop\Grade Two second'
class_cnt = 40;
train_cnt = 5;
eachclass_cnt = 10;
k_dimension = 2;

%% read data
[X_train, X_test, Label_train, Label_test, row, col] = read_data(address, class_cnt, train_cnt, eachclass_cnt);
%% PCA
P = PCA(X_train, k_dimension);
```

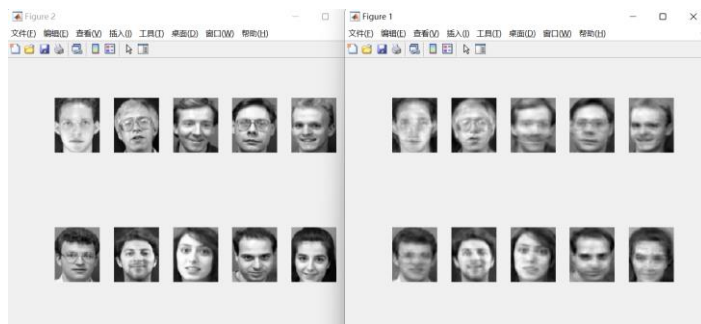
3. 使用 PCA 实现人脸重构（分别用 20、40、60、...、160 个投影）

编写了一个 rebuild 函数，如图：

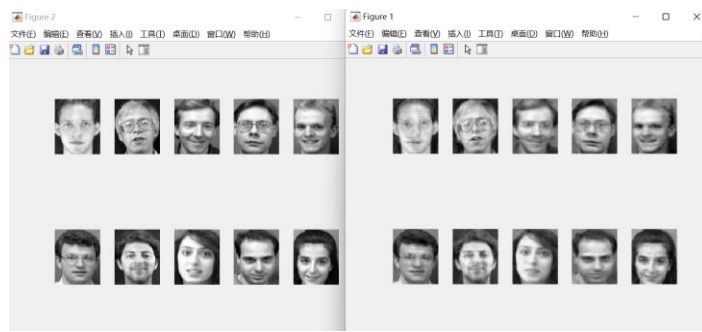
```
% input: the matrix for reducing dimension, testing set,
%         the row of original image, the column of original image,
%         the mean vector of training set
function rebuild(P, X_test, r, c, Mean)
```

采用 **ORL 数据集**（ 46×56 ，40 类每类 10 张图片）进行人脸重构实验，选取了 10 类人脸，使用每类人脸的 10 张图片作为训练集，采用第 1 张进行重构测试，测试如下：（下图中左边为原图，右边为重构图）

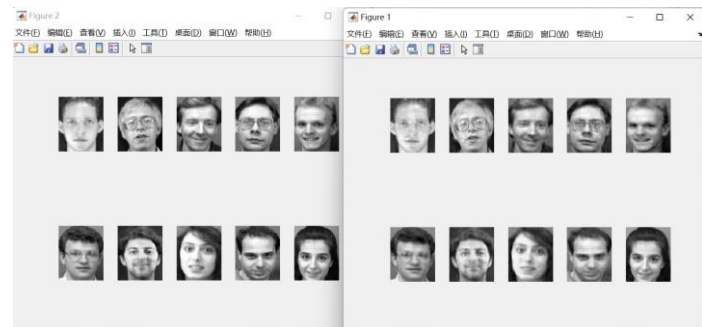
用 20 个投影：



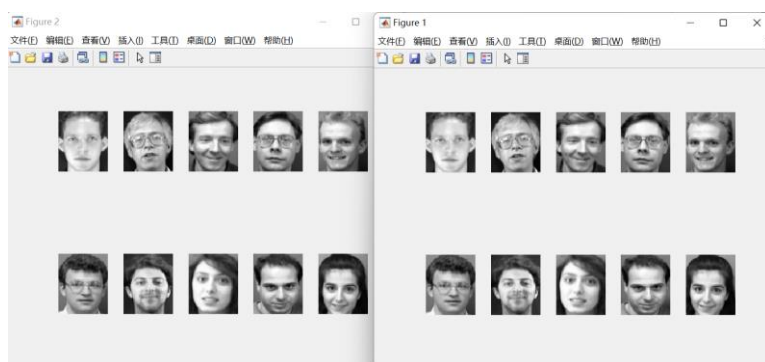
用 40 个投影：



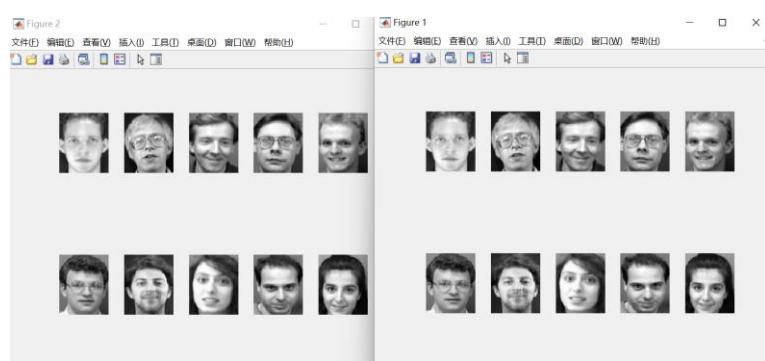
用 60 个投影：



用 100 个投影：



用 160 个投影：



我们发现，随着保留维数的增多，重构图像与原图越来越接近。当维数达到 100 时，重构图像已经与原图没有明显的差异了。由于 orl 数据集的原图像维数本身就比较小，所以保留的维数不需要很多。

4. 使用 PCA 进行人脸识别，展示识别率

编写了 recognize 函数，如下图：

```
% input: the matrix for reducing dimension, training set,  
%         testing set, the labels of training set,  
%         the labels of testing set, the mean vector of training set  
% output: the recognition rate  
function rate = recognize(P, X_train, X_test, Label_train, Label_test, Mean)
```

使用最近邻算法实现降维后的人脸向量比对，该方法原理也不再赘述。

采用 **ORL 数据集**（ 46×56 ，40 类每类 10 张图片）和 **FERET 数据集**（ 80×80 ，200 类每类 7 张图片）进行实验。对于 ORL 数据集，使用了全部图片，每一类人中使用前 5 张图片作为训练集，后 5 张图片作为测试集。对于 FERET 数据集，使用了 100 类人物图片，每一类人物图片中使用前 5 张图片作为训练集，后 2 张图片作为测试集。分别取前 10、20、...、160 个投影进行实验，实验结果如下：

Rate	Dimension															
Dataset	10	20	30	40	50	60	70	80	90	100	110	120	130	140	150	160
ORL	0.850	0.865	0.880	0.885	0.890	0.890	0.890	0.885	0.890	0.895	0.895	0.895	0.900	0.900	0.900	0.900
FERET	0.440	0.490	0.510	0.520	0.520	0.510	0.500	0.510	0.510	0.510	0.510	0.510	0.510	0.510	0.510	0.510

通过实验我们可以发现，识别率随着选择维数的增多基本呈现上升趋势，但也会在一定的维数范围内达到饱和状态。这说明，前面一定量的维数所占能量比重已经很大，足以表示图像主成分。

但是 FERET 数据集的识别率并不是很理想，通过观察数据集发现，FERET 人物图像侧面图占比很大，加上 PCA 本身就是无监督学习，不能利用类别信息，使得识别率并没有很高。不仅如此，FERET 中还存在着非常相似的人脸，这也加大了误判的概率。

由于 FERET 每类人物的后五张图片中正脸角度居多，我尝试用后五张图片作为训练集，前 2 张图片作为测试集进行实验，识别率有了大幅好转。实验结果如下：

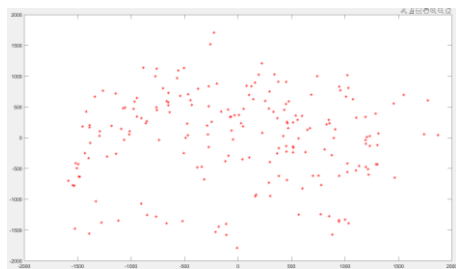
Rate	Dimension															
Dataset	10	20	30	40	50	60	70	80	90	100	110	120	130	140	150	160
FERET	0.660	0.750	0.765	0.795	0.810	0.805	0.815	0.815	0.815	0.810	0.805	0.815	0.810	0.815	0.810	0.810

5. 使用 PCA 进行降维，实现二维空间与三维空间可视化

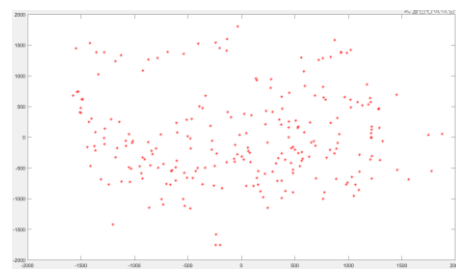
将维度选择为 2 和 3，运行 PCA 降维之后，使用 plot 函数与 plot3 函数可视化即可。

(1) 首先使用 ORL 数据集，取 40 类人物照片，分别取每类前五、六、七张图片作为训练集，进行二维空间与三维空间可视化。

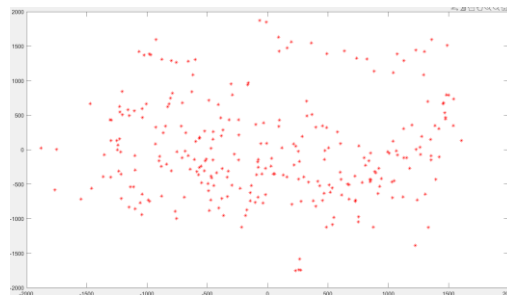
二维空间：



前五张图片作为训练集

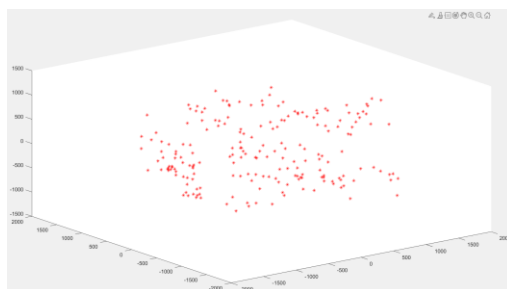


前六张图片作为训练集

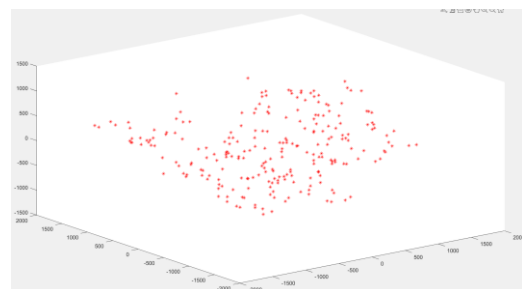


前七张图片作为训练集

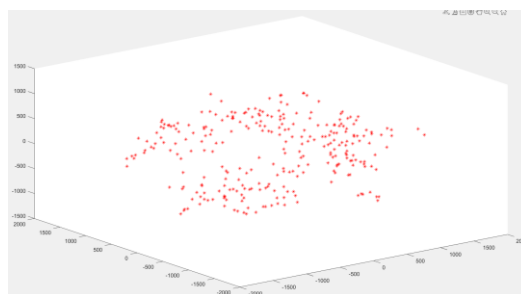
三维空间：



前五张图片作为训练集



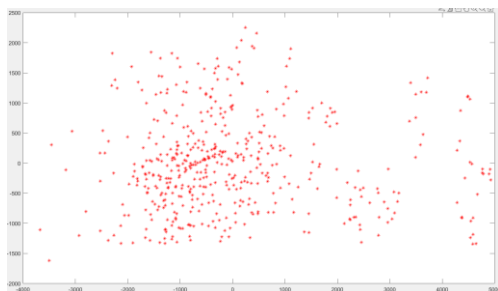
前六张图片作为训练集



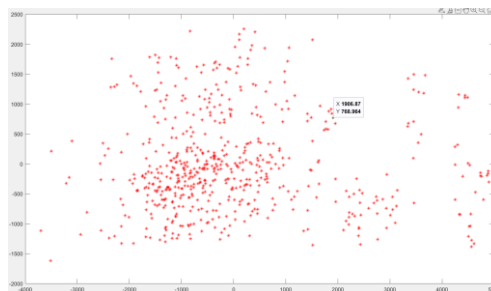
前七张图片作为训练集

(2) 然后使用 FERET 数据集，取前一百类人物照片，分别取每类前五、六、七张图片作为训练集，进行二维空间与三维空间可视化。

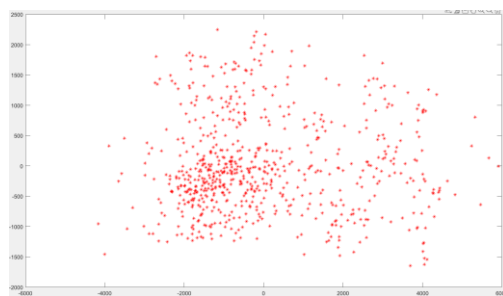
二维空间：



前五张图片作为训练集

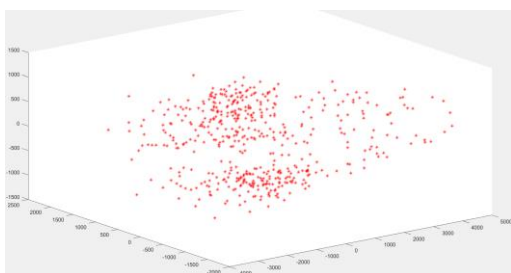


前六张图片作为训练集

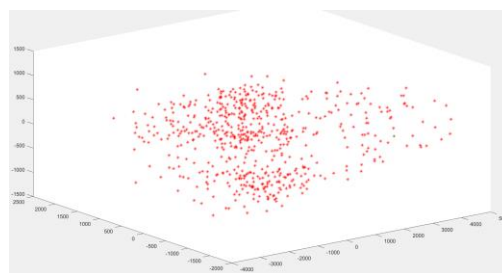


前七张图片作为训练集

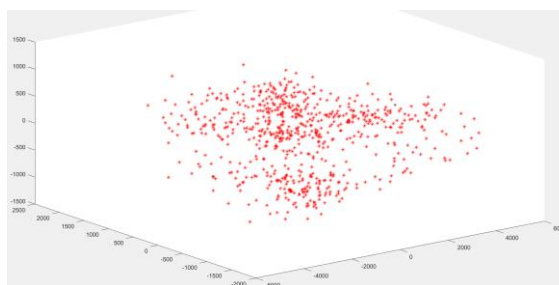
三维空间：



前五张图片作为训练集



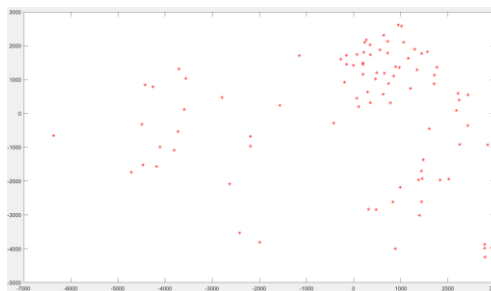
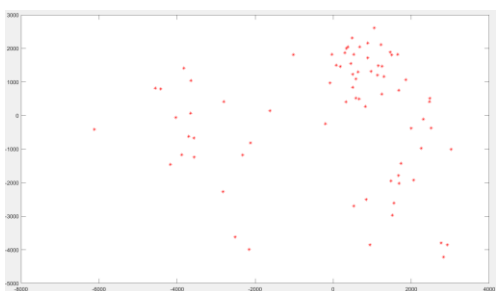
前六张图片作为训练集



前七张图片作为训练集

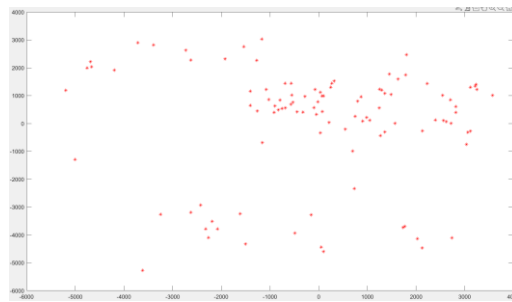
(3) 然后使用 **Yale** 数据集，取所有共 15 类人物照片，分别取每类前五、六、七张图片作为训练集，进行二维空间与三维空间可视化。

二维空间：



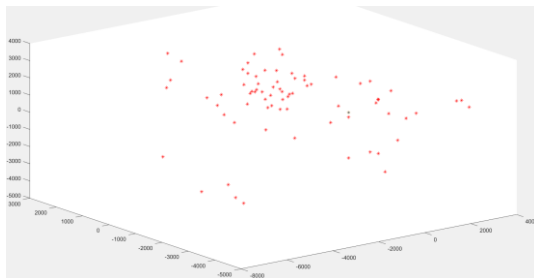
前五张图片作为训练集

前六张图片作为训练集

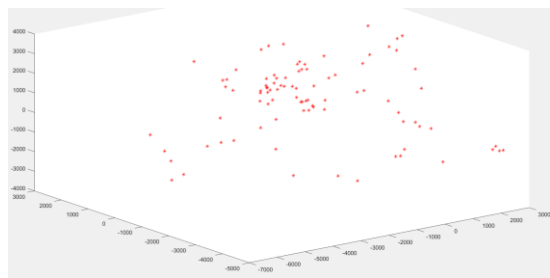


前七张图片作为训练集

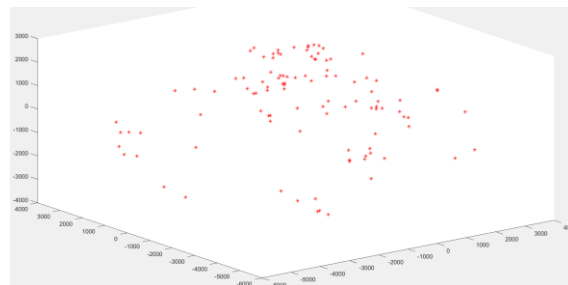
三维空间:



前五张图片作为训练集



前六张图片作为训练集



前七张图片作为训练集

(4) 总结

我们可以发现，无论在哪个数据集里，PCA 降维之后在二维和三维空间中的效果都很好，数据点都比较分散，尽可能多的保留了原始信息。

6. 简述新的 PCA 算法

阅读了相关论文，学习了 2DPCA 算法。与 PCA 类似，2DPCA 也是最大化投影数据的散度。不同的是，2DPCA 直接使用原图像数据矩阵进行降维，具体的说就是把原始图像矩阵通过投影向量变换成一个列向量。用数学语言形式化起来就是，对于某个原始图像矩阵 A ，找到一个列向量 X ，使得变换之后的 $Y = AX$ 的方差最大。也就是最大化 Y 的协方差矩阵的迹：

$$tr(S_x) = X^T [E(A - EA)^T (A - EA)] X$$

令：

$$G = E(A - EA)^T (A - EA)$$

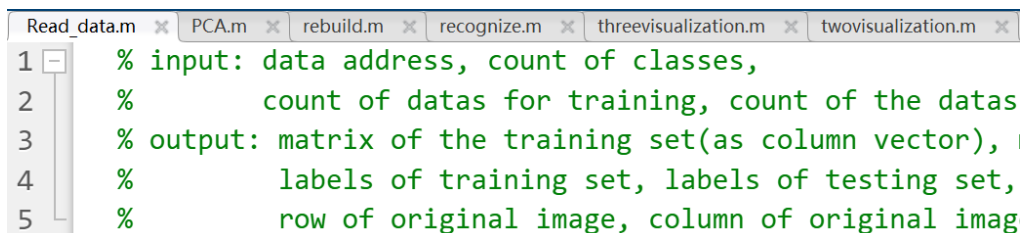
那么就是最大化：

$$J(X) = X^T G X$$

与 PCA 相比,2DPCA 计算的协方差矩阵维数较小,效率较高。论文实验也验证了 2DPCA 的识别准确率,在大多数情况下比 PCA 好。

四、实验结论或体会

在程序设计上,我把每一个功能都包装成一个函数,实现起来比较简洁方便。



```
1 % input: data address, count of classes,  
2 %       count of datas for training, count of the datas  
3 % output: matrix of the training set(as column vector),  
4 %       labels of training set, labels of testing set,  
5 %       row of original image, column of original image
```

在数学原理的理解上,我从本质出发,一步一步推导出整个算法。当然,在数学原理的理解部分还是有所欠缺,线性代数的一些结论还没有给予证明,这也是接下来应该加强的方面。

对 PCA 的理解更加的深入。PCA 在降低维度的过程中,需要计算一个维度矩阵的特征值与特征向量,这无疑要耗费巨大的时间成本。而如果我们把维度矩阵转换为大小为样本数量的矩阵的话,保存的维度信息可能不够。PCA 还受限于无监督学习方式,无法利用图像本身的信息,这也导致了它在一些数据集上的识别率较低。

对论文的阅读速度有所提高,通过博客等相关资料了解了算法大致原理之后,再读论文会大幅提升效率。

由于知识储备还不太够,暂时没有什么新的 idea,不过等阅读了大部分论文之后,会提出自己的想法。

五、参考资料

- [1] PCA 的数学原理 <http://blog.codinglabs.org/articles/pca-tutorial.html>
- [2] Two-Dimensional PCA: A New Approach to Appearance-Based Face Representation and Recognition -Jian Yang, David Zhang, Senior Member, IEEE, Alejandro F. Frangi, and Jing-yu Yang
- [3] Eigenfaces for Recognition -Matthew Turk and Alex Pentland

<p>指导教师批阅意见：</p>	
<p>成绩评定：</p>	
	指导教师签字：
	年 月 日
<p>备注：</p>	

<p>指导教师批阅意见：</p>	
<p>成绩评定：</p>	
	指导教师签字：
	年 月 日
<p>备注：</p>	

<p>指导教师批阅意见：</p>	
<p>成绩评定：</p>	
	指导教师签字：
	年 月 日
<p>备注：</p>	

<p>指导教师批阅意见：</p>	
<p>成绩评定：</p>	
	指导教师签字：
	年 月 日
<p>备注：</p>	

<p>指导教师批阅意见：</p>	
<p>成绩评定：</p>	
	指导教师签字：
	年 月 日
<p>备注：</p>	

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。