程序报告

学号：3190100923　　姓名：陈志博

一、问题重述

本实验采用特征脸算法进行人脸识别。特征脸是第一种有效的人脸识别方法，通过在一大组描述不同人脸的图像上进行主成分分析获得。主成分分析是一种通过降维 技术把多个变量化为少数几个主成分的统计方法，通过消除数据的相关性，找到一个空间，使得各个类别的数据在该空间上能够很好地分离。是重要的特征提取方法之一。
特征人脸识别是一种把人脸从像素空间变换到另一个空间，在另一个空间中做相似性的计算的算法。

二、设计思想

因为主成分分析需要找出协方差矩阵的前 k 个特征向量，而对于转化为一维的图像数据而言，其协方差矩阵是极为庞大的，可能是几万乘几万的规模。这对于求其特征值来说计算量过于庞大了，因此我们采用奇异值分解的方法来求出左奇异矩阵相对较小规模的特征值，并再通过这个来求其原本的特征向量，可以大大减少计算量，提高算法的效率。

三、代码内容

一、求特征人脸：

```
def eigen_train(trainset, k=20):
    avg_img=np.mean(trainset,axis=0)
    norm_img=trainset-avg_img
    leftvar=np.matmul(norm_img,norm_img.T)
    eigvals,eigVects = np.linalg.eig(leftvar)
    index=np.argsort(-eigvals)
    lefteigvectors=np.zeros((k,trainset.shape[0]))
    for i in range(k):
        lefteigvectors[i,:]=eigVects[:,index[i]]
    #print(lefteigvectors)
    feature=np.matmul(lefteigvectors,norm_img)
    feature=(feature.T/np.linalg.norm(feature,axis=1)).T
    return avg_img, feature, norm_img
```

==========

二、人脸识别：

```
def rep_face(image, avg_img, eigenface_vects, numComponents = 0):
    diff=image-avg_img
    representation=np.matmul(diff,eigenface_vects[:numComponents,:].T)
    numEigenFaces=numComponents
    return representation, numEigenFaces
```

三、人脸重建：

```
def recFace(representations, avg_img, eigenVectors, numComponents, sz=(112,92)):
    face=(np.matmul(representations,eigenVectors[:numComponents,:])+avg_img).reshape(sz)
    return face, 'numEigenFaces_{}'.format(numComponents)
```

四、实验结果

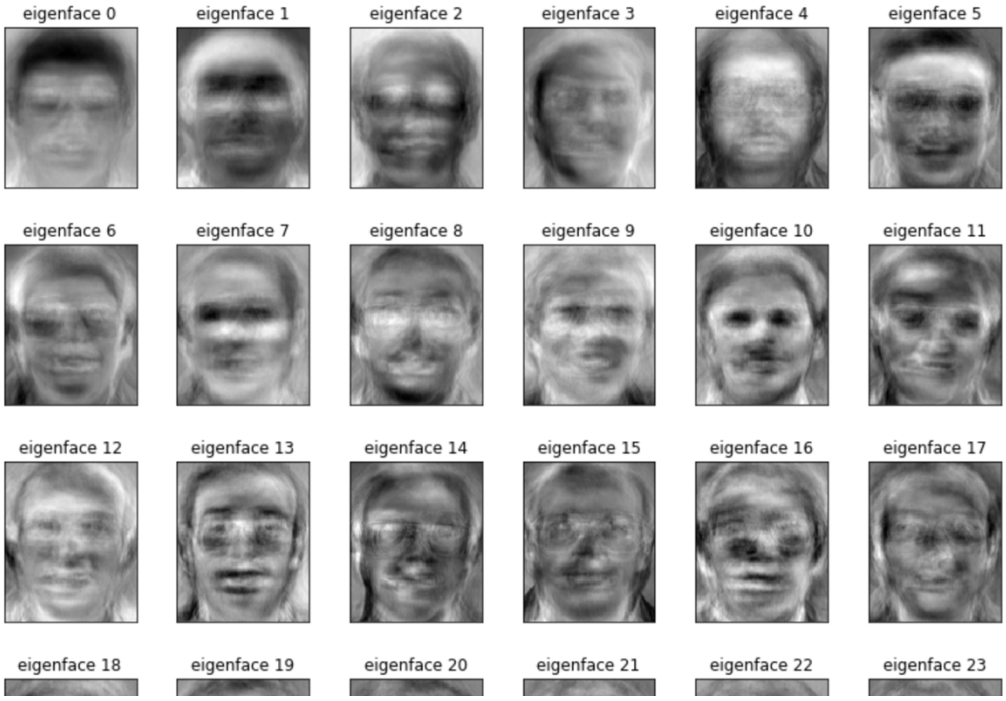
人脸识别准确率：91.25%

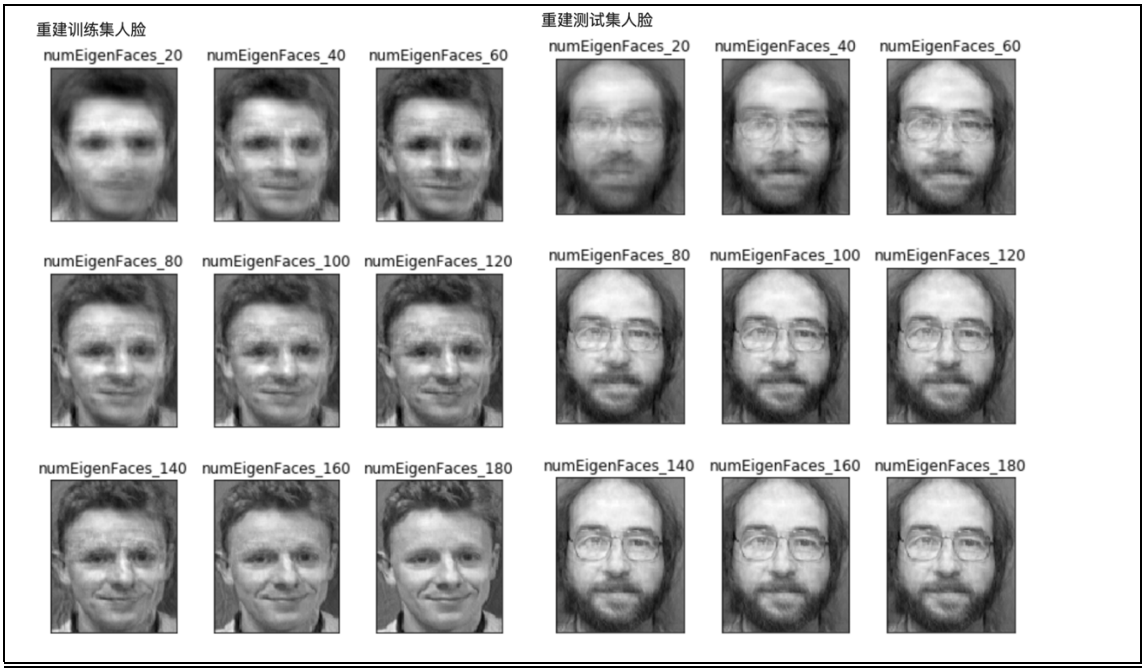
```
print("人脸识别准确率: {}%".for
```

```
人脸识别准确率: 91.25%
```

==========

生成特征人脸图像：





五、总结

一开始我的实验重建出来的人脸一直是一团马赛克，看不出具体的人脸特征。经过排查我后来发现在求特征向量时，调用的 numpy.linalg.eig 函数所返回的 array 是以一列为一个特征向量的，而我把一行当成了特征向量，因此并没有把人脸投影到方差最大的维度里，导致人脸的分类失败。

之后经过改正我发现重建人脸的实验结果中对于训练集的人脸图像重建几乎完美复建，而测试集的人脸重建效果却波动较大，有部分测试结果的人脸重建后依然较为模糊，难以看清其本征的样貌。要提高在测试集的表现我觉得可以扩大人脸的数据库，从而提高分类的准确率。