《计算机视觉》实验报告

姓名：林艺珺 学号：18120189

实验 3

任务 1

在 ATT 数据集上进行 Eigenfaces 方法的实验，
用每个人的前 9 张图片进行训练，最后 1 张作为测试
可用 OpenCV 自带的 EigenFaceRecognizer 函数进行实验

a）核心代码

个人笔记本为 MacOS 平台，选择开发语言为 Python，IDE 为 JetBrains 的 Pycharm。

读取训练数据

读取文件时使用 Python 的 os 模块，在 sorted 状态下，图片的排序为 1-10-2-3-4-5-6-7-8-9，代替计数，忽略编号为 2 的读取。返回包含图片数组和标签数组的 tuple 类型供主函数使用。

```
def read_person():
    img = []     # 创建一个图片数组
    lab = []     # 创建一个标签数组
    for personNum in range(1, 41):
        i = 0
        path = './att_faces/s' + str(personNum)
        # 列出目录中的所有文件，逐个从文本文件中读取点
        for filePath in sorted(os.listdir(path)):
            fileExt = os.path.splitext(filePath)[1]
            if fileExt in ['.pgm']: # 读取图片
                imagePath = os.path.join(path, filePath)
                im = cv.imread(imagePath, 0)
                i += 1
                if im is None:
                    print('image:{} not read properly'.format(imagePath
                        ))
                else:   # 将读取的图片添加到图片数组
                    if i != 2:
                        img.append(im)
                        lab.append(personNum)
        print('Person ' + str(personNum) + ' ' + str(len(img)) + '
            files read.')
    return img, lab      # 返回一个tuple类型
```

主函数

```
if __name__ == "__main__":
    retval = cv.face.EigenFaceRecognizer_create()
    # 读取并训练
    data = read_person()
    retval.train(np.asarray(data[0]), np.asarray(data[1]))
    # 创建一个测试图片数组
    ten = []
    # 读取测试用图片
    for personNum in range(1, 41):
        test = cv.imread('./att_faces/s' + str(personNum) + '/10.pgm',
            0)
        ten.append(test)
        if personNum == 40:
            print('All test files read.')
    # 测试并输出结果与置信度
    print('Here is the RESULT:')
    for testPersonNum in range(0, 40):
        result = retval.predict(np.asarray(ten[testPersonNum]))
        print('Test Image Data is from Person '+str(testPersonNum + 1))
        print('The Recognition Result is Person '+str(result[0])+',
            with confidence '+str(round(result[1],4)))
```

b）实验结果截图

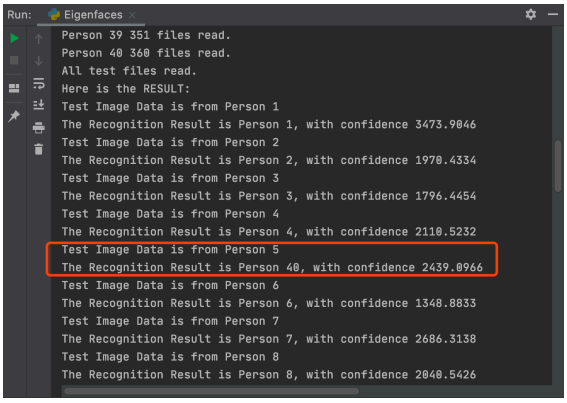


图 1: 结果图片 1

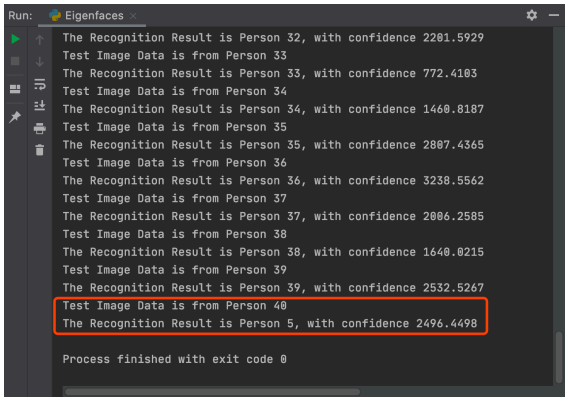


图 2: 结果图片 2

c）实验小结

自己还是很不习惯阅读英文文档，也没有足够了解 OpenCV 官方文档的组织结构。一开始点进 Eigenface 之后根本无法搜寻到相关函数和功能，只得求助中文网站，但相关资料少之又少，最终不得不回到官方网站，这才发现 Eigenface 是一个子功能，其相关内容都在父级内容的展示里。解决几个格式问题后，最终成功完成任务。运用 os 模块本来是为了方便读取，但是被排序坑了一

下，一直没想明白为什么置信度都是 0，后来根据经验意识到排序并不是数值的从大到小，于是解决了问题。实验的结果在 5 和 40 之间出现了偏差，返回查看素材并未直观发现出错的原因，可能说明 Eigenface 在 OpenCV 库中的性能为一般水准，并不完全准确。

任务 2

对某张人脸，用不同个数的主成分重建，并显示重建效果

a）核心代码

个人笔记本为 MacOS 平台，选择开发语言为 Python，IDE 为 JetBrains 的 Pycharm。使用机器学习的 sklearn 中的降维函数 PCA 实现实验内容。

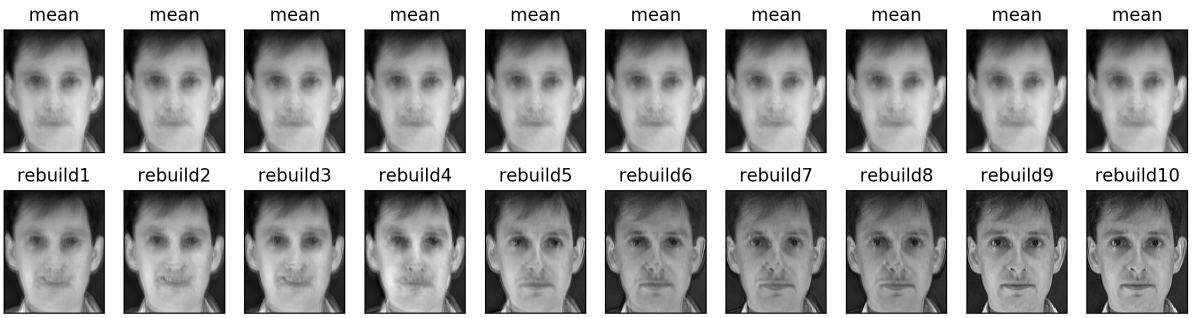
```
import os
from sklearn.decomposition import PCA
import cv2 as cv
import numpy as np
from matplotlib import pyplot as plt


path = './att_faces/s24'     # 文件路径
img = []                     # 图片数组
X = []                       # 图片转成列向量数组
cnt = 0                      # 图片数量
# 列出目录中的所有文件，逐个从文本文件中读取点
for filePath in sorted(os.listdir(path)):
    fileExt = os.path.splitext(filePath)[1]
    if fileExt in ['.pgm']:
        # 读取图片
        imagePath = os.path.join(path, filePath)
        im = cv.imread(imagePath, 0)
        s = im.shape
        dots = int(s[0] * s[1])
        img.append(im)
        X.append(im.reshape(1, dots)[0])
        cnt += 1

# OpenCV自带的PCA似乎没有Python支持
# 使用sklearn的PCA进行实验
for n_comp in range(1,11):  # 为对比结果建立循环
    X = np.asarray(X)       # 转换
    pca = PCA(n_components=n_comp)  # 建立PCA参数
    pca.fit(X)              # 进行PCA
    XS = pca.fit_transform(X)   # 降维
```

```
    mean = pca.mean_.reshape((s[0], s[1]))  # 均值并重设为原始大小
    rebuild = pca.inverse_transform(XS[0])  # 使用inverse_transform重建
        24人的第一张图
    rebuild = rebuild.reshape((s[0], s[1])) # rebuild恢复原始大小
    plt.figure("PCA")
    plt.subplot(2, cnt, n_comp)
    plt.imshow(mean.astype(np.float32), cmap='gray')
    plt.title("mean")
    plt.xticks([])
    plt.yticks([])
    plt.subplot(2, cnt, n_comp+cnt)
    plt.imshow(rebuild.astype(np.float32), cmap='gray')
    plt.title("rebuild"+str(n_comp))
    plt.xticks([])
    plt.yticks([])
plt.show()
cv.waitKey()
```

b）实验结果截图



c）实验小结

依然使用上一次突出对比的 plt 来显示结果。不添加 cmap='gray' 会导致颜色异常输出，是 plt 和 OpenCV 结合使用必然碰见的问题，以后需要注意。

使用的是 sklearn.decomposition.PCA。

fit(self, X, y=None) 每个需要训练的算法都会有 fit() 方法，其实就是算法中的“训练”这一步骤。因为 PCA 是无监督学习算法，此处 y 自然等于 None。fit(img)，表示用数据 img 来训练 PCA 模型。函数返回值为调用 fit 方法的对象本身。

fit_transform(self, X, y=None) 用 X 来训练 PCA 模型，同时返回降维后的数据。transImg=pca.fit_transform(img)，transImg 就是降维后的数据。

inverse_transform(self, X) 将降维后的数据转换成原始数据。pca.inverse_transform(transImg)的返回值即是重建后的数据。

没有想到最终的代码其实还挺短的，中间花的时间其实非常长。最开始使用 API 的时候没有把相应的参数与原理中的对应上，第一版完成的实验是有错误的，一直觉得不太对，终于思考出来问题所在并且重新修改。没想到自己在用库的实验题上也花费了很多时间，说明自己的计算机触觉还是很不行，需要阅读更多的内容来培养。

实验的过程中也尝试了自己写 inverse_transform 的部分，但是其中一些参数的格式问题导致没有进行下去，时间限制原因只能现行上交作业以保基本盘。

PCA 降维的原理了解得差不多，但实际操作，包括寻找已有的资源和自建都遇到了困难。在这方面现成的资料不多，大多需要自己重新探索。但本周时间安排紧张，因此深入研究不够，选做的题目只完成了一小部分。五一假期若有空，将继续完成，以加深对 PCA 的理解。在 YouTube 上观看了几个原理讲解视频，配合 MATLAB 和 Python 的代码进行学习，受益匪浅。