数字图像处理 Final Project 人脸识别

22366010 淡昱翔

1.代码安装使用说明

算法属于传统机器学习算法，基于 matlab 编写，需要安装 matlab 的深度学习工具包 deep learning toolbox 作为前置。程序包含一个脚本文件和人脸数据集文件夹，两个文件（夹）需要放在一个根目录下。其中，脚本文件的内容可分为两个函数实现（数据集分割和人脸识别与测试）和一个主程序，直接在 matlab 中运行脚本文件即可。

2.算法描述

算法主要分为三个部分：

1）数据集分割：

提供的数据集是剑桥大学 ORL 人脸数据库，其中有 40 人的每人各 10 张共 400 张 256 灰度级的 92x112 大小的灰度图像，数据集文件夹命名为 att_faces。

在数据分割部分，我们首先将图像读取成对应维数的双精浮点数向量矩阵，其中，对每个人的 10 张图像中的随机 5 张作为训练集，剩余 5 张作为测试集，读取其标签。分割完成以后，为了减少特征提取的工作量，我们将 5 张训练图片暴力相加后取平均得到这个人的一张特征图像（向量矩阵）。

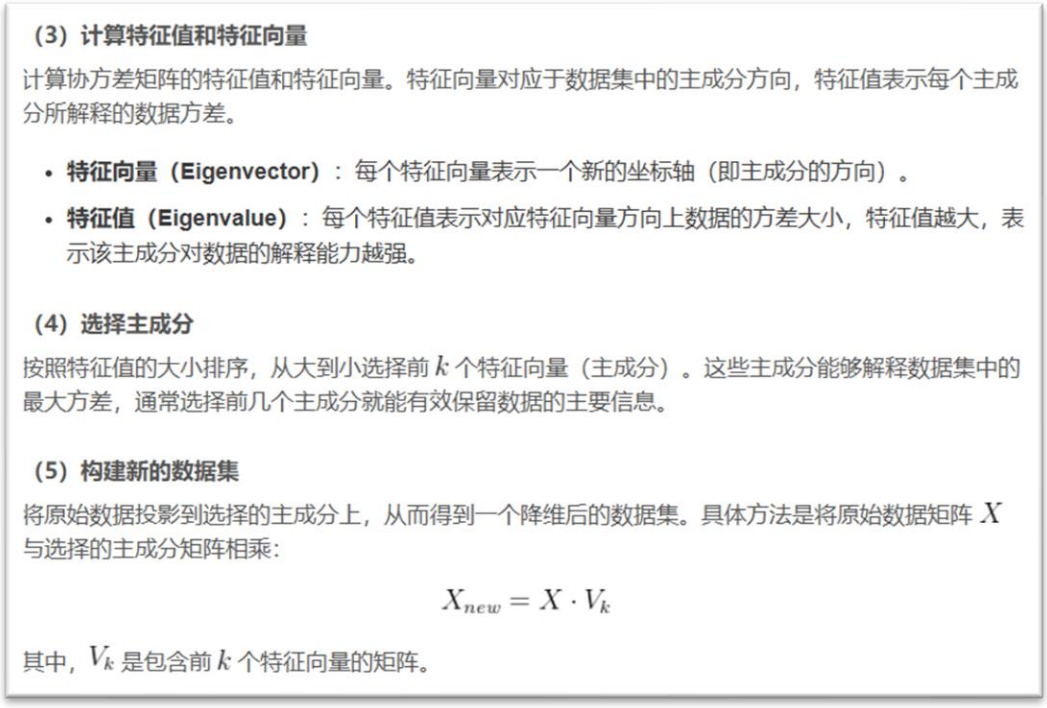
```
function [train_data, test_data, label] = divide_data(n)
    train_data = zeros(40, 10304);   % 将每个图像展平即为92 * 112 = 10304维
    test_data = [];
    label = [];

    % 遍历每个人
    for i = 1:40
        randnum = randperm(10, n);  % 从每个人的10张照片中选出n=5张作为训练集
        for j = 1:10
            img = reshape(double(imread(['./att_faces/s', num2str(i), '/', num2str(j), '.pgm'] ...
                                        )),1,10304);
            if ismember(j, randnum)
                train_data(i, :) = train_data(i, :) + img;    % 填充训练集i行对应第i个人
            else
                test_data = [test_data; img];
                label = [label; i];   % 记录测试集和测试集对应的标签
            end
        end
    end
    train_data = train_data / n;    % 将训练图像平均成一张特征图像
end
```

Fig1. 数据集分割函数

2）人脸图像特征提取、识别与测试：

这个函数首先得到分割好的训练用特征图像和测试集，使用 pca 函数对特征图像进行

特征分析，得到特征向量矩阵。再把特征向量矩阵投影到原特征图像上即得到降维的数据。



（3）计算特征值和特征向量

计算协方差矩阵的特征值和特征向量。特征向量对应于数据集中的主成分方向，特征值表示每个主成分所解释的数据方差。

- **特征向量（Eigenvector）**：每个特征向量表示一个新的坐标轴（即主成分的方向）。
- **特征值（Eigenvalue）**：每个特征值表示对应特征向量方向上数据的方差大小，特征值越大，表示该主成分对数据的解释能力越强。

（4）选择主成分

按照特征值的大小排序，从大到小选择前 k 个特征向量（主成分）。这些主成分能够解释数据集中的最大方差，通常选择前几个主成分就能有效保留数据的主要信息。

（5）构建新的数据集

将原始数据投影到选择的主成分上，从而得到一个降维后的数据集。具体方法是将原始数据矩阵 X 与选择的主成分矩阵相乘：

$$X_{new} = X \cdot V_k$$

其中，V_k 是包含前 k 个特征向量的矩阵。

Fig2.PCA 算法原理

对于得到的每人的训练出的特征数据，将其与测试集图片计算 2 范数（欧氏距离），把最小的即最相似的也即预测出的目标返回。对于返回的值，验证其是否是正确的目标，若正确则正确数量加一。遍历完所有图像后，计算整体的正确率。在如下图的代码段中，M=m\5=40，故：$正确率 = \frac{识别正确的图像数}{m = 200}$ 。

```
function accuracy = Identify(train_data, test_data, label)
    count = 0;
    [M, ~] = size(train_data);  % 读取训练集大小,M=40
    [m, ~] = size(test_data);   % 读取测试集大小

    [coeff, ~, ~] = pca(train_data);    % 使用pca训练数据
    pca_train = train_data * coeff; % 把训练的数据映射到k维空间上

    for i = 1:m
        distance = [];
        test_pca = test_data(i,:) * coeff;
        for k = 1:M   % 遍历每个人像的特征向量
            distance = [distance,norm(pca_train(k,:) - test_pca, 2)];   %  计算与被测图片距离(2范数)，距离越小越相似
        end

        [~, index] = min(distance);   % 从所有可能种类中找出相似度最高的向量
        if index == label(i)
            count = count + 1;
        end
    end
    accuracy = count / m;   % 直接计算整体准确率
end
```

Fig3. 特征提取、识别和验证函数

3）主运行程序:

由于我们采用的是传统线性机器学习算法解决小数据集的分类任务，而且在数据集的分

割上是随机选取的，故得到的准确率结果也是包含极大随机性的。故我们设置多轮学习，把每次学习的结果取极大和平均作为更健壮的参考结果。

```
rounds = 20;    % 训练轮数，避免训练的随机性
for i = 1:rounds
    [train_data, test_data, label] = divide_data(n);
    acc = Identify(train_data, test_data, label);
    accuracy = [accuracy, acc];
    % 打印当前迭代的准确率
    fprintf('Iteration %d Accuracy: %.4f\n', i, acc);
end

max_acc = max(accuracy);    % 找出最大的正确率
mean_acc = mean(accuracy);  % 找出平均正确率
fprintf('max=%f, min=%f\n',max_acc, mean_acc);

figure;

x = 1:rounds;
plot(x, accuracy,'r--o');
% plot();
ylim([0.8,1]);
xlabel('times');
ylabel('accuracy');
title(['max=',num2str(max_acc), 'mean=', num2str(mean_acc)]);
```

Fig4. 主运行程序代码（部分）

3. 结果

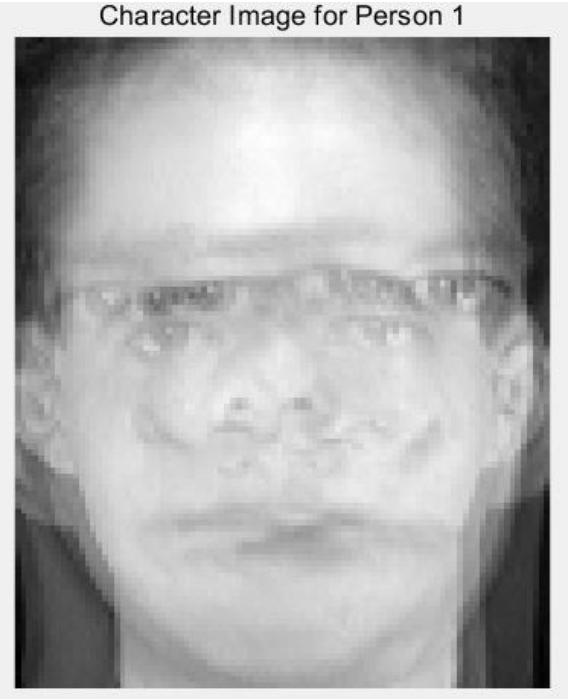


Fig5. 某人物特征图像

参数设置：n=5　rounds=20

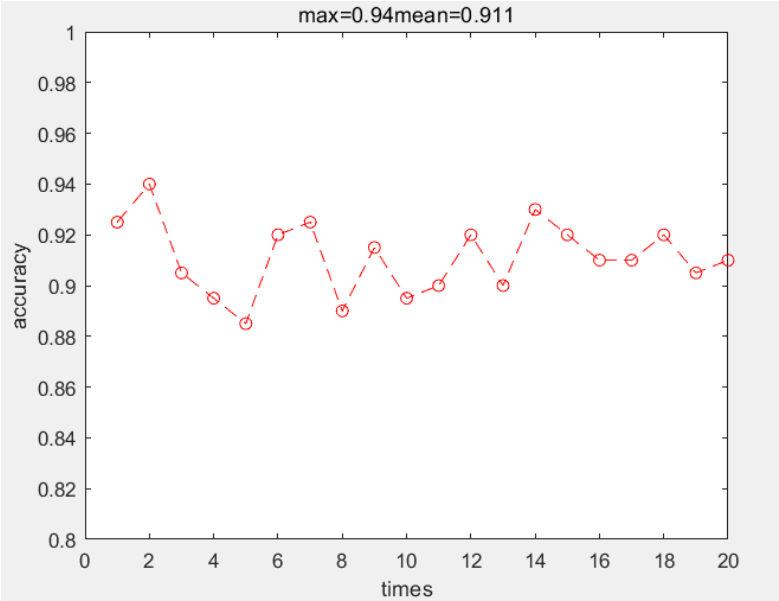


Fig6.训练结果

结果分析：20 轮的训练下，最高的准确率为 94%，平均准确率 91.1%，算法表现较好。（完整代码和结果图像见压缩包）