**南 开 大 学Python语言程序设计大作业论文**

中文题目：基于mediapipe的人脸识别

学 号： 2013852

姓 名： 郁万祥

年 级： 2020 级

专 业： 软件工程

学 院： 软件学院

指导教师： 王斌辉

完成日期： 2022 年 12 月 30 日

摘 要

随人脸识别技术的快速发展，人脸识别在生产生活中的使用场景十分广泛，人脸录入与识别功能在互联网买机票、车票，医院挂号，政府惠民工程项目，以及各种证券开户、电信开户、互联网金融开户都会用到。

MediaPipe是一款由Google Research开发并开源的多媒体机器学习模型应用框架。在谷歌，一系列重要产品，如、Google Lens、ARCore、Google Home以及，都已深度整合了MediaPipe。这款框架，可以使用python库进行直接地掉用，可以十分方便的使用。

本项目旨在通过opencv，python，mediapipe以及flask等技术搭建一个简单的人脸信息录入与识别系统。用户进行采集并录入人脸信息库中，已经录入人脸信息的用户可以通过媒体设备进行人脸识别。

关键字：人脸识别、python、mediapipe、SVM、PCA

Abstract

With the rapid development of face recognition technology, face recognition in production and life use scenarios are very extensive, face input and recognition function in the Internet to buy air tickets, bus tickets, hospital registration, government benefiting the people engineering projects, as well as a variety of securities account opening, telecommunications account opening, Internet financial account opening will be used.

MediaPipe is a multimedia machine learning model application framework developed and open sourced by Google Research. At Google, a number of important products, such as Google Lens, ARCore, Google Home, and MediaPipe, have been deeply integrated. This framework can be directly used using the python library, which can be very convenient to use.

This project aims to build a simple face information entry and recognition system through OpenCV, Python, MediaPipe and FLASK. Users collect and enter face information database, and users who have entered face information can perform face recognition through media devices.

Keywords：face recognition，python，mediapipe，SVM，PCA

**目 录**

[第一章 团队介绍](#_Toc29216)

[1.1 团队成员介绍](#_Toc8372)

[1.2 团队成员工作重点](#_Toc9392)

[第二章 系统需求分析](#_Toc8133)

[2.1 引言](#_Toc14042)

[2.2 项目概述](#_Toc16458)

[2.3 系统描述](#_Toc16609)

[第三章 系统概要设计](#_Toc3477)

[3.1 概要设计](#_Toc3198)

[3.2 架构设计](#_Toc12136)

[第四章 系统实现](#_Toc8009)

[4.1 图像处理](#_Toc15999)

[4.2 SVM模型](#_Toc32223)

[4.3 PCA降维](#_Toc5478)

[4.4 mediapipe人脸检测](#_Toc1998)

[4.5 网页模块](#_Toc6091)

[第五章 系统测试](#_Toc30781)

[5.1 引言](#_Toc7905)

[5.2 测试需求分析](#_Toc22282)

[5.3 计划测试过程](#_Toc9923)

[第六章 系统安装手册](#_Toc6576)

[6.1 安装说明](#_Toc31379)

[第七章 系统用户手册](#_Toc19226)

[7.1 引言](#_Toc24419)

[7.2 用途](#_Toc12962)

[7.3 运行需求](#_Toc25478)

[7.4 其它需求](#_Toc35)

[7.5 输入与输出](#_Toc20190)

1. **团队介绍**
   1. **团队成员介绍**

郁万祥

* 1. **团队成员工作重点**

1、设计项目整体实现思路，并且完成项目框架及功能的编写。

2、开发人脸识别模块，构建PCA+SVM模型。

3、进行前端Web开发，实现程序的可视化。

4、开发人脸检测模块，实现mediapipe人脸检测。

1. **系统需求分析**

**2.1引言**

2.1(1) 编写目的

本文档为了让用户和软件开发者双方对开发软件的初始规定有一个共同的理解。使之成为整个项目开发中系统开发设计的依据，也是项目后期系统测试依据。

本文档定义人脸录入与识别的运行环境、功能需求以及各个子系统的功能分析，指导将来系统升级或重写设计及编码。

2.1(2) 读者对象

本文档可作为系统设计人员，程序员，测试人员，文档编写人员，系统维护人员，系统使用人员的参考资料。

**2.2项目概述**

随人脸识别技术的快速发展，人脸识别在生产生活中的使用场景十分广泛，人脸录入与识别功能在互联网买机票、车票，医院挂号，政府惠民工程项目，以及各种证券开户、电信开户、互联网金融开户都会用到。

本项目旨在通过opencv，python以及flask等技术搭建一个简单的人脸信息录入与识别系统。

用户进行采集并录入人脸信息库中，已经录入人脸信息的用户可以通过媒体设备进行人脸识别。

**2.3系统描述**

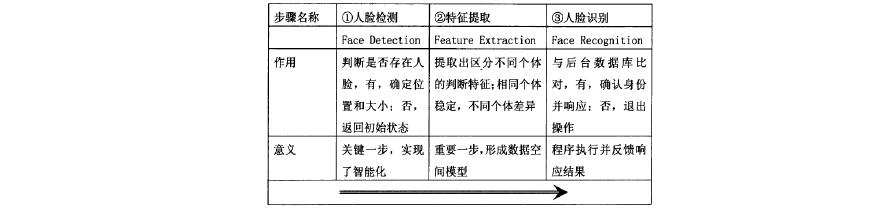
人脸识别作为生物特征识别技术中最为直观的一种识别方法，具有使用友好、操作隐蔽、使用简便、定位准确、经济实惠及系统扩展性好等其他生物识别技术不可替代的优势。利用计算机进行完全自动的人脸识别却存在着许多困难，这主要表现在以下凡个方面:首先，采集的面部表情只是某一时刻的，在进行人脸识别的时候，用户的姿态表情等会随意出现，对识别造成干扰;其次，人的面貌会随着时间的变化而变化，需要对采集的人脸数控库不时更新;再次，在采集图像时，用户的发型、眼镜等装饰物会对人脸特征的提取造成遮挡;最后，光线强度、距离角度以及采集所用的硬件的参数不同，对获取的人脸特征也会造成影响。

归一化(旋转、裁切文件，统一标准)、图像灰度化技术(将彩色图像转换为灰度图像以加快处理速度)、直方图均衡化技术(调整图像的灰度分布，改善质量)、中值滤波技术(去除图像中的干扰噪声)等方面，保证人脸识别的质量。

经过图像预处理之后得到的图像质量明显改善，干扰项目基本剔除，就进入到模式识别过程中很关键的一步，即特征提取。提取的特征点的好坏将直接影响到人脸识别的质量，因为目前我们大多还是使用图像的关键特征点的像素信息来描述人脸特征。如果提取的特征对不同样本出现很大的差异性区分，而对相同样本的不同状态表现出很接近、最好是相同的人脸特征，就说明当前所选择的特征提取方法是有效地，所选择的分类器对该特征的分类效果越好，人脸识别的识别效率也就越高

人脸检测模块的设计是系统设计的关键，将直接影响人脸识别系统的性能。现在的主要工作有两项:其一，检测获取的图像中是否有人脸的存在，此时需要对大量的人脸与非人脸样本进行训练，获取图像之间的细节差异;其二，检测确定人脸的位置和外观，尽量详细地描述出人脸的整体特征和细节特征。主要包括眼睛和鼻子这两个最为重要的器官特征，根据这两个主要特征区域信息，考虑嘴巴、耳朵、毛发等观测区域的特征信息，同时还要考虑到特征的提取要对光线、表情、视角等干扰因素有比较好的健壮性。

对于本系统而言，人脸检测目标区域背景相对比较简单，人脸数量和大小可以控制，大大降低了人脸检测的难度。首先，对人脸特征搜索定位;其次，对人脸区域几何归一化处理，再次，对检测到的人脸进行特征提取



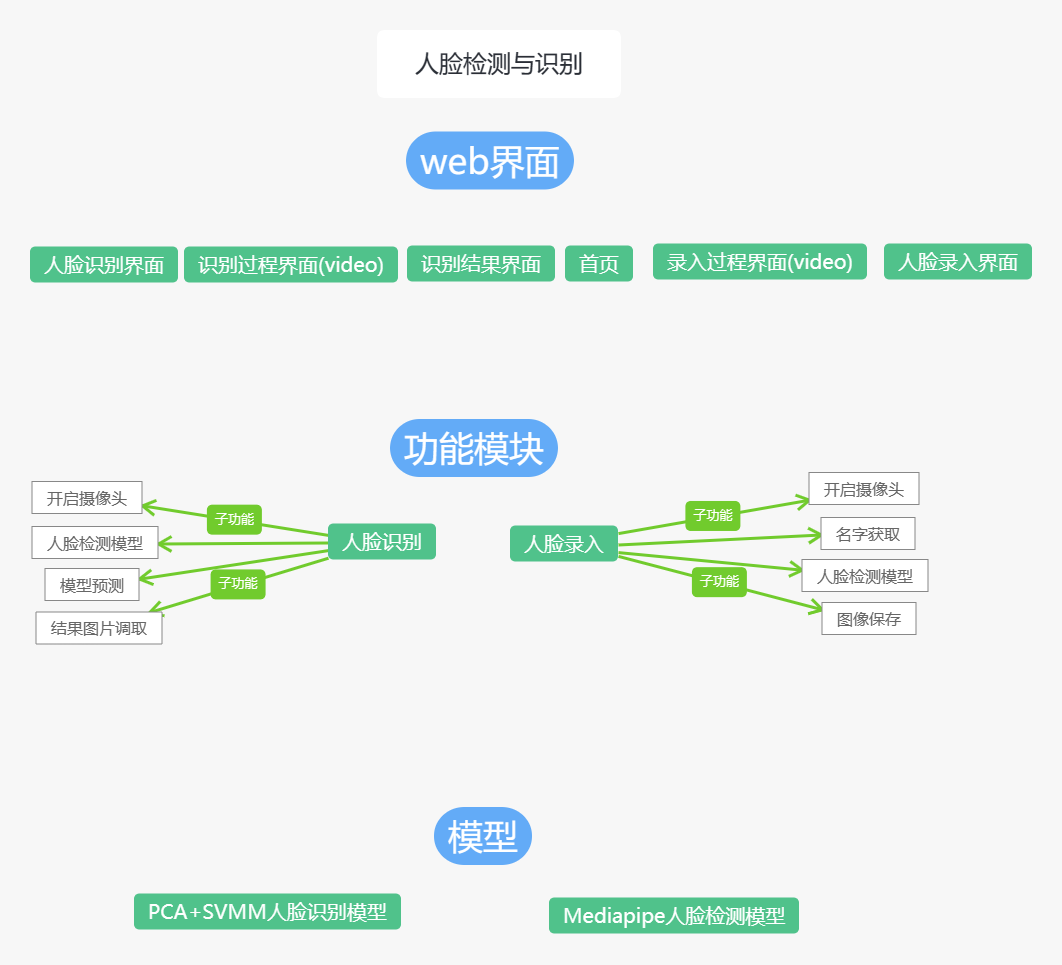
1. **系统概要设计**

**3.1 概要设计**

项目模型可分为人脸检测模型，人脸分类模型两大模块；项目具体功能分为人脸录入和人脸识别两大功能。

其中，人脸的录入功能通过使用人脸检测模型以及对文件进行写操作进行实现；人脸识别通过人脸检测模型和人脸分类模型进行实现。

**3.2 架构设计**



1. 系统实现

**4.1 图像处理**

读取数据：

使用os.listfir()访问目标路径下的每一个jpg文件，并将最近一级的文件夹名作为其label，通过cv2.imread()读取灰度图像

处理数据：

若维度太高算法复杂度也会随之升高，故将得到的一通道的灰度图像进行PCA降维，从横 轴和纵轴两个方向进行降维

获取数据：

因为有多张照片，而SVM模型需要的是二维的数据，故将得到的降维后的图像展平为一维， 每张图像作为一行，有多少张图片就有多少行。

**4.2 PCA降维**

因为处理后传入的图像为92\*112的大小，若将该值全部传入到SVM模型中，后续处理大量照片时会因数据量太大而导致程序运行速度慢，而SVM本身就是小样本的学习方法，故我们将图像降低为50\*50的大小，既不至于损失太多数据，又降低了数据量。

**4.3 SVM模型**

因为本程序面向的是小规模的数据，并且因为设备原因需要追求程序的运行速度，故最终采用SVM来训练数据。SVM的构建主要使用了skearn中的SVC模型进行实现。最总在项目过程中实现实时继续训练的效果。

**4.4 mediapipe人脸检测**

(一) 、原理摘要

MediaPipe人脸检测是一种超快的人脸检测解决方案，具有6个landmarks和多人脸支特。它基于BlazeFace,这是一种轻量级且性能良好的人脸检测器，专为移动GPU推理量身定制。检测器的超实时性能使其能够应用于任何需要准确的面部感兴趣区域作为其他特定任务模型输入的实时取景器体验，例如3D面部关键点或几何估计（例如MediaPipe Face Mesh)、面部特征或表情分类以及面部区域分割等。BlazeFace使用了一个轻量级的特征提取网络，其灵感来自于MobileNetV1W2,但与MobileNetV1V2不同，它是一个gpu友好的锚定方案，由Single Shot MultiBoxDetector(SSD)改进而来，使用某种策略(an improved tie resolution strategy)替代非极大值抑制的。

1. 、解决方案的API

参数配置：

·MODEL\_SELECTION:索引为0或1的整数。使用0选择一个短距离模型，最适合距离相机2米以内的人脸，1选择一个全距离模型，最适合距离相机5米以内的人脸。对于全距离选项，采用稀疏模型来提高推理速度。如果未指定，默认为0。貌以此参数已经被淘汰。

·MIN\_DETECTION\_CONFIDENCE:来自人脸检测模型的最小置信值(O.O,1.O),以便将检测视为成功。默认为0.5。

输出：

·DETECTIONS：检测到的人脸的集合，其中每个人脸都表示为一个检测原始消息，其中包含一个边界框和6个关键点（右眼、左眼、鼻尖、嘴巴中心、右耳和左耳）。边界框由xmin和width(由图像宽度归一化为[0.0,1.0)以及ymin和neight((由图像高度归一化为[0.0,1.0)组成。每个关键点由x和y组成，分别通过图像宽度和高度归一化为[0.0,1.0]。

**4.5 网页模块**

该网页项目主要分为人脸录入页面和识别页面，人脸录入页面包括姓名输入，摄像头录入模块。识别页面包括摄像头识别模块。

  用户可以打开人脸录入页面，输入姓名并打开摄像头完成用户人脸录入。在存在用户人脸信息的条件下可以打开人脸识别网页，实现用户人脸识别并显示结果。

    1.前端设计

前端主要采用了HTML标签，CSS样式，JS语言。布局主要采用的是Flex布局，再加上自己对页面样式的改写和优化，基本搭建了项目的前端页面

2.web开发框架

Web开发框架使用flask

 人脸录入模块开发流程：

1. 编写录入html网页并渲染到模板中。
2. 通过表单执行录入操作，可以记录、保存当前用户姓名
3. 操作成功后重定向到摄像头页面，并在视图函数中得到相应的值。<img src="{{ url\_for('video\_feed1') }}>
4. 摄像头采用flask流式传输，流式传输的实现是采用 **生成器函数（yeild关键字）** ，**img**这个标签，它定义了一张图片元素，其URL是**/video\_feed**。从Flask应用程序代码的Line17-20可以知道，**/video\_feed**是由一个**video\_feed()**方法提供服务的，它返回的是一个multipart应答。这个应答的内容是由生成器函数**gen()**提供的。而**gen()**函数就是不停地从camera里获取一帧一帧的图片，并通过生成器返回给客户端。客户端浏览器在收到这个流媒体时，会在**img**标签定义的图片里，逐帧地显示图片，这样一个视频就播放出来的。目前市面上绝大部分浏览器都支持这个功能。

人脸识别模块开发流程：

1. 编写录入html网页并渲染到模板中。
2. 录入成功后重定向到摄像头页面，并在视图函数中得到相应的值。<img src="{{ url\_for('video\_feed0') }}>
3. 摄像头采用flask流式传输，同时将识别到的用户名返回后端
4. 显示结果从后端获取用户姓名与录入时的图片与

实现流程：

1. 后端(视图层)编写API
2. 前端(模板层)调用API
3. 渲染页面，绑定功能
4. **系统测试**

**5.1 引言**

为考量程序是否能按照预期正常运行，我们将在脱离于开发的设备上部署程序进行验证

**5.2 测试需求分析**

分别测试人脸录入和识别功能能否正常、正确的运行

**5.3 计划测试过程**

创建一个虚拟环境，将项目部署到这个环境中，运行程序，检查前端和后端是否正常连接。至少进行十个人脸的录入，并检查是否能够准确的

1. **系统安装手册**

**6.1 安装说明**

运行环境： python3.8

运行IDE：pycharm

运行所需下载类库： mediapipe、tensorflow2.9.1 、opencv-contrib-python4.6.0.66、Flask2.2.2、numpy1.23.1 等（可按照IDE提示进行下载类库）

项目源码不需做任何改动，直接运行app文件，即可开启服务器，点击进入控制台所显示的网址，即可使用项目所有功能。

注意：1、只有进行两次以上不同人脸录入后才能够使用人脸识别功能（SVM模型规定分类数目>=2）

1. 用户输入名字不能包含中文
2. **系统用户手册**

**[7.1 引言](#_Toc24419)**

为了实现动态的人脸录入和人脸检测的任务，本项目使用机器学习svm支持向量机的算法思想以及深度学习的理念，利用python语言编写，开发的集成环境为pycharm，实现了在web网站中进行人脸的识别和检测任务。

**[7.2 用途](#_Toc12962)**

本项目可以用于多种涉及到人脸识别和人脸检测的任务中，包括简单的人脸识别程序的开发，人脸解锁，人脸识别进行签到等。

**[7.3 运行需求](#_Toc25478)**

该项目需要Python的语言环境，所依赖的库文件如下所示：

Mediapipe==0.9.0.1

Flask==2.2.2

keras==2.9.0

matplotlib==3.5.2

numpy==1.23.1

opencv\_python==4.6.0.66

Pillow==9.2.0

pycocotools==2.0.4

scikit\_learn==1.1.2

scipy==1.8.1

tensorflow==2.9.1

tqdm==4.64.0

**[7.4 其它需求](#_Toc35)**

该项目中的模型为训练好的模型，如果想要重新训练模型，需要有GPU的深度学习的开发环境。

**[7.5 输入](#_Toc20190)与输出**

1.运行根目录下的app.py文件，控制台会弹出网址，该网址即为项目人脸识别系统的web前端网站。

2.进入网站后，会有两个按钮分别为“人脸识别”和“人脸录入”。

3.初次使用该系统的用户需要首先点击“人脸录入”按钮，进行人脸信息的录入。点击按钮后进入录入页面，点击开始录入，会弹出“姓名录入”的提示框，输入姓名（注意：需为英文名称），点击提交即可。提交后会开启摄像头进行人脸信息的录入。

4.当完成信息的录入之后，注意至少需要两个及以上的人脸信息录入才可开始人脸的识别，点击“人脸识别”,点击“开始识别”，会自动开启摄像头，对人脸进行识别，识别完成后会在图片上显示识别得到的结果，然后可以选择点击显示结果，会出现一个小头像和对应的名字。

5.项目具体的使用方式如上。