

# 网络编程课程设计项目文档

项目名称： 《智慧教室人脸考勤系统》

项目负责人： 郭赞

指导老师： 赵晓京

# 目录

网络编程课程设计项目文档 .....	1
一、需求分析 .....	3
1.1 项目背景 .....	3
1.2 目标人群 .....	3
1.3 小组分工 .....	3
二、设计框架 .....	4
2.1 总体功能 .....	4
2.2 客户端功能 .....	4
2.2.1 人脸识别模块 .....	4
2.2.2 网络通信模块 .....	6
2.2.3 数据可视化模块 .....	7
2.3 服务器功能 .....	8
2.3.1 数据库模块 .....	8
2.3.2 签到管理模块 .....	9
2.3.3 网络通信模块 .....	10
三、详细结构 .....	11
3.1 整体流程图 .....	11
3.2 数据库说明 .....	12
3.2.1 E-R 图 .....	12
3.2.2 数据字典 .....	13
3.2.3 数据流图 .....	13
3.3 数据加密 .....	14
四、技术难点 .....	14
4.1 图像编码格式转换 .....	14
4.2 嵌入式部署 .....	15
五、系统集成与测试 .....	16
5.1 系统集成 .....	16
5.1.1 3D 建模仿真 .....	16
5.1.2 外壳 3D 打印 .....	16
5.1.3 硬件封装 .....	17
5.2 产品测试 .....	17
六、总结与展望 .....	18

# 一、需求分析

## 1.1 项目背景

长期以来，签到考勤是考查学生出勤率与学习情况的重要方式。目前，中小学与高等学校的老师普遍使用人工考勤，这种传统的人工考勤方式效率低下，且往往存在“代签”现象；使用学习通等第三方签到软件则存在远程签到、代签等弊端，不仅给任课老师带来不必要的负担，还浪费了课堂宝贵的上课时间。

为了解决上述问题，本团队开发了一款基于基于人脸识别的考勤设备：**智慧教室人脸考勤系统**。从教师点名到人脸识别，课堂考勤的升级换代，本质上是身份识别方式的迭代演变。随着技术的成熟，考勤机得到了飞跃式发展，产品形态和应用模式呈现出多元化、智能化的发展态势。对于学校而言，除了考勤效率和安全性能，智能化、在线化、管理精准化等也是考勤机选择的重要考量标准，而人脸考勤系统兼具了这些功能，这也是众多学校与任课老师选择人脸考勤的原因。

人脸考勤系统以人脸为介质，人员经过设备时只需轻轻一瞥，1秒内即可自动识别人员身份并完成签到，实现对人员的精准管理。无需担心错签或漏签问题，只需要刷脸就能秒速签到，大大提升了学生通行效率和体验。该系统成本低廉，复用性强，对未来学校智慧教室的实现与推广具有巨大积极作用。

## 1.2 目标人群

包括中小学、高等院校、企业等各种实行考勤绩效评比的单位。

## 1.3 小组分工

郭赞：嵌入式开发、系统集成

吴晗宁：人脸识别、签到管理系统

刘宛君：数据库

周禹彤：socket 网络通信

陈嫣冉：软件界面设计

## 二、设计框架

### 2.1 总体功能

一种智慧教室人脸考勤系统，其特征在于，基于**客户端-服务器**结构设计，包括**客户端的人脸检测系统**和服务器的**签到管理系统**两部分。

客户端为一台搭载摄像头的嵌入式微型运算设备，具有**基于深度学习 AI 的人脸识别功能**，并通过网络通信将人脸检测图像发送至服务器。

服务器包括学生个人信息数据库，能够通过网络通信接收人脸图像并与数据库中的特征信息进行匹配，完成签到，并将匹配结果与个人信息发送到客户端设备。

该系统的总体功能设计框图，如图 1 所示。

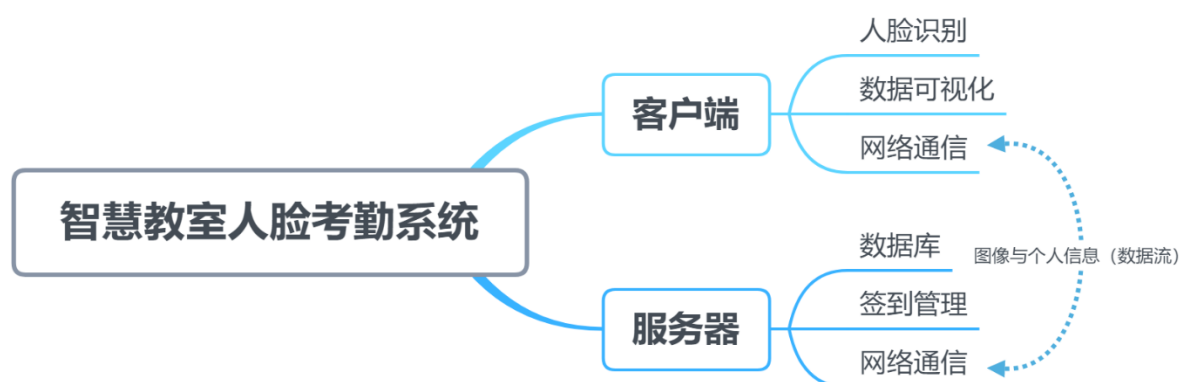


图 1 总体功能设计框图

### 2.2 客户端功能

人脸考勤系统客户端的主要功能有：**人脸识别、网络通信、数据可视化**等。

#### 2.2.1 人脸识别模块

人脸识别模块是基于 face recognition 库实现的，face recognition 是最简单的人脸识别库。人脸识别模块分为**人脸目标检测、人脸特征向量编码以及人脸身份匹配**三个子模块。

人脸目标检测实现方式是通过调用人脸预训练模型进行目标检测，输入摄像头采集的实时画面 `img`，调用 `face_recognition` 的方法 `face_locations`，输出包含人脸坐标点的列表 `face_locations`。

人脸特征向量编码的实现方式是通过调用 `face_recognition` 的方法 `face_encodings`，输入人脸图像 `face` 以及包含人脸图像坐标的 `face_locations`，输出该人脸的 **128 维特征向量**，该向量中包含人脸中五官特有的权重。

人脸身份匹配的实现方式是通过将人脸图像的 128 维特征向量与人脸特征数据库中的特征向量进行逐一对比，通过调用 `face_recognition` 的方法 `face_distance`，计算待测向量与数据库向量的欧氏距离，距离最小即为匹配成功，从而实现人脸识别结果与身份信息的匹配。

上述方法的关键代码实现，如图 2 所示。

```
1 //人脸目标检测函数
2 face_locations = face_recognition.face_locations(img)
3 //人脸特征向量编码函数
4 face_encodings = face_recognition.face_encodings(face, face_locations)
5 //人脸身份匹配函数（计算特征向量之间的欧氏距离）
6 face_dis = face_recognition.face_distance(face_encodings[i], face_encoding)
```

图 2 人脸识别模块关键代码

使用 OpenCV 的 `cv2.VideoCapture` 函数捕获摄像头实时采集的图像，使用 `cv2.resize` 函数对图像进行 **1/4 降采样处理**（降低分辨率有利于减少人脸目标检测时间）；使用 `face_recognition` 的方法 `face_locations`、`face_encodings`、`face_distance` 等方法进行人脸目标检测与定位等一系列处理后，得到人脸识别结果，流程图与效果图如图 3、图 4 所示。

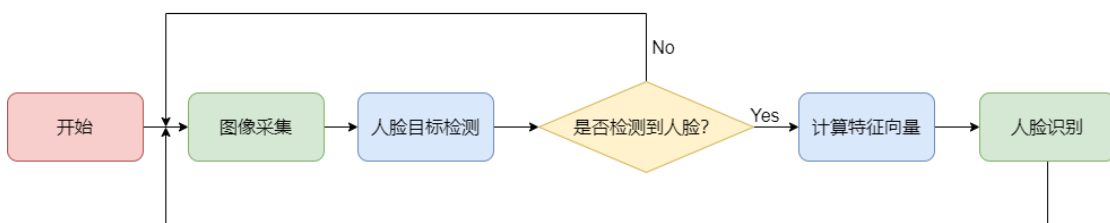


图 3 人脸识别模块流程图

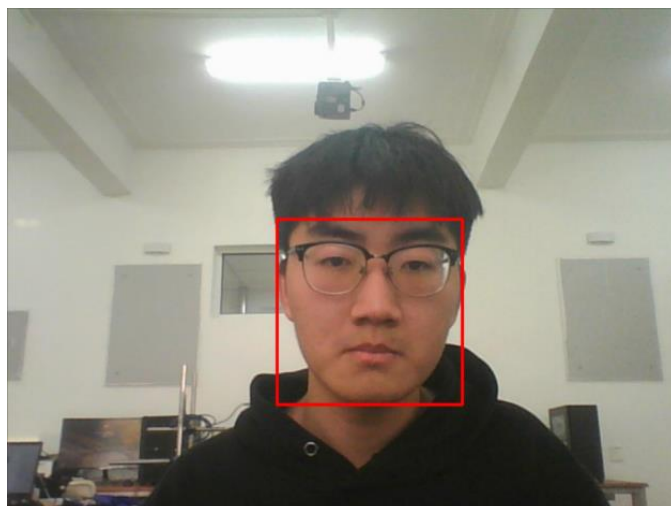


图 4 基于 face recognition 的人脸识别

### 2.2.2 网络通信模块

客户端具备网络通信功能，基于 TCP/IP 协议的 socket 套接字实现，依靠 WIFI 无线网络实现客户端与服务器的连接。客户端的 IP 地址由路由器通过 DHCP 自动分配，主要作用是发送人脸图像到服务器进行个人信息匹配，并接收相应的个人信息，如姓名、班级、学号、专业、签到状态等。

网络通信模块包括**图像数据流编码**、**数据流发送与接收**两个子模块。

**图像数据流编码：**主要实现了将 RGB 三通道人脸图像编码为数据流的功能，主要的实现方法是使用 OpenCV 的 cv2.imencode 函数以及 tostring 函数将人脸图像矩阵转换为 socket 能够发送的数据流形式。

**数据流发送与接收：**通过 socket 建立了对服务器 ip 地址与端口的连接，并通过 socket.send 与 socket.recv 函数实现了数据流基于 WIFI 无线网络的发送与接收功能。网络通信模块工作流程图，如图 5 所示。

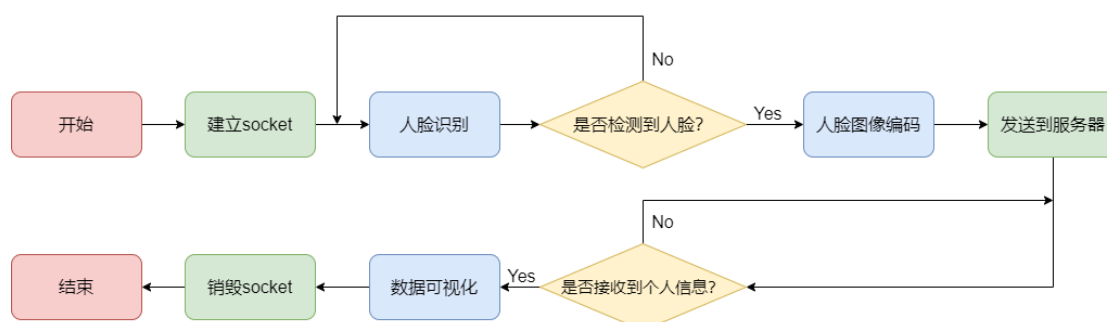


图 5 客户端网络通信模块工作流程图

### 2.2.3 数据可视化模块

该模块主要以软件界面的形式呈现，基于 PyQt5 搭建，主要使用的控件有 QLabel（显示签到状态、当前课程、系统时间、人脸检测结果图以及服务器发送的个人证件照、个人信息等）、QPushButton（控制客户端开关机）、QTime（用于显示系统时间）。

通过 Anaconda 自带的 QTDesigner 进行可视化界面设计，包括控件添加、控件大小设置与空间布局等，保存得到.ui 格式的界面文件。使用外部工具 PyUIC 将 ui 文件转化为 python 代码。

通过 setGeometry 函数实现控件的大小、坐标调整；通过 setText 函数实现 QLabel 的字符串显示，可用于显示显示签到状态、当前课程以及个人信息等；通过 setPixmap 函数实现 QLabel 的图像显示，可用于显示人脸检测结果图以及个人证件照；通过 setStyleSheet 函数实现 QLabel 背景颜色的更改。

使用本地 time 库的 time.localtime 函数获取当前的系统时间，并使用 setText 显示到界面上方的 QLabel 上，使用 setFont 调整字体类型、粗细以及字号。

效果图如图 6 所示。



图 6 客户端数据可视化界面效果图

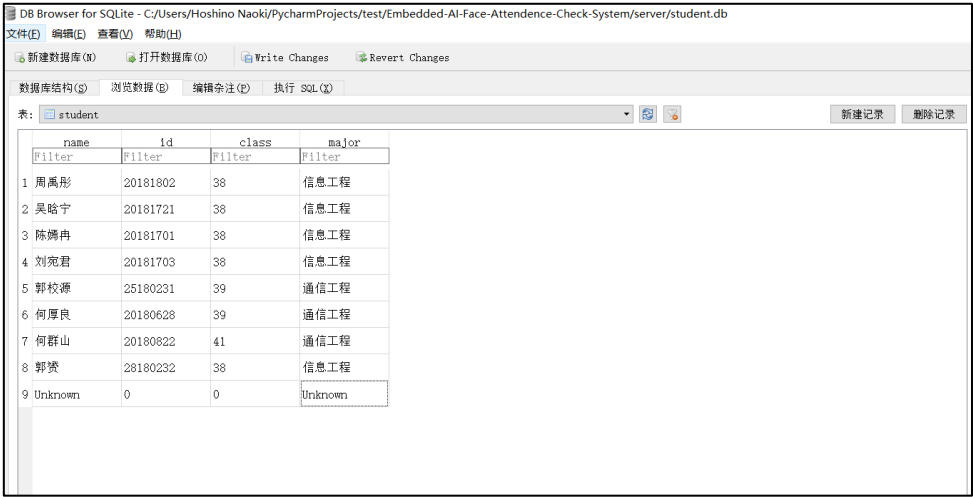
## 2.3 服务器功能

人脸考勤系统服务器的主要功能有：**数据库、签到管理、网络通信**等，主要用于查看签到名单、管理学生数据库、管理课程列表。

### 2.3.1 数据库模块

数据库模块基于 MySQL 实现，包括**个人信息数据库、人脸数据库**两部分。

首先建立**学生个人信息数据库**，主要用于存储学生个人信息，包括个人姓名、证件照、学号、班级、专业等。使用 DB Browser 软件进行数据库的创建、查看以及修改，如图 7 所示。



	name	id	class	major
1	周禹彤	20181802	38	信息工程
2	吴晗宁	20181721	38	信息工程
3	陈婉冉	20181701	38	信息工程
4	刘宛君	20181703	38	信息工程
5	郭校源	25180231	39	通信工程
6	何厚良	20180628	39	通信工程
7	何群山	20180822	41	通信工程
8	郭赞	28180232	38	信息工程
9	Unknown	0	0	Unknown

图 7 使用 DB Browser 进行数据库创建与修改

Python 定义了一套操作数据库的 API 接口：SQLite3，所以本系统使用 SQLite3 直接来操作数据库，使用 Python 操作数据库的方法很简单：先**创建光标函数 cursor**，相当于命令行 cmd 中的光标，然后将相应的 **mysql** 指令表示为字符串，再通过 **execute** 函数执行相应的操作，使用 **fetchall** 函数获得返回结果。关键代码如图 8 所示。

```
1 #导入数据库驱动
2 import sqlite3
3 #连接数据库
4 conn = sqlite3.connect("student.db")
5 cursor = conn.cursor()
6 sql = "select * from student"
7 cursor.execute(sql)
8 value = cursor.fetchall()
```



图 8 使用 python 建立并调用个人信息数据库的所有信息

人脸数据库包括个人证件照与人脸特征向量两部分。人脸特征向量由于数据量较大，不便于使用 MySQL 进行存取，故将其存于 **facedata.pkl** 文件中进行管理；由于使用 MySQL 存储图片需要额外进行图像编码，导致增加额外的编解码时间，故使用 **OpenCV** 的 **cv2.imread** 函数实现本地读取个人证件照图片。

### 2.3.2 签到管理模块

服务器需要具备签到管理的基本功能，需要建立新的抽象类别对学生、课程等对象，以及签到、迟到等抽象方法进行管理。本系统使用 Python 面向对象编程，实现了课堂签到管理系统，具体类与成员定义如下：

#### (1) **student** 学生类

成员变量：姓名、学号、班级、专业、签到状态

成员函数：初始化、个人信息显示

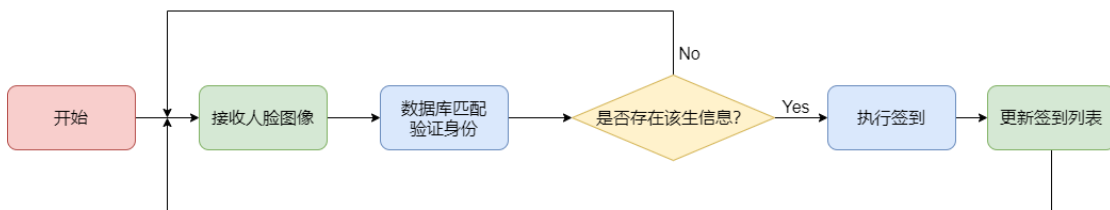
#### (2) **course** 课程类

成员变量：课程名称、全体学生列表、已签到列表、迟到列表

成员函数：签到、全体成员显示、已签到成员显示、迟到成员显示

接收客户端发送的图像并识别后，通过姓名字段调用数据库个人信息，用来声明 **student** 学生实例，并调用 **course** 执行签到功能，更新学生签到状态、已签到和迟到列表并通过服务器界面进行显示。

服务器可以获取当前时间，并监控不同教室客户端的当前课程以及签到情况；若该教室的客户端未连接服务器，显示“客户端未连接”字样，反之显示“已连接客户端！”字样；下方显示已签到名单与未签到名单，实时刷新；右方三个按钮分别为“签到复位”、“课程列表”、“退出系统”，实现相应的管理功能。签到管理系统的工作流程图、效果图，如图 9、图 10 所示。



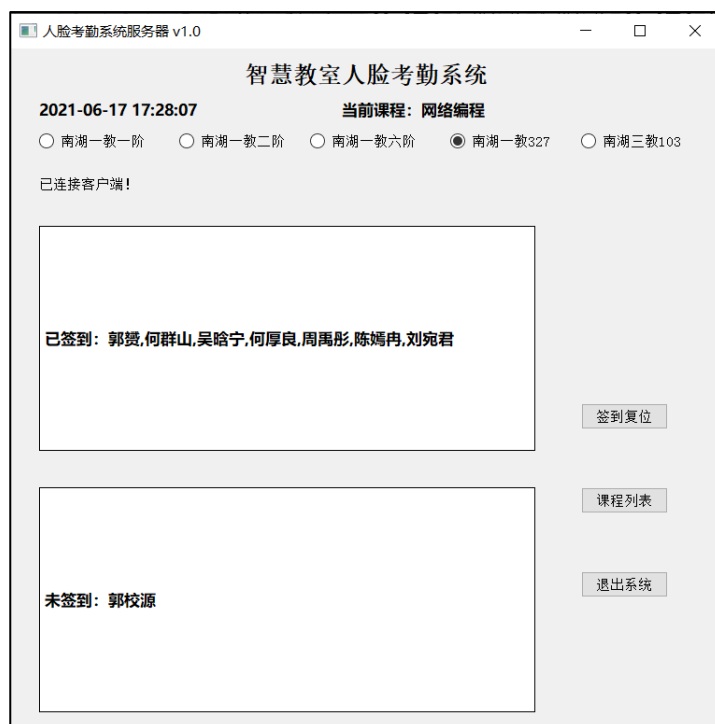


图 9 签到管理系统工作流程图

图 10 服务器签到管理界面

### 2.3.3 网络通信模块

服务器具备网络通信功能，基于 TCP/IP 协议的 socket 套接字实现，其 IP 地址由路由器通过 DHCP 自动分配。主要作用是接收人脸图像到服务器进行个人信息匹配，并发送相应的个人信息，如姓名、班级、学号、专业、签到状态等。

网络通信模块包括图像数据流解码、数据流发送与接收两个子模块。实现方式与 2.2.2 中客户端网络通信模块实现方式对称。

服务器网络通信模块流程图，如图 11 所示。

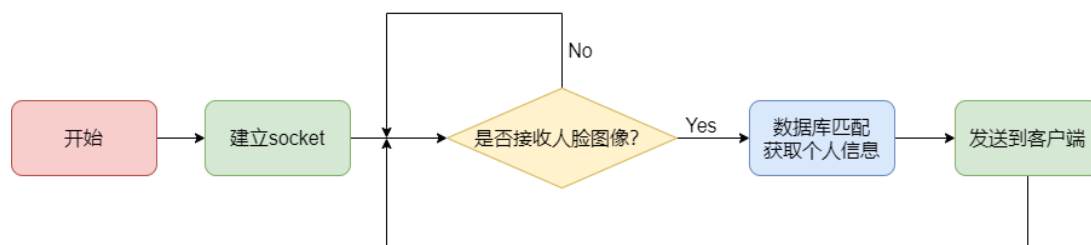


图 11 服务器网络通信模块流程图

## 三、详细结构

### 3.1 整体流程图

本系统采用**客户端-服务器工作模式**，由客户端先执行人脸识别，将人脸图像发送到服务器进行身份匹配，待服务器确定身份后，执行个人信息的发送，最终将人脸检测结果，个人信息同时显示在客户端设备上。以下是本系统的整体工作流程图，如图 12 所示。

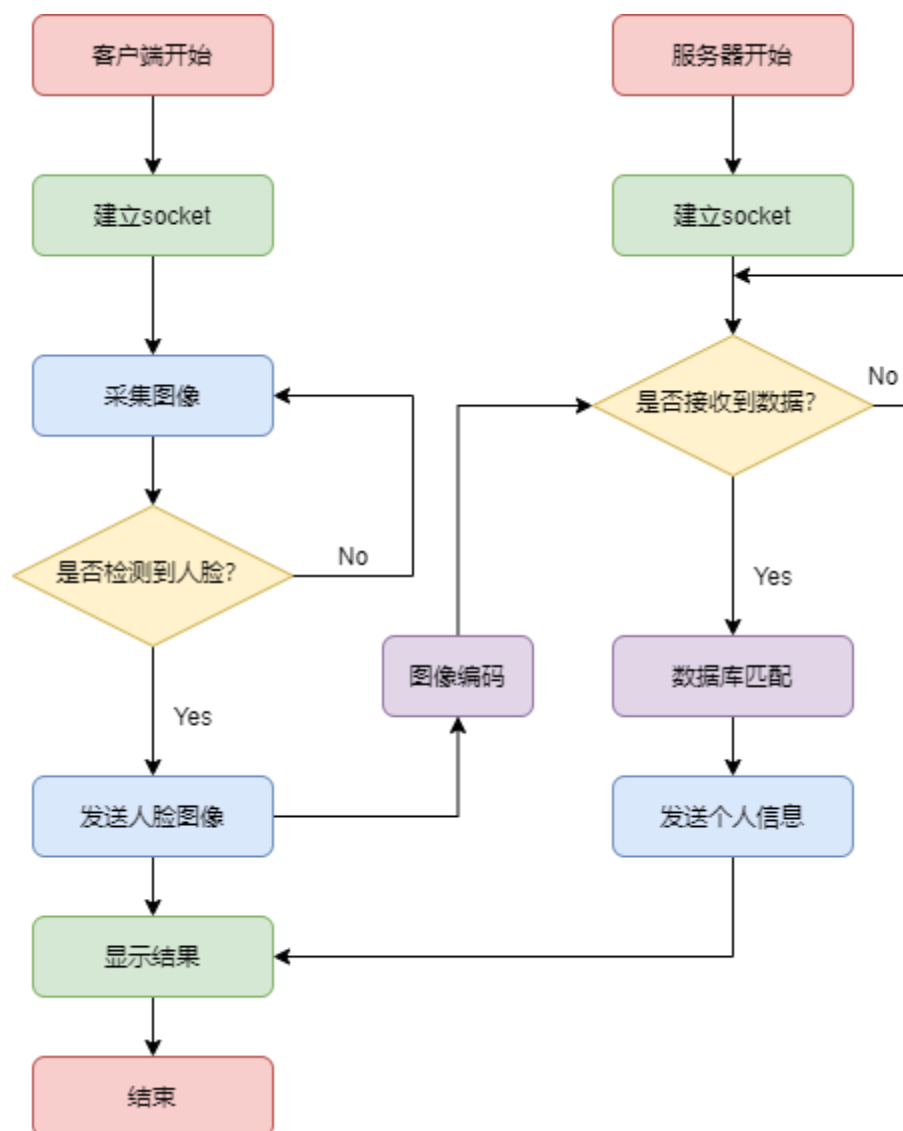


图 12 本系统的整体工作流程图

## 3.2 数据库说明

本系统使用 Python 的数据库接口 SQLite3 来操作数据库。创建了一个数据库文件 student.db，内含有一个名为 student 包含多个学生及所有个人信息的表格，包含 **name/id/class/major** 四个字段。

### 3.2.1 E-R图

本系统的数据库包含学生、人脸两个实体。学生与人脸是一对一的关系。学生实体包含属性有：**姓名（主键）**、学号、班级、专业；人脸实体包含属性有：**人脸特征向量（主键）**、个人证件照。学生实体与人脸实体之间是“拥有”关系。本系统的数据库 E-R 图如图 13 所示。

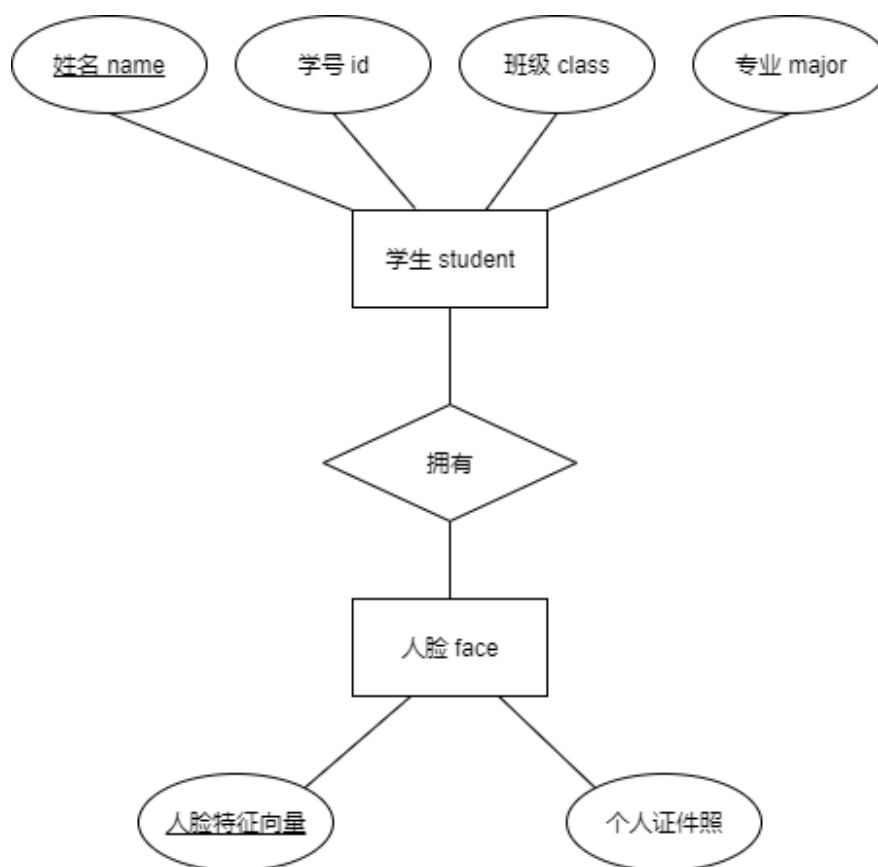


图 13 数据库 E-R 图

### 3.2.2 数据字典

学生数据库含有姓名、班级、学号、专业、签到状态等属性，其中“专业”属性由于取值较多，且取值需要更改，使用了数据字典，增加了新的“专业”表，当输出学生专业时，需要索引专业表，取出专业代号所对应的专业名称。结构如图 14 所示。

	name	id	class	major		major	code
	Filter	Filter	Filter	Filter		Filter	Filter
1	Unknown	0	0	Unknown	1	通信工程	001
2	周禹彤	20181802	38	002	2	信息工程	002
3	吴晗宁	20181721	38	002	3	自动化	003
4	陈嫣冉	20181701	38	002	4	测控技术	004
5	刘宛君	20181703	38	002			
6	何群山	20180822	41	001			
7	郭赞	28180232	38	002			

图 14 “专业”字段所对应的数据字典

### 3.2.3 数据流图

以客户端 A 为数据起点，采集人脸图像，经过人脸识别模块后，将人脸特征向量发送到人脸数据库进行匹配；匹配结果姓名将被传送到学生数据库用于查询该学生的个人信息，并通过数据可视化模块，将个人信息与证件照一同显示在界面上，返回到客户端 A 处。本系统的数据流图，如图 15 所示。

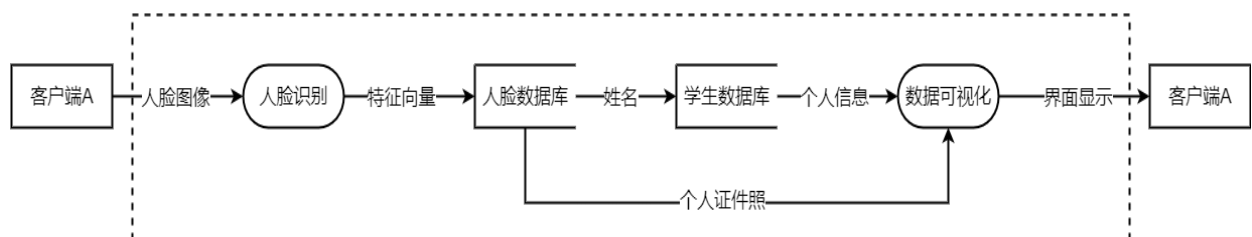


图 15 本系统的数据流图

### 3.3 数据加密

凯撒密码（Caesar 密码）是最早的代换密码，也是**移位加密算法**的雏形。其算法是：将每个 ANSI 码用字母表中它之后的第 k 个 ANSI 码（称作位移值）替代，本系统使用的移位加密算法的详细代码，如图 16 所示。

```
1 def encode():
2     list_s = []
3     r_move = int(input('请输入加密移位参数(右移): '))
4     s = input('请输入需要加密的字符: ')
5     for i in s:
6         list_s.append(ord(i))
7     for i in list_s:
8         #处理空格
9         if i == 32:
10            print(' ', end='')
11        #对大写字母进行处理
12        elif 65 <= i <= 90:
13            i += r_move
14            while i > 90:
15                i -= 26
16
17        #对小写字母进行处理
18        elif 97 <= i <= 122:
19            i += r_move
20            while i > 122:
21                i -= 26
22        print(chr(i), end='')

```

图 16 移位加密算法代码实现

## 四、技术难点

### 4.1 图像编码格式转换

按照原理图的设计，客户端需要实现人脸图像采集，并将图像发送到服务器进行特征匹配，接着服务器将识别结果再回复至客户端，完成一次工作周期。之前使用的方案是检测并提取出人脸，将图片保存至本地，再用 socket 读取本地图片，转成字节格式发送，服务器接收图片后再存到本地，用 OpenCV 读取

并通过 face recognition 的匹配功能进行身份验证，最后再通过 socket 回复客户端，完成通信工作。该方案导致文件读写频繁，程序速度变慢。

为了解决上述问题，我们将之前本地图片存取方案修改为客户端直接发送 Mat 矩阵的方式。通过 Mat 转 numpy，再转字符 string 的方式，将 Mat 矩阵直接通过 socket 发送至服务器，大大提升了传输速度。代码实现如图 17 所示。

```
1  result, imgencode = cv2.imencode('.jpg', src, encode_param)
2      # 建立矩阵
3      data = numpy.array(imgencode)
4      # 将numpy矩阵转换成字符形式，以便在网络中传输
5      stringData = data.tostring()
6
7      # 先发送要发送的数据的长度
8      # ljust() 方法返回一个原字符串左对齐,并使用空格填充至指定长度的新字符串
9      s.send(str.encode(str(len(stringData)).ljust(16)));
10     # 发送数据
11     s.send(stringData);
```

图 17 使用 socket 直接发送 Mat 矩阵的代码实现

## 4.2 嵌入式部署

客户端程序需要运行在一台搭载摄像头的微型运算设备上。如何选用一款算力强大、体积较小、成本合理的边缘 AI 计算设备，是产品设计阶段需要慎重考虑的问题。

**Jetson Nano 开发板**是 Nvidia 推出的 GPU 嵌入式运算平台，使用 Linux 操作系统，具有强大的边缘 AI 算力，符合人脸识别、无线网络模块和摄像头驱动要求，适合部署客户端的人脸识别与发送程序，配置好 PyCharm，Face recognition 以及 PyQt5 环境后，顺利运行。产品效果图和实物图如图 18 所示。



图 18 本项目使用的 Jetson Nano 开发板以及 USB 摄像头

# 五、系统集成与测试

## 5.1 系统集成

系统集成包括产品 3D 建模仿真、外壳 3D 打印、硬件封装等环节。

### 5.1.1 3D建模仿真

使用 **Fusion 360** 对人脸考勤系统进行硬件的 3D 建模仿真，本系统硬件包括三部分：**Jetson Nano** 嵌入式开发板、摄像头和显示器，3D 模型构建如图所示。仿照校园里自动售货机上搭载的刷脸支付系统，此处使用矩形圆角外壳对以上三部分进行封装外壳设计，此处忽略硬件之间的连线，仿真效果如图 19 所示。

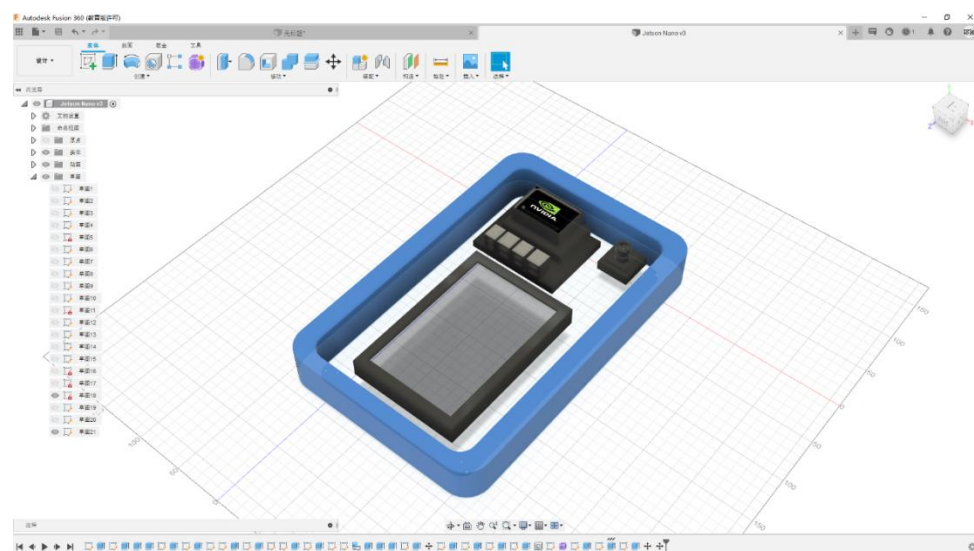


图 19 设备封装 3D 效果图

### 5.1.2 外壳3D打印

以上介绍了外壳的 3D 建模仿真，下面只需要将建好的模型导出工程文件，并使用 3D 打印机进行制造，就能够用来进行最后的设备封装了。这里我们导出 3D 模型的 STL 文件，外壳的尺寸范围为 270\*164\*30 (mm)。在此感谢南岭校区吉甲大师机械组的队员，他们为我们提供了打印尺寸为 350x250x300 (mm) 的大型 3D 打印机，机械组的队员将我导出的 STL 文件导入切片软件进行逐层分析，设计出打印方案。历时一天半后，外壳打印完成。



### 5.1.3 硬件封装

规划好硬件的位置后，这里就需要用电钻将外壳的相应位置上打孔，用螺钉将硬件和外壳进行固定，经历了几次轻微的错位之后，终于打出了几个跟硬件钉位对的上好孔。使用内六角 M3\*10 螺钉进行固定，硬件封装完成。完成设备封装的人脸考勤系统效果图，如图 20 所示。



图 20 完成设备封装的人脸考勤系统

## 5.2 产品测试

目前，集成好的设备已经于南湖一教 327 实验室完成了  $\alpha$  测试环节，如图 21、图 22 所示。已经实现了人脸识别、网络通信、数据库存取、签到管理以及界面交互等基本功能。工作状态下，客户端单帧平均运行时间为 1.3s，服务器平均运行时间为 0.7s。

测试视频链接如下：<https://www.bilibili.com/video/BV13M4y1u7oJ>。



图 21  $\alpha$  测试所用的客户端、服务器设备

图 22 人脸考勤系统运行实测图

但是在反复测试的过程中，依然存在一些 BUG 和问题有待解决：

1. **客户端接收的个人信息排序有时候会混淆**：例如姓名和学号同时显示在“姓名：”后，可能是由于 socket 接收字节长度过大，导致不同的字符串被同时接受；
2. **目前客户端设备只能连接路由器 WIFI**：只支持局域网场景下的使用，并且连不上手机热点，可能会影响设备使用的灵活性；
3. **人脸识别的视频帧数较低**：使用时比较卡顿，容易造成误识别，甚至停止识别等问题；
4. **Jetson Nano 缺少散热装置**：跟屏幕靠的太近，导致局部发热严重，时间久了程序会停止工作；
5. **缺少活体检测功能**：使用手机照片依然能够完成签到，属于伪签到。

## 六、总结与展望

项目历时两个月，完成了人脸考勤系统从框架设计、软件编程、系统集成到设备测试等一系列任务，涉及嵌入式、人脸识别、数据库、网络通信、管理系统、3D 建模、软件界面、硬件封装等多项技术要点，最终成功研发出一台能够实现基于人脸识别的智能化考勤设备。通过本次项目，我们团队充分发挥团结协作的精神，将复杂的任务通过合理分工化繁为简，并且在项目经历中加深了对网络编程理论知识的理解和转化，是一次非常有意义的课程设计经历。

未来，本团队将继续完善该系统的网络通信方式，不仅要实现效率更快的传输方案（如 UDP 等），还要实现该系统在广域网下的使用、增加活体检测功能以提高考勤的有效率，以及服务器对多设备多进程并行管理的程序实现；此外，还要改造硬件结构，减小发热对设备和程序的影响。

本项目代码已开源，开源链接如下：

<https://github.com/guoyun2020/Embedded-AI-Face-Attendance-Check-System>