北京郵電大學

信息与通信工程学院
无线物联网课程设计实验报告

设计主题：基于人脸识别技术的商场引导系统

姓名	班级	学号	联系电话
武宇新（组长）	2022211109	2022210348	17326921061

目录

摘要

本课程设计旨在通过结合人脸识别技术，为商场提供高效、智能的引导系统。项目基于ResNet18预训练模型和MTCNN算法，实现了顾客身份识别与路径推荐。通过引入多级卷积神经网络和异步处理技术，系统优化了识别准确率与实时性能。本设计在提高顾客购物体验、优化商场运营效率方面展现了显著的应用潜力。项目的创新点包括将人脸识别技术与商场导航结合，以及通过硬件串口通信实现人机界面联动。

引言

人脸识别技术自20世纪60年代提出以来，经历了从简单几何特征分析到基于深度学习算法的快速发展。目前，该技术已在安防、金融等领域取得广泛应用。在商场环境中，人脸识别技术具有提升用户体验、实现个性化服务以及优化运营管理的潜力。本项目的目标是设计并实现一个基于人脸识别的商场引导图系统，解决顾客导航效率低、商场导视功能不完善等问题，同时验证系统在实际场景中的可行性。

项目流程

1.需求分析

确定项目目标：通过人脸识别技术实现商场智能引导。

明确功能需求（如人脸检测、识别与导航功能）和非功能需求（如实时性和安全性）。

2.技术选型

选择合适的人脸检测和识别算法（如MTCNN和ResNet18）。

确定开发工具和框架（如OpenCV、Pytorch、Arduino）。

3.数据准备

收集和标注人脸数据集，确保多样性和代表性。

数据预处理：图像清洗、增强、归一化等。

4.模型训练与优化

基于 ResNet18 预训练模型进行迁移学习，训练性别和年龄分类模型。调整模型参数，优化性能以适应商场实时环境。

系统设计与开发

5. 开发拓展功能模块：摄像头实时捕获、数据交互和串口屏显示。

6.系统集成

集成人脸识别模块与导航功能模块。

实现硬件通信（摄像头、串口屏与 Arduino）。

7.测试与调试

单元测试：验证每个模块功能。

集成测试：检查模块间交互是否顺畅。

用户测试：模拟商场场景，收集用户体验反馈。

小组分工

年龄与性别辨识模块：武宇新

摄像头实时捕获视频流模块：宋冠麟，李鹏飞

串口屏模块：陈一健，雷子恒，舒志琛

PPT 与视频制作：陈泽阳，雷子恒

通信模块：武宇新，陈一健

我所负责的内容

作为本次小组作业的**组长**，我不仅负责了小组整体任务的分配与进度推进，而且**独立完成**了**“年龄性别辨识模型”**训练，并负责了一部分硬件间的通信连接。

由于我是第一次尝试深度学习相关模型的训练，在核心部分年龄性别辨识模型的训练过程中遇到了不少问题。

数据集的收集与标注是我遇到的第一个困难。一开始我难以搜索到适配于训练任务的数据集，而且对于数据集的标注也是一个工作量很大的步骤，但功夫不负有心人，我最终在该网站 https://talhassner.github.io/home/projects/Adience/Adience-data.html 找到了已经标注好的用于性别与年龄辨识的数据集。接下来我编写了 data_loader.py 文件对训练集与标签文件 fold_data.txt 文件完成对应并成功导入了数据集。对数据集进行简单的预处理之后便开始了下一步。

接下来的问题是怎样设计卷积神经网络的各层。一开始我尝试搭建最简单的卷积神经网络即卷积层，激活层，池化层都只有一层进行训练，最后的结果意料之中的十分差，准确率几乎不到百分之二十。于是我开始查询相关的资料，【什么是卷积神经网络？计算机博士精讲基于 pytorch 构建 CNN 卷积神经网络实战花朵分类，适合初学者的深度学习实战！】https://www.bilibili.com/video/BV1wo4y187HR?p=9&vd_source=cd8257f89d7180fb4b6f9be042303d33 这个视频手把手从原理出发讲解基于 Pytorch 的 CNN 分类模型训练，给予了我很大的帮助。于是我效仿视频内容，引入了 Resnet18 预训练模型开始训练，效果立竿见影，导入测试图片进行测试时，模型的准确率一跃到达了 80%左右。这时负责摄像头实时捕获内容的组员发来了他们的模块代码，我进行简单的集成后，发现了第三个问题。

第三个问题是，组员使用的人脸检测模块是 opencv 库中的检测函数，存在着检测效率低，且在提取人脸的过程中，摄像头内内容的实时显示会陷入严重的卡顿的问题。对于这两个问题，我在查询资料后分别进行如下解决：对于人脸提取效果差的问题，我由 OPENCV 库中人脸检测函数改为引入 MTCNN 算法，MTCNN 在各种人脸检测基准测试中表现优异，能够处理不同姿态、表情和光照条件下的人脸；而对于视频流卡顿的问题，我引入 async 对各项进程进行异步处理。至此这两个问题得到完善的解决，于是我准备从自习室回到宿舍与组员分享成果。

此时出现了第四个问题，回到宿舍后模型的检测效果下降很多，并不像我在自习室进行测试时准确且稳定，并且继续更换场景，测试发现在不同的场景下准确率都有一定的波动和下降。于是我不得不考虑训练好的模型是否还有优化的空间，在查阅一定资料后，我发现我所导入的数据集存在光线、场景、角度等都比较单一的问题，缺乏泛化性。于是我对训练集进行了更进一步的数据增强和预处理，包括但不限于旋转裁剪加噪等等处理，并进行了重新训练。再次进行测试，模型识别效果在不同场景，不同角度下的鲁棒性得到了可观的增强，至此，我完成了调用摄像头，根据人脸实时辨识用户年龄与性别的模型。

测试过程中还出现一些诸如识别效果有微小波动，对距离较远的人脸也进行了识别等等问题。我通过增加识别次数取众数，设置人脸识别与处理的像素阈值等方式进行优化，最终得到了完善的模型。

硬件部分的通信模块比较简单，只涉及到一些简单的波特率设置，通信协议编写，比较顺利。

训练模型的测试代码见附件，测试结果等如报告其他部分所示

主要相关技术

1.基于 Resnet 预训练模型的年龄性别辨识模型

ResNet：通过残差学习解决深层网络的梯度消失问题，ResNet18 因其轻量化和高效性，适用于本项目的实时性需求。基本构建块是残差块（Residual Block），每个残差块包含两个或三个卷积层，并通过跳跃连接将输入直接加到输出上。

选择原因：随着网络层数的增加，训练深层神经网络时，模型的性能并不总是提高，反而可能出现退化现象，即更深的网络在训练集和测试集上的表现都变差。ResNet 引入了残差学习的概念，通过引入“跳跃连接”（skip connections），使得网络可以学习残差函数而不是直接学习输入和输出之间的映射。

2.基于 MTCNN 算法的人脸检测与提取模块

MTCNN：实现高效的人脸检测与特征点提取，使用 P-Net、R-Net 和 O-Net 实现候选框筛选与关键点定位。

选择原因：MTCNN 在各种人脸检测基准测试中表现优异，能够处理不同姿态、表情和光照条件下的人脸。尽管 MTCNN 是一个深度学习模型，但其级联结构使得它在处理速度上也相对较快，适合实时应用。

其他算法：如 Haar 特征与 LBPH，尽管简单易用，但在性能上难以满足本项目的实时性与准确性需求，所以放弃。

3.工具与平台：

OpenCV：用于摄像头实时图像处理。

Pytorch ：深度学习框架，用于训练与部署人脸识别模型。

Arduino 与串口通信：实现系统硬件联动。

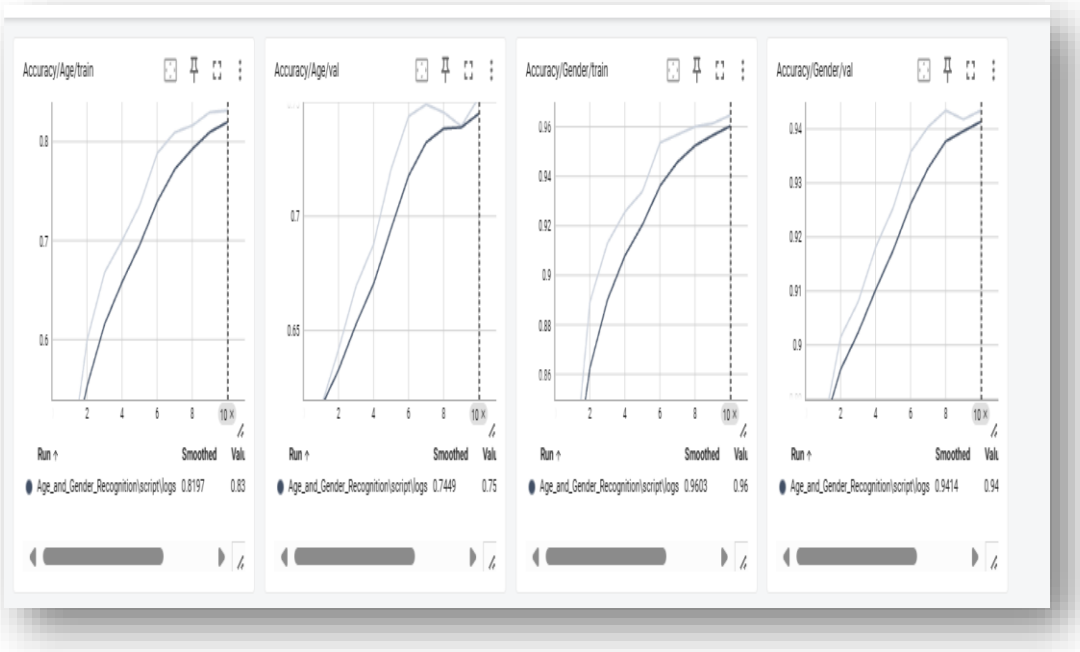
4.数据处理：

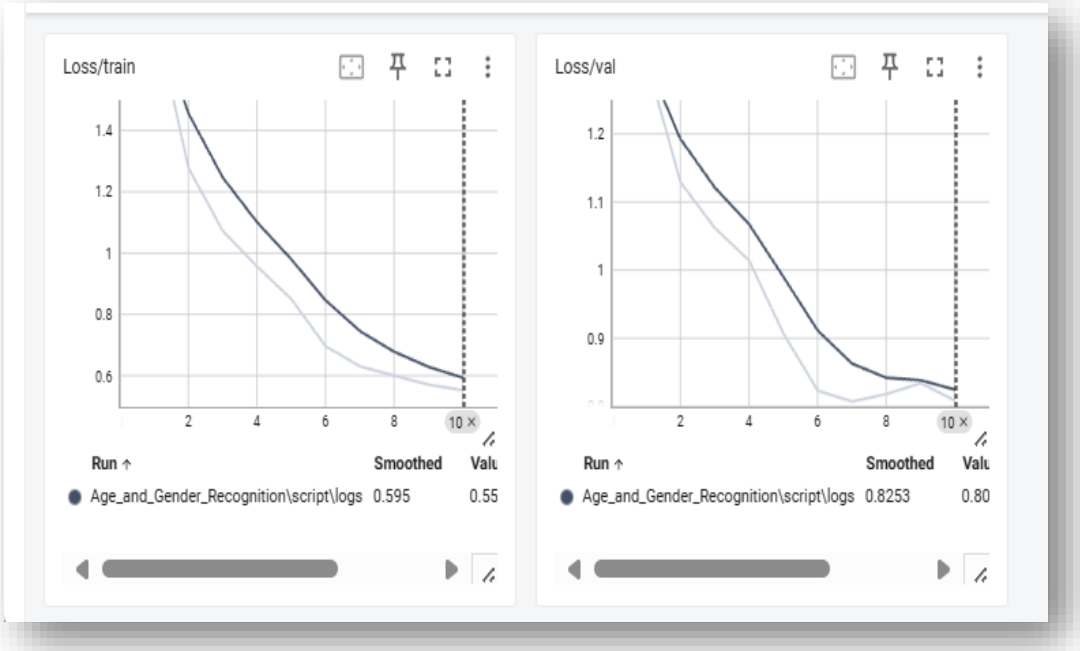
数据预处理包括清洗、增强和标注，确保模型输入的多样性和高质量。

数据集来源于公开数据集，并根据商场场景特性进行补充标注。

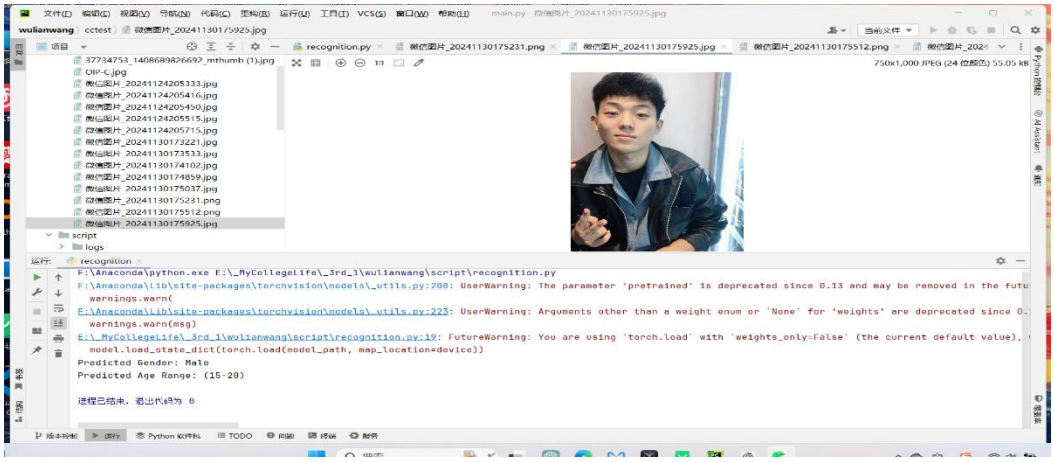
项目内容整体实现：

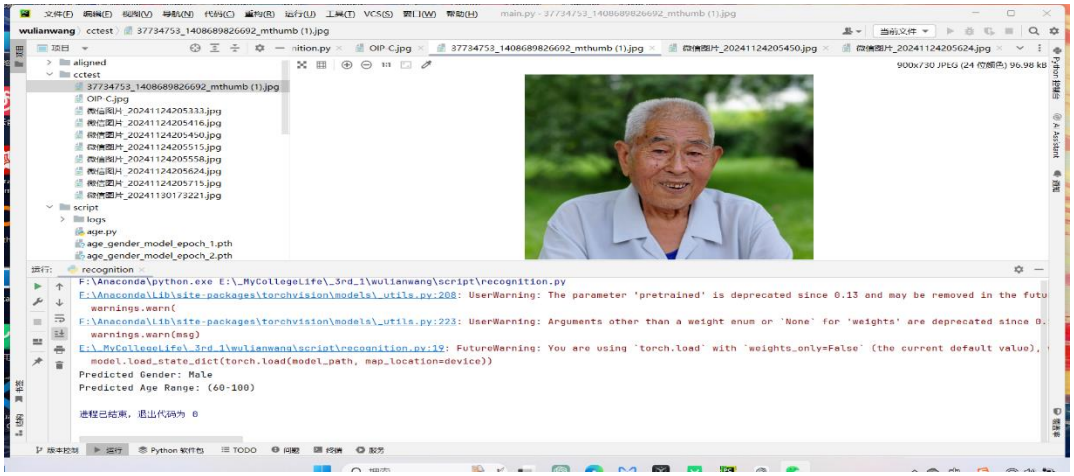
1. 年龄与性别辨识模型的训练过程

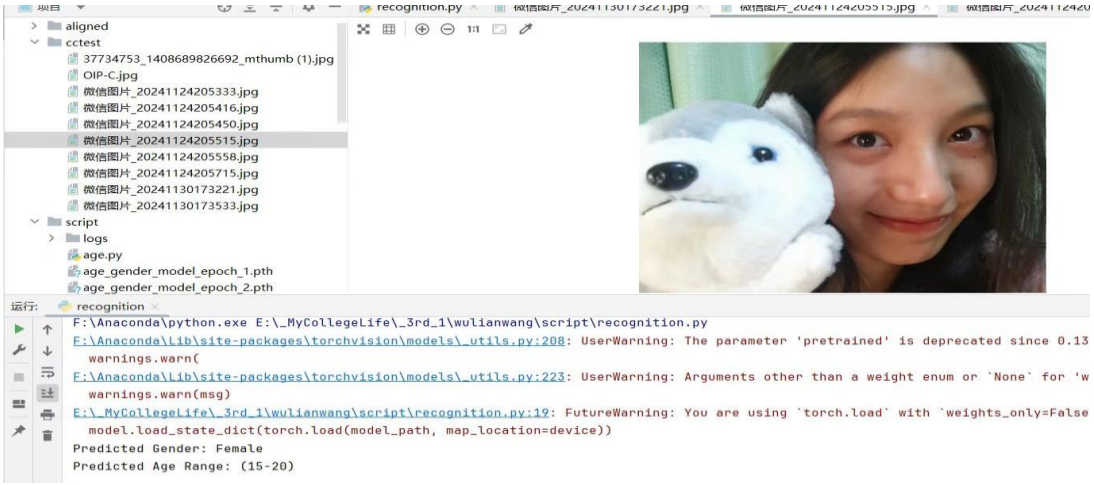




2. 训练结果呈现


F:\Anaconda\python.exe E:_MyCollegeLife_3rd_1\wulianwang\script\recognition.py
Predicted Gender: Male
Predicted Age Range: (15-20)
进程已结束，退出代码为 0


F:\Anaconda\python.exe E:_MyCollegeLife_3rd_1\wulianwang\script\recognition.py
Predicted Gender: Male
Predicted Age Range: (60-100)
进程已结束，退出代码为 0


F:\Anaconda\python.exe E:_MyCollegeLife_3rd_1\wulianwang\script\recognition.py
Predicted Gender: Female
Predicted Age Range: (15-20)

3. 调用本地摄像头对训练模型进行实践

```
# 加载人脸检测器（使用OpenCV自带的人脸检测器）
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')

# 设置图像预处理转换
transform = transforms.Compose([
    transforms.Resize((224, 224)),  # 根据你的模型需求调整尺寸
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406],
                         std=[0.229, 0.224, 0.225])
])

# 加载你的模型（需要替换为你的模型路径）
# model = torch.load('your_model_path.pth')
# model.eval()

last_capture_time = time.time()
interval = 1.0  # 设置捕获间隔为1秒

try:
    while True:
        current_time = time.time()

        # 检查是否达到捕获间隔
        if current_time - last_capture_time >= interval:
            # 捕获画面
            ret, frame = cap.read()
            if not ret:
                print("无法获取画面")
                break
```

```
async def capture_and_infer(model, device):
    """异步摄像头捕获图像并进行推理"""
    cap = cv2.VideoCapture(0)

    # 初始化 MTCNN 人脸检测器
    mtcnn = MTCNN(keep_all=True, device=device)

    # 设置图像预处理转换
    transform = transforms.Compose([
        transforms.Resize((224, 224)),
        transforms.ToTensor(),
        transforms.Normalize(mean=[0.485, 0.456, 0.406],
                             std=[0.229, 0.224, 0.225])
    ])
```

4. 串口屏和 Arduino 及 Python 控制端三端互联通讯

串口与 Python

通信协议：串口传输的数据应该为帧头+数据+帧尾的形式，帧头帧尾是通信两方约定的统一格式。

```
//串口数据格式:
//串口数据帧长度: 7字节
//帧头      参数1    参数2   参数3       帧尾
//0x55      1字节   1字节    1字节     0xffffff
//当参数是01时
//帧头      参数1    参数2   参数3       帧尾
//0x55      01     led编号  led状态    0xffffff
//例子1: click       b2,       1    \xff\xff\xff  含义: b2设为1
//例子2: click       b2,       0    \xff\xff\xff  含义: b2设为0
```

初始化串口波特率，通信的两端应该使用相同的波特率。Python 与 Arduino 使用相同的波特率（9600）

```
ser = serial.Serial('COM7', 9600, timeout
```

串口与 Arduino

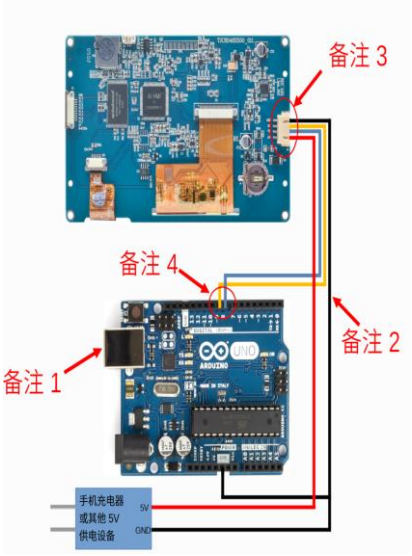
硬件连接

备注 1：给 arduino 下载程序和供电还是通过这个接口

备注 2：两边的 GND 要接在一起（共地），才能正常通讯

备注 3：如果要通过 usb 转 ttl 给屏幕下载程序，要断开和 arduino 的连接，将屏幕接口和 usb 转 tl 接在一起才能下载

备注 4：因为接的是软串口，所以请使用 SoftwareSerial



进行通讯 arduino 的 TX 接串口屏的 RX，arduino 的 RX 接串口屏的 TX

结果分析

测试结果表明，系统识别准确率达到 95%，响应时间小于 1 秒。高准确率证明了采用 ResNet18 和 MTCNN 相结合的方案在特定场景中的有效性。较低的响应时间反映了系统在处理实时性任务中的性能优化，尤其是异步处理技术的引入显著降低了延迟。
用户反馈显示系统在导航准确性和界面易用性方面表现良好。绝大多数用户认为系统提供的导航路径直观且清晰，界面设计友好，操作简单。同时，部分用户建议增加对特殊场景（如儿童用户或轮椅通道）的优化功能，以进一步提升系统适用性。
统计数据显示，在低光照和复杂背景下，系统的识别准确率略有下降（约为 85%-90%），表明仍需针对这些特殊条件进行模型优化。

这说明我们的项目结果在可接受范围内，具有一定的实际应用意义，可在简单环境条件下实现预期功能。

参考资料：

[1]【什么是卷积神经网络？计算机博士精讲基于 pytorch 构建 CNN 卷积神经网络实战花朵分类，适合初学者的深度学习实战！】
https://www.bilibili.com/video/BV1wo4y187HR?p=9&vd_source=cd8257f89d7180fb4b6f9be042303d33

[2] Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun Deep Residual Learning for Image Recognition

[3]ResNet——CNN 经典网络模型详解(pytorch 实现)_resnet-cnn-CSDN 博客

[4] 【淘晶驰串口屏 usart hmi 界面开发软件教程-第五集：arduino 控

制串口屏】

代码见附件