**《Python高级应用》**

**大作业实验报告**

项目名称 ： 多维人脸分析系统

姓 名 ： 　　 高心阳

学 号 ： 　 　 084623237

班 级 ： 　 计算机232

成 绩 ：

**人工智能与信息技术学院**

**南京中医药大学**

|  |
| --- |
| **项目要求：** |
| **项目描述**  开发一个基于计算机视觉的多维人脸分析系统，通过摄像头或图片检测人脸，并分析年龄、性别、情绪和种族信息。支持实时监测和图片分析，用户可根据需求自定义检测选项。  **主要功能：**   1. 选择本地图片进行人脸检测和分析。 2. 显示检测结果，包括性别、年龄、情绪和种族。 3. 支持用户配置检测选项。 4. 实时显示检测结果，并生成可下载的分析报告。 |
| **采用的技术** |
| **技术栈**  OpenCV：处理和显示图像及视频流。绘制检测框。  DeepFace：进行人脸分析，支持年龄、性别、情绪和种族检测。  Flask：搭建Web服务端。  HTML/CSS/JavaScript：构建用户界面，实现交互功能。  **系统架构**  系统分为前端和后端两部分：  前端：  使用HTML构建结构，CSS美化页面，JavaScript实现交互。  提供文件上传、选项选择和结果展示功能。  后端：  Flask负责接收请求、处理数据并返回分析结果。  DeepFace实现核心人脸检测与分析。 |
| **关键代码及相应运行结果** |
| |  | | --- | | 文件上传与检测选项处理 | | 接收前端上传的图片和用户勾选的检测选项。  保存图片到指定目录，解析检测选项。 | | # 获取上传的文件和检测选项  file = request.files.get('image')  if not file:  raise ValueError("未上传图片")  # 保存上传的文件  timestamp = datetime.now().strftime('%Y%m%d\_%H%M%S')  original\_filename = f"original\_{timestamp}.jpg"  file\_path = os.path.join(UPLOAD\_FOLDER, original\_filename)  file.save(file\_path)  # 获取检测选项  detection\_options = json.loads(request.form.get('detection\_options', '{}'))  actions = []  if detection\_options.get('age', False): actions.append('age')  if detection\_options.get('gender', False): actions.append('gender')  if detection\_options.get('emotion', False): actions.append('emotion')  if detection\_options.get('race', False): actions.append('race')  if not actions:  raise ValueError('请至少选择一个检测选项') |  |  | | --- | | DeepFace 分析人脸 | | 调用 DeepFace 框架分析人脸信息（年龄、性别、情绪、种族）。  支持多脸检测。 | | # 读取图像并转换颜色空间  frame = cv2.imread(file\_path)  rgb\_frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)  # 使用 DeepFace 分析  analysis = DeepFace.analyze(  img\_path=rgb\_frame,  actions=actions,  enforce\_detection=False,  detector\_backend='opencv'  )  if not isinstance(analysis, list): # 单人脸返回非列表形式  analysis = [analysis] |  |  | | --- | | 绘制分析结果 | | 在图片上绘制人脸框和检测结果。  根据性别判断框的颜色（绿色：性别明确，红色：性别不明确）。 | | results = []  for idx, face in enumerate(analysis):  region = face.get('region', {})  x, y, w, h = region.get('x', 0), region.get('y', 0), region.get('w', 0), region.get('h', 0)  # 性别判断决定框的颜色  gender\_dict = face.get('gender', {})  male\_prob = gender\_dict.get('Man', 0)  female\_prob = gender\_dict.get('Woman', 0)  color = (0, 255, 0) if abs(male\_prob - female\_prob) > 20 else (0, 0, 255)  # 绘制人脸框和序号  cv2.rectangle(frame, (x, y), (x + w, y + h), color, 2)  cv2.putText(frame, f"#{idx + 1}", (x, y - 10), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, color, 2)  # 整理结果数据  result = {  'id': idx + 1,  'gender': f"未知(男：{male\_prob:.2f},女：{female\_prob:.2f})" if abs(male\_prob - female\_prob) <= 20 else (  "男性" if male\_prob > female\_prob else "女性"),  'age': face.get('age', '未知'),  'emotion': EMOTION\_TRANSLATIONS.get(face.get('dominant\_emotion', '').lower(), '未知'),  'race': RACE\_TRANSLATIONS.get(face.get('dominant\_race', '').lower(), '未知')  }  results.append(result) |  |  | | --- | | 保存结果 | | 将绘制的结果图片保存到本地。  将分析结果保存为 CSV 文件。 | | # 保存结果图片  result\_filename = f"result\_{timestamp}.jpg"  result\_path = os.path.join(RESULT\_FOLDER, result\_filename)  cv2.imwrite(result\_path, frame)  # 保存分析结果到 CSV  df = pd.DataFrame(results)  csv\_filename = f"analysis\_{timestamp}.csv"  csv\_path = os.path.join(RESULT\_FOLDER, csv\_filename)  df.to\_csv(csv\_path, index=False, encoding='utf-8-sig') |   过程截图如下:    图1 初始化界面 图2 图片预览界面    图3 分析界面 图4 结果显示界面 |