SICHUAN UNIVERSITY

智能老人看护系统

项　目　计　划　书

任课教师 杨波

学　　院 计算机学院

专　　业 计算机科学与技术

组　　别 第二组

组　　长 曾灵杰

成　　员 陈治杰, 刘子鸣

2025 年 4 月 20 日

目录

1 系统概述

目前中国老龄化现象严重,60 周岁以上公民近 3 亿, 而 90% 以上的养老需求依托于居家养老, 因而家庭养老的智慧化改造是解决老人急难愁盼的关键因素. 数据表明, 随着老年人的身体协调性和反应速度下降, 发生跌倒等意外的风险增加, 独居老人特别是失能老人面临较高的生活风险, 但是现有的视觉识别方案存在性能开销大, 对特定场景的优化不佳缺陷. 因此, 本项目提出了一种基于行为识别的老人危险检测方案, 通过视觉技术和机器识别, 面向家庭和养老机构提供智慧解决方案, 并建立以 app、小程序为基础的可视化交互方案, 极大减小老人照料的人力开支.

2 功能需求

2.1 视频危险行为预测

需求描述

本系统需要对视频中的危险行为进行预测, 主要包括跌倒、摔倒等危险行为, 通过轻量化的时空建模模型, 在有限算力的情况下对视频进行实时处理和预测. 给出行为的危险程度, 当危险程度超过设定阈值时, 给出报警提示.

约束条件

本系统需要在移动端进行实时处理, 要求模型的参数量可以在家用计算机上计算, 推理速度大于 30FPS, 以保证实时性.

需求编号

PSP-25-2-1

2.2 实时视频流传输

需求描述

通过局域 IP 访问摄像头, 进行视频的采集和传输, 在局域网中进行视频流的传输, 保证视频流的实时性和稳定性.

在后端服务器上进行视频流的处理和分解, 一部分通过 python.flask 进行实时传输, 另一部分通过 ffmpeg 进行视频流的处理和存储, 进一步输入到模型当中进行处理和预测.

约束条件

为保证视频流的实时性和稳定性, 需要提供足够大的网络带宽和足够低的网络延迟.

需求编号

PSP-25-2-2

2.3 行为记录与分析

需求描述

本系统需要对视频中的危险行为进行记录和分析, 通过对视频流的处理, 记录下危险行为发生的时间、地点、持续时间等信息, 并进行统计和分析, 为后续的行为预测提供数据支持.

计划通过 python 中的数据库联动 MySQL 进行数据的存储和分析, 并通过 python 中的数据可视化工具进行数据的可视化展示.

约束条件

为了保证能够有效的存储数据, 需要使用专门的数据库进行数据的存储和分析, 同时也也可以实现数据的高效存储和查询.

需求编号

PSP-25-2-3

3 性能指标

鉴于本系统的主要工作环境是在家庭使用场景, 所以模型应该需要在算力较低的情况下运行 (TFLOPs $\leq$ 30), 同时需要保证模型的推理速度大于 30FPS. 同时考虑到传输的实时性, 视频流的传输延迟应该小于 100ms, 网络带宽应该大于 1Mbps.

4 接口需求

本系统需要提供 API 接口, 以便于其他系统进行数据的交互和传输. 主要包括视频流的传输接口、行为预测接口、行为记录与分析接口等. 视频流的传输接口

4.1 视频流的传输接口

使用 python 中的 flask 框架进行视频流的传输, 提供 HTTP 接口, 支持 GET 和 POST 请求, 返回视频流的实时数据.

```
from flask import Flask, Response
import cv2
```

```
app = Flask(__name__)

def generate_frames():
    # 使用IP相机作为视频源
    camera = cv2.VideoCapture('rtsp://admin:password@192.168.1.100:554/stream')

    while True:
        success, frame = camera.read()
        if not success:
            break
        else:
            ret, buffer = cv2.imencode('.jpg', frame)
            frame = buffer.tobytes()
            yield (b'--frame\r\n'
                   b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')

@app.route('/video_feed')
def video_feed():
    return Response(generate_frames(),
                    mimetype='multipart/x-mixed-replace; boundary=frame')

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
```

4.2 行为预测接口

使用 python 中的 flask 框架进行行为预测, 提供 HTTP 接口, 支持 GET 和 POST 请求, 以文本的形式返回预测结果, 包括危险行为的类型、危险程度等信息.

```
from flask import Flask, request, jsonify
import numpy as np
import cv2
import time
from model import BehaviorModel  # 假设已有的行为识别模型

app = Flask(__name__)
model = BehaviorModel()  # 加载模型

@app.route('/predict', methods=['POST'])
def predict_behavior():
```

```
    if 'video' not in request.files:
        return jsonify({'error': '没有收到视频文件'}), 400

    file = request.files['video']
    # 转换视频帧为numpy数组
    video_bytes = file.read()
    np_array = np.frombuffer(video_bytes, np.uint8)
    frame = cv2.imdecode(np_array, cv2.IMREAD_COLOR)

    # 使用模型预测
    behavior_type, danger_level = model.predict(frame)

    return jsonify({
        'behavior_type': behavior_type,
        'danger_level': danger_level,
        'timestamp': int(time.time())
    })

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5001)
```

4.3 行为记录与分析接口

联系后端数据库, 回答前端的查询请求, 提供 HTTP 接口, 以 json 文本的形式返回行为记录和分析的结果, 包括危险行为的发生时间、地点、持续时间等信息.

```
from flask import Flask, jsonify, request
import mysql.connector
from datetime import datetime, timedelta

app = Flask(__name__)

def get_db_connection():
    return mysql.connector.connect(
        host="localhost",
        user="username",
        password="password",
        database="elderly_care"
    )
```

```
@app.route('/records', methods=['GET'])
def get_behavior_records():
    conn = get_db_connection()
    cursor = conn.cursor(dictionary=True)

    # 获取查询参数
    start_date = request.args.get('start_date',
                 (datetime.now() - timedelta(days=7)).strftime('%Y-%m-%d'))
    end_date = request.args.get('end_date', datetime.now().strftime('%Y-%m-%d'))
    behavior_type = request.args.get('type')

    # 构建SQL查询
    query = "SELECT * FROM behavior_records WHERE record_date BETWEEN %s AND %s"
    params = [start_date, end_date]

    if behavior_type:
        query += " AND behavior_type = %s"
        params.append(behavior_type)

    cursor.execute(query, params)
    records = cursor.fetchall()

    cursor.close()
    conn.close()

    return jsonify({
        'total_records': len(records),
        'data': records
    })

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5002)
```

5 其它要求

5.1 安全性、可靠性、可维护性等要求

视频流以及行为预测数据的加密传输

5.2 运行环境要求

本系统需要在 Linux 操作系统下运行, 建议使用 Ubuntu 20.04 或以上版本, 并安装 python3.6 及以上版本, 同时需要安装相关的依赖库和工具. 同时需要安装 MySQL 数据库, 并配置相关的数据库连接信息.