开发文档

1. 引言

本项目旨在开发一个基于视觉、语音和多模态技术的情绪识别系统，专注于精准识别用户的情绪并进行有效反馈。通过集成先进的面部表情识别和语音情绪分析技术，系统能够全面分析用户的视觉和声音输入，并通过调用具有大规模参数的人工智能模型进行情绪处理，确保情绪识别的高精度与稳定性。该系统不仅仅局限于情绪的检测，还着重于情绪管理与互动。通过动画化的情绪小人，系统可以针对不同情绪状态进行个性化的展示，帮助用户在面对负面情绪时获得视觉上的安慰。

2. 系统架构

本项目由前端、后端、API层和数据库四部分构成，旨在通过多模态技术实现情绪识别和反馈功能。

- 前端：前端设备集成开发板、音箱、麦克风、摄像头和显示屏等硬件，用于捕捉用户的面部表情与语音输入，实现与用户的交互。摄像头实时捕捉用户的面部表情，并通过显示屏展示情绪小人动画，情绪小人根据用户当前的情绪变化做出不同的反应，帮助用户进行情绪调节。麦克风用于录入用户的声音输入，音箱则负责播放系统生成的语音反馈。
- 后端：后端负责处理所有的情绪识别和反馈逻辑，使用开源库如SpeechEmotionRecognition-Pytorch、DeepFace、GPT-SoVITS等。后端服务器会通过摄像头与麦克风采集的数据，首先对用户的面部表情和语音进行分析。DeepFace库会对视频流中的面部表情进行情绪识别，而Whisper库将音频转换为文本，再通过SpeechEmotionRecognition-Pytorch库对其进行情绪分析。情绪识别的结果将由后端系统处理，并传送到API层进行进一步反馈处理。
- API层：API层是系统与外部大模型交互的关键部分，调用星火大模型API进行情绪处理。通过将用户的情绪数据和文本输入传递给API，系统能够生成高度个性化的文字反馈。API返回的文字反馈信息会被传递到后端，后端使用GPT-SoVITS库生成具有情绪特征的语音输出，再由前端的音箱播放。
- 数据库：系统的数据库用于存储用户的情绪识别结果和情绪历史记录。通过记录用户的每次情绪状态变化，系统能够不断学习和优化用户情绪反馈的个性化服务。数据库存储的数据不仅能提高系统对用户情绪的预测能力，还为后续的数据分析和模型优化提供了可靠的依据。

3. 功能描述

- 面部情绪识别：通过集成DeepFace库，系统能够实时分析用户的面部表情并识别出其情绪状态。摄像头捕捉用户的面部特征后，DeepFace库对图像进行情感分析，能够识别出诸如愤怒、快乐、悲伤、惊讶、平静等情绪。这一功能使系统能够在用户未发出声音时，凭借表情变化获取他们的情绪状态。
- 语音情绪识别：系统通过麦克风捕捉用户的语音输入，使用Whisper库将语音转化为文本，随后通过SpeechEmotionRecognition-Pytorch库对文本进行情绪识别。SpeechEmotionRecognition-Pytorch库能够根据用户的语调、语速和音频特征，识别出他们的情绪状态，如愤怒、焦虑、喜悦等。语音情绪识别功能是对面部表情分析的补充，能够更准确地了解用户的当前情感状态。
- 情绪反馈：一旦识别出用户的情绪，系统将通过API层调用星火大模型生成个性化的文字反馈。系统会根据用户情绪的变化，自动生成一段适当的安慰或鼓励文本，以帮助用户调节情绪。生成的文本反馈会再通过GPT-SoVITS库生成对应情感的语音文件，音箱将播放这些语音反馈，使得情绪反馈更加直观和人性化。
- 情绪小人动画展示：为了帮助用户更好地理解反馈结果并调整情绪，系统将使用OpenCV库展示与用户情绪状态相匹配的动画情绪小人。这些动画会根据识别到的情绪状态发生变化，例如在用户愤怒时展

示愤怒的小人，在用户悲伤时展示悲伤的小人。这种可视化的情绪反馈，不仅能让用户直观感知到系统的反应，还能起到积极的情绪调节作用。

4. 技术栈

- 编程语言：Python
- 库与框架：DeepFace、Whisper、SpeechEmotionRecognition-Pytorch、OpenCV、GPT-SoVITS
- API：星火大模型API

5. 模块设计

- 情绪识别模块：面部表情和语音情绪识别，分别使用DeepFace和SpeechEmotionRecognition库。
- API集成模块：调用大模型API，将用户情绪与语音传递给API获取反馈。
- 反馈模块：处理API返回的数据，并通过GPT-SoVITS生成语音输出。
- 前端动画模块：使用OpenCV展示不同情绪的小人动画。

6. 数据库设计

用户数据表：

- user_id: 用户唯一标识
- emotion: 当前识别到的情绪
- timestamp: 识别时间

情绪记录表：

- emotion_id: 识别情绪的唯一标识
- face_emotion: 面部识别情绪结果
- voice_emotion: 语音识别情绪结果
- record_text: 用户语音输入内容
- response_text: 大模型返回的相应内容

7. 接口文档

讯飞星火，Spark4.0 Ultra。

9. 部署指南

- 依赖项安装：项目部署前，需使用Python的包管理工具 pip 安装所有依赖库。在项目目录下创建一个虚拟环境，并通过 pip install -r requirements.txt 安装项目依赖，包括DeepFace、Whisper、SpeechEmotionRecognition-Pytorch、GPT-SoVITS以及OpenCV等必要的开源库。确保开发板和其他硬件设备（如摄像头、麦克风、音箱）与系统驱动已成功集成。
- 模型与API配置：部署前，配置好星火大模型API的密钥，确保API调用能够正常工作。在系统的环境变量中配置API密钥和相关参数，保证大模型的文本生成和语音合成功能可以被顺利调用。
- 配置至华为云服务器：项目后端与数据库可以部署在华为云服务器上。首先，通过华为云服务器的命令行工具远程连接并配置服务器环境，安装必要的软件（如Python环境、Docker等）。然后将项目代码推送至服务器，并确保服务器与前端硬件设备能正常通信。

10. 测试计划

- 单元测试：针对系统中的每个独立功能模块，编写相应的单元测试。特别是情绪识别功能和API调用功能，通过测试确保DeepFace和SpeechEmotionRecognition-Pytorch库在各种情境下能够正确识别用户情绪。
- 集成测试：在集成测试阶段，确保所有模块无缝衔接，保证面部情绪识别、语音情绪识别与API调用之间的协作工作正常。测试内容包括从用户输入表情和声音，到系统生成反馈并展示动画的整个流程是否顺利运行。模拟多种情绪状态下的用户输入，检验API反馈和语音合成的完整流程，确保反馈与用户情绪高度一致。
- 用户测试：邀请不同年龄段、不同情绪状态的用户进行系统测试，评估情绪识别的准确性以及反馈效果。通过采集用户的实时反馈，了解系统在实际使用中的表现，尤其是在情绪识别和反馈的准确性和及时性上。