2022 年英特尔杯大学生电子设计竞赛嵌入式系统专题邀请赛

2022 Intel Cup Undergraduate Electronic Design Contest

- Embedded System Design Invitational Contest

作品设计报告

Final Report

Intel Cup Embedded System Design Contest

报告题目：基于多模态情感分析的学习状态评估系统

学生姓名: 郑晨德 霍嘉 彭宣尧

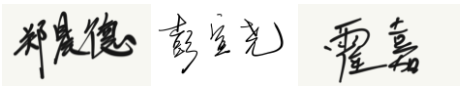
指导教师: 陈伟

参赛学校: 西安交通大学

2022 年英特尔杯大学生电子设计竞赛嵌入式系统专题邀请赛

参赛作品原创性声明

本人郑重声明：所呈交的参赛作品报告，是本人和队友独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果，不侵犯任何第三方的知识产权或其他权利。本人完全意识到本声明的法律结果由本人承担。

参赛队员签名： 

日期：2022 年 7 月 26 日

基于多模态情感分析的学习状态评估系统

Learning Status Assessment System Based on Emotion Analysis with Multimodal Information Input

摘要

在后疫情时代，线上模式主导的课堂教学给以云计算为中心的教学质量评估带来了巨大的计算压力。本作品基于Intel GNS-V40边缘计算主机，开发了一套适用于边缘端的基于多模态情感分析的学习状态评估系统，应用在家庭、课堂等场景中进行实时计算，并能将最终结果上传至学校服务器，减轻了云端教学评估的计算压力。同时，本作品部署了包括EmoNet在内的多类神经网络，用深度学习的方法进行学习状态评估，拥有优良的性能，具有广阔的应用前景。

关键词：深度学习；线上教育；边缘计算；多模态情感分析

Abstract

In the post-epidemic era, classroom teaching online has brought enormous computational pressure to cloud computation centers which are used to evaluate teaching quality. Based on the Intel GNS-V40 edge computing host, this work develops a learning status assessment system based on emotion analysis with multimodal information input for the edge. The school server relieves the computing pressure of cloud teaching evaluation to the edge. At the same time, this work deploys multiple types of neural networks including EmoNet, and uses the deep learning method to evaluate the learning state, which has excellent performance and broad application prospects.

Key words: Deep Learning; Online Education; Edge Computing; Multimodal Sentiment Analysis

目 录

第一部分 项目背景

1.1 立项意义

自 2020 年新冠疫情爆发以来，世界各地均受到了不同程度的影响。疫情早期，医疗资源匮乏问题是抗击疫情工作的核心，但经过长达两年全民抗疫后，世界已经步入后疫情时代。目前，在教育领域，国内普遍推行“停课不停学”的政策，且截止 2020 年，全国 98.4%的中小学（含教学点）已实现网络接入。据不完全统计，在 2020 年新冠疫情期间，全国各高校约有 103 万名教师线上授课，1775 余万名学生参与学习，涉及课程达 107 万门。

对于大多数学生而言，线上授课逐步成为新型教学模式的重要内容。这带来了新的困扰：教育经验不足的家长和缺乏直接接触的老师较难发现学生在学习过程中产生的情绪和学习习惯问题。因此，我们设计本项目，旨在通过人工智能和深度学习技术，基于情绪分析与人机交互，帮助发现学生在学习过程中的不足，并鼓励学生积极、健康地完成学习任务，并试图减少由于长期隔离可能产生的抑郁、孤僻等负面情绪。本项目应用的目标群体主要为疫情隔离的学生家庭群体，力图减轻父母在教育方面的负担，培养学生的学习习惯。

1.2 研究现状

对于人类情感状态的检测，就目前而言，研究重点在于通过各类传感器获取由人类情感引起的生理指标或者行为特征发出的信号（例如语音、面部表情、手势、姿态、脑电波、脉搏等），以建立可计算的情感模型。在具体的研究中，多模态（主要是音频和视频）情感识别往往备受青睐，但如何抽取有效的特征参数并运用恰当的模型来表达这些特征参数和情感之间的关联性，是亟待解决的一个关键问题。

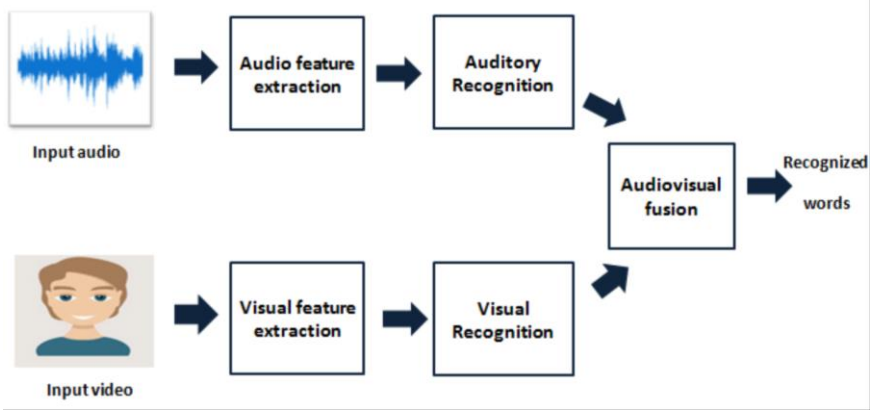


图1.1 多模态信息分析

关于情感语音的声学特征分析主要围绕韵律、频谱和音质特征。研究者已经发现很多声学特征与情感状态有关，如持续时间、语速、基音频率、共振峰、强

度、Mel频率倒谱系数（MFCC）等。研究人员将它们表示为固定维数的特征向量，其中的各个分量为各声学参数的统计值，包括平均值、方差、最大或最小值、变化范围等。近年来，神经网络提取优良特征参数的能力越来越受到关注。深度语音情感特征是基于语音信号或者频谱图，并通过语音情感识别相关任务学习到的深度特征。但是由于情感数据集的匮乏，目前应用比较广泛的是通过语音事件检测或者语音情感识别等任务，采用在大规模的训练数据学习到的深度语音特征作为语音情感特征，比如VGGish和wav2vec等。

在视频情感识别中，局部二值模式（Local Binary Pattern, LBP）、局部相位量化特征（Local Phase Quantization, LPQ）、Gabor 特征被广泛应用于静态图像的情感识别工作中；时序信息为情感识别提供了关键信息，许多基于上述特征的时空特征，如LBP-TOP（LBP from Three Orthogonal Planes）、 LPQ-TOP在基于视频的情感识别中广泛应用。计算机视觉中常用的方向梯度直方图（Histogram of Oriented Gradient, HOG）描述子、尺度不变特征变换（Scale-Invariant Feature Transform，SIFT）描述子、词袋模型（Bag of Words，BoW）和Gist描述子均在情感识别工作中有所涉及。另一类是基于深度神经网络的深度情感特征。深度情感特征避免了繁琐的图片预处理以及特征提取，相较于传统方法在情感识别相关任务上的表现更好，对光照、姿态、遮挡物等情感识别稳定性更高。深度情感特征主要从人脸情感识别数据集上训练的模型中进行抽取，比如目前应用广泛的深度特征是从人脸情感识别数据集（比如FER+）上训练的VGGNet、DenseNet等神经网络模型中抽取，并在主流的情感竞赛中取得了不错的结果。

图1.2 人类情感综合分析

多模态信息的分析方法有很多，从信息融合层次来看，多模态信息融合的方法主要有决策层融合和特征层融合，也有一些学者将这两个融合方式混合使用。决策层融合方式操作方便灵活，允许各个模态采用最适合的机器学习算法进行单独建模。特征层融合的通常做法是将各个通道的特征相串联，组合成一个长的特征向量，然后再将该特征向量放入机器学习算法进行分类或是回归输出。最新的认知神经科学表明，大脑在整合多感官信息时存在多阶段融合的现象，受此启发，研究者提出了多阶段多模态情感融合方法。首先训练一个单模态模型，然后将其隐含状态与另一个模态特征拼接再训练双模态模型，以此类推得

到多模态模型。这种建模方法在每个阶段只关注多模态信息的一个子集，然后综合考虑所有模态信息得到预测结果。多模态情感融合的关键在于实现了跨模态之间的有效整合以获得多模态信息的互补，从而比单模态情感识别具有更大的优势。

1.3 应用前景

1.3.1 家庭教育

随着科技进步和时代发展的现实需要，教育机器人受到越来越多学者的关注，成为国内外越来越多学者的关注，并涌现了一批教育机器人的研究机构。自 2017 年开始，我国政府高度重视机器人技术，希望与各国共同抓住工业化与信息化深度融合的契机，加快发展以机器人为代表的智能产业，加快机器人技术创新和产业发展，建设共创共享共赢的智能社会。教育机器人是机器人应用于教育领域的代表，是人工智能、语音识别和仿生技术在教育中应用的典型，以培养学生的分析能力、创造能力和实践能力为目标具有教学适用性、开放性、可扩展性和人机交互等特点的新型机器人。

就目前的市场发展现状来看，教育机器人主要被应用在大学和家庭领域中，如智能玩具、儿童娱乐教育同伴和家庭智能助理等，但是大部分的产品目前还处于概念性阶段，虽然已经明确了需求的应用场景，但是仍未投入批量生产，尚未得到市场的验证。教育机器人产品应用在各群体和情境中，仍具有多样性的发展潜力。

1.3.2 青少年心理健康

《中国国民心理健康发展报告》指出，中国青少年抑郁检出率达 24.6%，重度抑郁为 7.4%，心理健康问题日益凸显。同时，由于疫情隔离政策，更容易造成难发现、难预防等问题。近年来，随着情感计算的快速发展，人机情感交互已经成为人机交互和情感计算领域的研究热点，其内容涉及数学、心理学、计算机科学、人工智能和认知科学等众多学科。人机情感交互就是要实现计算机识别和表达情感的功能，最终使人与计算机能够进行自然、和谐的交互。当前，基于语音、面部表情、手势、生理信号等方式的人机情感交互系统和情感机器人的开发取得了一定的成果，它在诸多领域都有着广阔的应用前景。

情感识别是通过对情感信号的特征提取，得到能最大限度地表征人类情感的情感特征数据，据此进行建模，找出情感的外在表象数据与内在情感状态的映射关系，从而将人类当前的内在情感类型识别出来。在情感计算中，情感识别是最重要的研究内容之一。情感识别的研究主要包括情感识别、人脸表情识别和生理信号情感识别等

本项目的核心技术——综合音频以及图像的多模态情绪分析系统可以应用于实时检测青少年的心理问题，并及时将其心理状况报告给监护人，帮助其保持良好心理健康状态。

1.3.3 线上授课辅助

疫情期间，“网课”可以算得上最热门的话题之一。全国禁严、“停课不停学”的号召，催化了教学模式的革新，而这不仅涉及到学生的学习方式、教师的职业技能，还对网课平台以及整个在线教育行业的发展提出了新的考验。与线下

教育相比，线上教育这一借助了互联网平台和直播技术的新型授课模式，有着显著的优势。学生的学习场景不受限制，可以随时随地利用终端设备进行自主学习。

但是由于线上教育与线下教育有着比较大的差别，课堂效果远不如线下上课好。首先，因为不是面对面的上课，所以老师很难得到来自学生的课堂效果的及时反馈；另外由于沟通不如线下教学方便，内容无法当面向老师请教，导致对于部分知识不能及时掌握；最为严重的是因为学生比较分散，学习的环境各异，老师对于学生也不容易监管，这就导致部分控制能力比较差的学生在上课期间很难跟着老师的节奏，并且由于家长也不能一直监视着孩子，导致居家学习效率低下。

本团队设计的系统可以做到实时的监视学生的学习状态，在其分神时给予及时的提醒，并且可以收集学生在课堂上的表现生成一份学习状态报告交由老师和家长查看，方便及时了解学生的状态，并对其进行及时的纠正。

1.3.4 学校教学质量评估

本文设计的主要是单人学习状态评估系统，可以继续拓展设计多机位、多人模式下的学校教学质量评估系统打下基础。

在实际应用上，由于一个计算平台的性能可能难以同时评测一个教室中的所有学生，我们共提出两种方案，第一种方案是先将录像视频存储起来，然后在后台处理视频文件，这种方式成本较低，但是实时性差并且对于服务器的存储、计算要求较高，第二种方案是在一个教室中布置多个计算平台，每一个平台仅检测教室中一部分区域，这样可以将服务器的计算压力平分到多个边缘计算平台的上，每个平台在计算完成之后再将数据发送回主服务器，不仅实时性更好，而且很大程度上减少了主服务器的计算压力，另外也可以在一堂课结束之后很快得到反馈结果，帮助老师更好的了解学生的上课状态。现在教室中普遍安装的高清摄像系统，通过该分布嵌入式情绪分析系统的植入，容易综合分析得到课堂学生的整体学习状态，进一步得到的课堂教学情况。

其在分布较广的中小学课堂上具有广阔的应用前景，及时分析得到学生的学习状态、心理状态等。有利于及时进行学校教育质量的反馈和改进，同时早发现学生的不良习惯、学习状态，给予合适的干预，做到及时止损，有利于青少年健康成长。

图 1.3 线下教学课堂评估系统

第二部分 系统方案

2.1 研究内容

针对当下疫情期间网课频繁的情况，在线上教学中，老师大多数情况下无暇顾及学生的学习状态（同时，线下也会有类似的情况发生），导致学生课堂效率普遍不高、教学质量难以保证的情况时有发生。本文针对此类问题，研究和设计了一套基于情绪分析和环境检测、用于分析学习状态和辅助提高学习质量的学习状态评估系统。同时，该类系统也不仅仅可以用于学习状态的评估，还可广泛地应用于健康心理辅导、教学质量评估、家庭教育辅助等场合，我们做了的研究工作包括如下：

1. 了解学生疫情期间学习情况，针对不同学龄的学生的需求进行分析和总结，提炼出有针对性的功能。
2. 对目前已有的智能学习检测产品进行分析，对当前产品存在的痛点进行提炼，完善系统的功能定义。
3. 对基于多模态情绪分析的学习状态检测方法进行研究，分析当前已有的情绪检测方法，以及定义学习状态的量化指标，完善学习状态的检测方法。
4. 对基于神经网络的情绪识别、目标检测算法进行优化和部署，在实际的运用过程中，使得整个系统能够满足实时性要求。
5. 基于边缘计算主机（GNS-V40）构建学习状态评估的边缘应用案例，通过情绪信息、环境信息等多种信息融合，对学生的学习过程中的情绪状态进行分析和评估，得到学习状态反馈结果，帮助改善学习情况。
6. 实现智能的语音交互，能够与学生进行语音交流，具备实时提醒、状态反馈的功能。
7. 增加实用的辅助功能，包括网课视频播放、语音通话、学生用户存档等，提高学习辅助系统的完善度和实用性。

2.2 系统框架

2.2.1 软件流程

本项目主要通过麦克风和摄像头来感知学生当前的状态，通过对不同类型的输入信号分别进行预处理（裁剪、放大等），然后通过深度神经网络对学生当前情感状态进行预测，通过不断的离散采样来得到不同时间段学生的情感状态，生成该时间段的情感变化曲线，最后对所得到的总体数据进行滤波去噪处理，并由此来分析学生这段时间的学习状态。同时，我们也会实时记录当下学生的学习状态等信息，给予学生学习必要的提醒和帮助。

整个项目的软件构建在 QT 的框架之下，通过 QtDesigner 工具设计了软件的 UI 界面，并通过 QtThread 类的调用，实现了多线程的运行。为了使得整个项目的工程代码更加易于阅读，我们将代码的前后端进行了分离，前端是 UI 界面的

展示、事件的监听等交互功能的实现，后端主要是各个神经网络的实时信号处理、系统逻辑的实现。最后封装好各个界面、线程，将代码前后端进行对接，实现了整个基于多模态情感分析的学习状态评估系统的软件部分。

本次项目中，系统软件流程图所下所示：

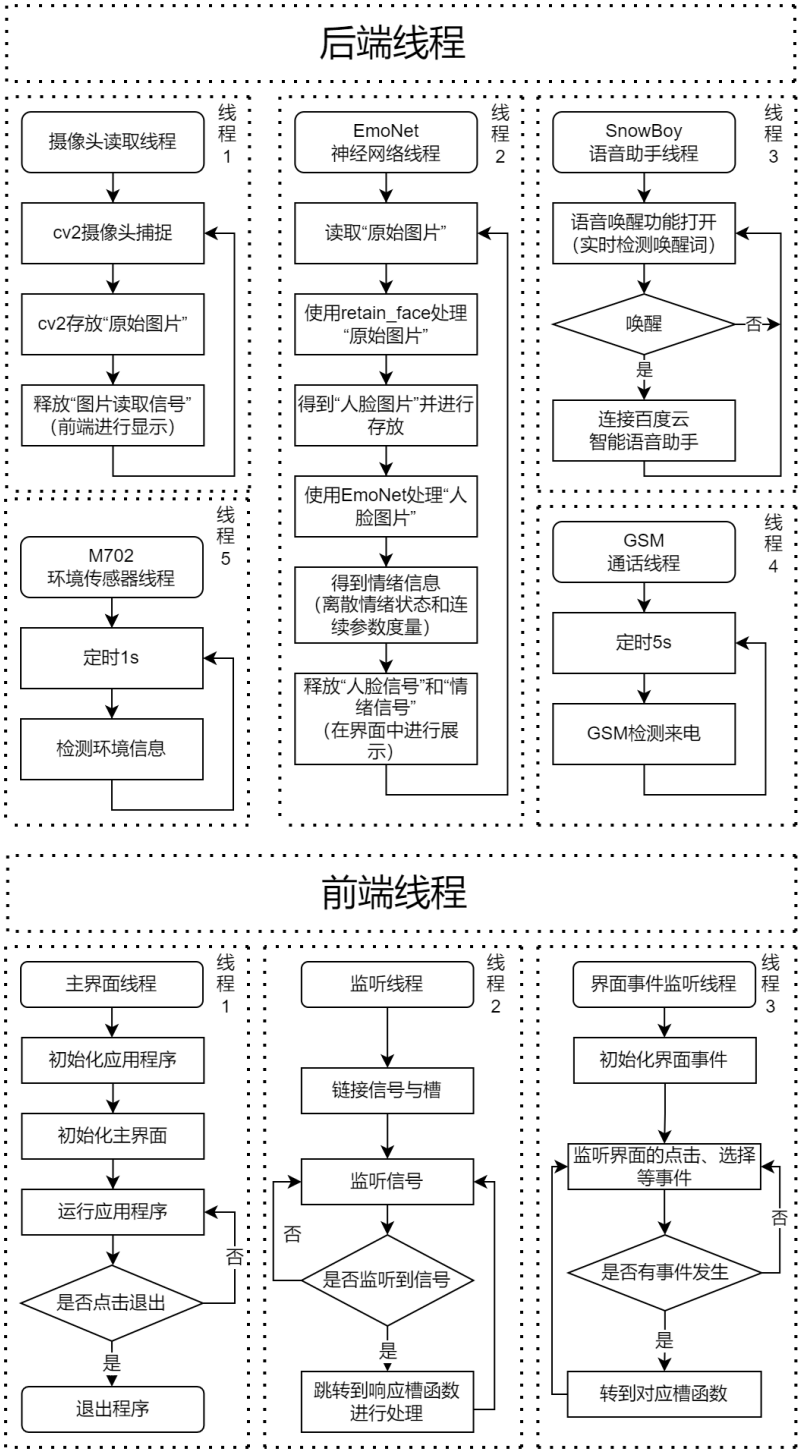


图 2.1 软件流程图

2.2.2 硬件架构

本次项目的硬件系统将会以 GNS-V40 边缘计算主机为核心，以多个信息采集模块（摄像头、麦克风语音等）组成的多模态信号（视频、音频、环境参数等信号）作为系统输入，通过边缘计算主机进行智能处理（包括进行基于神经网络的情绪识别和物品分类、处理多源输入信号、实时语音唤醒）。在系统输出方面，系统包括一个显示器和扬声器，在显示器上进行实时处理画面的显示，在扬声器上进行语音交互等功能。为了模拟真实的学习环境，我们将整个硬件系统部署在一张常见的课桌上，进行了整个系统的部署和测试。

本次项目采用的硬件架构如下图所示：

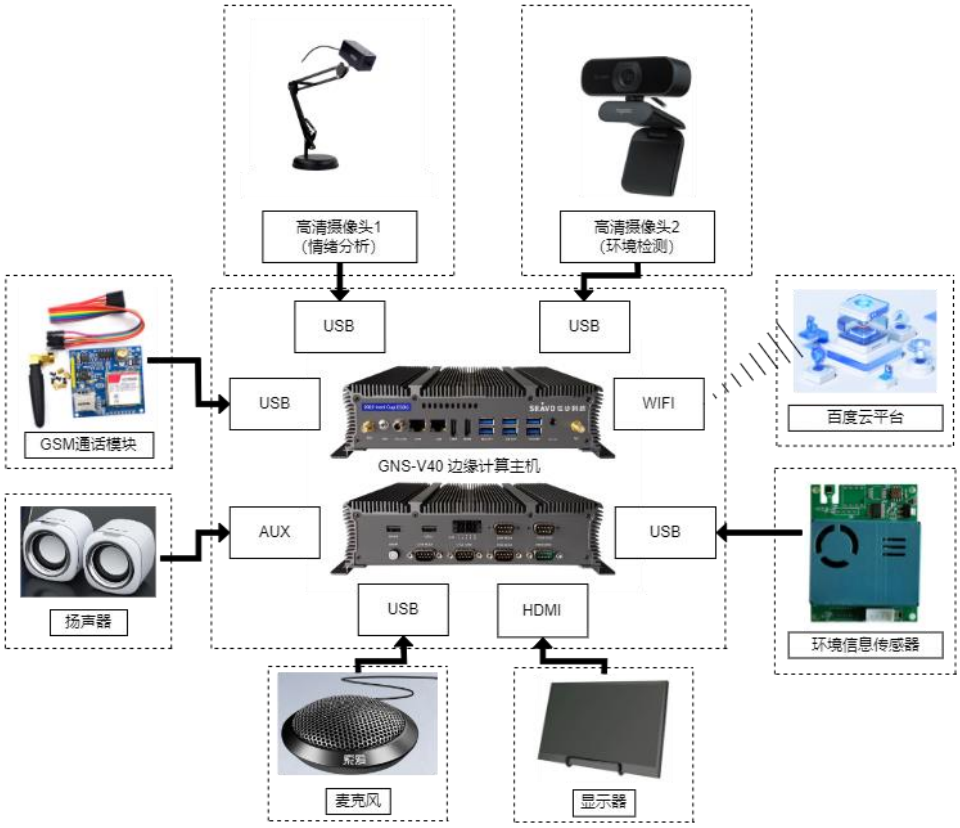


图 2.2 硬件架构示意图

2.3 关键技术

2.3.1 情感分析算法

此部分是项目的核心。目前基于计算机视觉的情感分析算法大多数都是将情感分析分为多个步骤来分开完成，主要包括有识别并裁剪人脸，找出面部基准点并投影到标准坐标系上，检测面部情绪特征三个步骤。而该算法的优点是可以使用单一的网络来同时完成这些任务并且取得优于上述算法的准确率，该模型主要对曾经最优的算法做出如下改进：

- 分类和连续情绪的联合预测，使得网络对数据集中的异常值更鲁棒

- 使用注意力机制，将焦点驱动到与情感估计相关的面部区域
- 通过知识蒸馏阿里平滑网络学习的标签
- 使用特别定制的损失函数来优化识别相关的指标

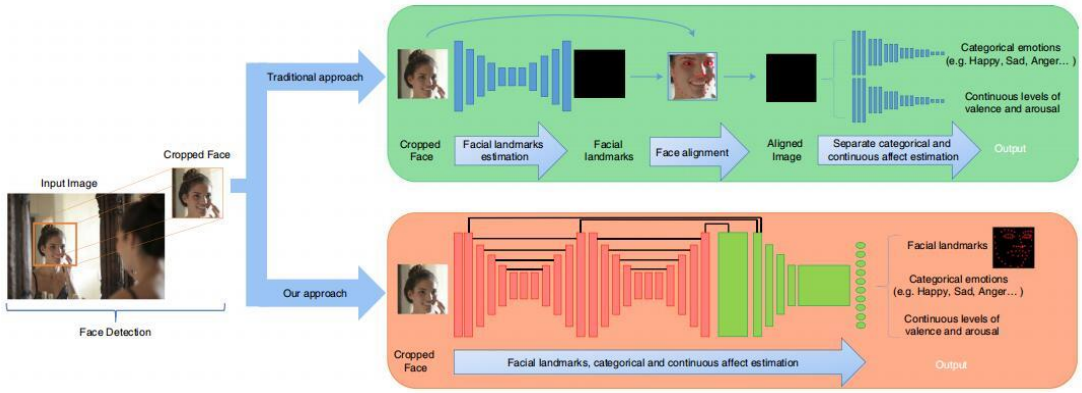


图 2.3 传统情绪识别算法与 EmoNet 比较

同时，该算法在三个著名的自然条件下获取的数据集（AffectNet，AFEW-VA，SEWA）上均进行了验证，虽然三个数据集内容不尽相同，但是实验结果展示出该算法均优于以前的所有方法。

该算法的最大特点是其可在一次推理中得到所需的各种信息，包括有面部标志、离散情绪和连续情绪。也正是因为由于整个推理过程只需要一次推理使得将该算法应用于实时情感分析成为可能。

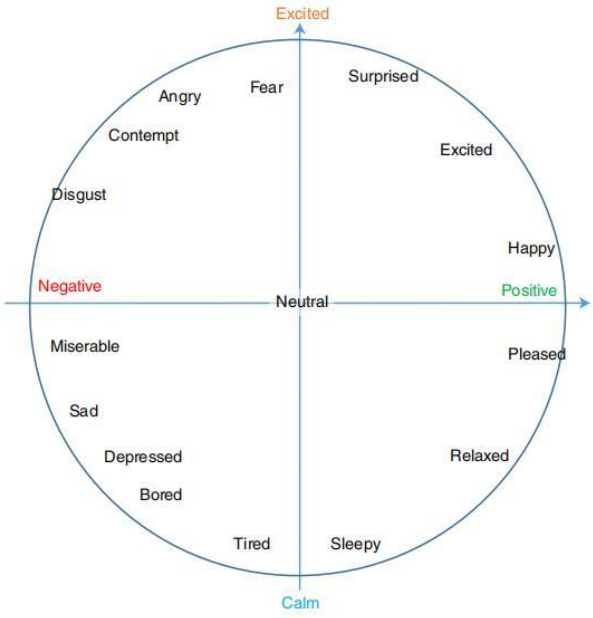


图 2.4 EmoNet 算法情感输出环

该算法具体过程是先通过一个面部检测算法在图片中定位出人脸位置，然后对其进行裁剪，将裁剪后的图片作为网络的输入，在人脸上找出面部的基准点，将其投影到一个标准坐标系上，来消除旋转平移缩放的影响，然后使用这些基准点来检测面部的情绪特征，最后会输出情感的激烈程度和情感的积极程度两个连续指标以及当前情感状态。

2.3.2 知识蒸馏

知识蒸馏（Knowledge Distillation），简称 KD，顾名思义，就是将已经训练好的模型包含的知识提取到另一个模型里面去。知识蒸馏是一种模型压缩方法，是一种基于“教师-学生网络思想”的训练方法，由于其简单，有效，因此在工业界被广泛应用。

其被提出是因为一般在训练模型的时候会使用复杂的模型，使用大量的计算资源，以便从非常大、高度冗余的数据集中提取出信息，在实验中，效果最好的模型往往规模很大，甚至由多个模型继承得到，但是大模型一般由于其推断速度慢，对部署资源要求高等原因不能直接部署到服务中使用。因此，模型压缩成了一个重要的问题，这里的知识蒸馏正是属于模型压缩的一种。

知识蒸馏使用的是 Teacher-Student 模型，其中 teacher 是知识的输出者，student 是知识的接受者，知识蒸馏共包含两个过程：

原始模型训练：训练 Teacher 模型，它的特点是模型相对复杂，可以由多个分别训练模型组合而成，在训练时，对于任何输入的 X，都能输出经过 softmax 映射的 Y，输出值 Y 对应相应类别的概率。

精简模型训练：训练 Student 模型，它是参数量较小、模型相对简单的单模型。在训练时，对于任何 X，其都能输出相应的 Y，Y 是经过 softmax 映射后同样能输出对应相应类别的概率值

其是将 Teacher 模型得到的输出 Y 直接作为 Student 模型的预期输出，除了正例之外，负标签也包含有大量的信息，而在传统的训练过程中，所有的负标签都被统一对待，也就是说，KD 的训练方式使得每个样本给 Student 模型的信息量大于传统的训练方式

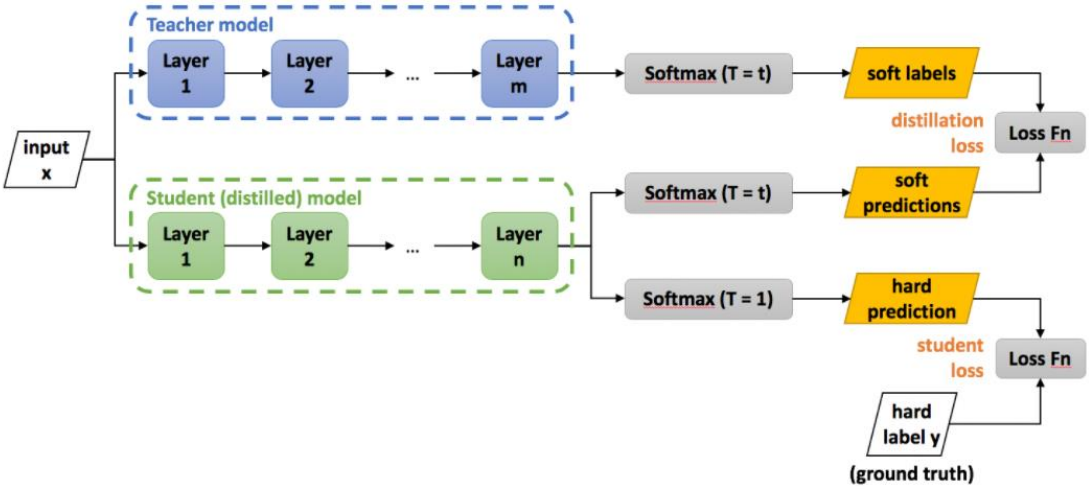


图 2.5 知识蒸馏模型示意图

2.3.3 YOLO

基于深度学习方法的一个特点就是实现端到端的检测。相对于其它目标检测与识别方法（比如 Fast R-CNN）将目标识别任务分类目标区域预测和类别预测等多个流程，YOLO 将目标区域预测和目标类别预测整合于单个神经网络模型中，实现在准确率较高的情况下快速目标检测与识别，更加适合现场应用环境。

YOLO 采用单个卷积神经网络来预测多个 bounding boxes 和类别概率，本方法相对于传统方法有如下有优点：

一、非常快。YOLO 预测流程简单，速度很快。基础版一般可以达到 45 帧/s；快速版可以达到 150 帧/s。因此，YOLO 可以实现实时检测。

二、YOLO 采用全图信息来进行预测。与滑动窗口方法和 region proposal-based 方法不同，YOLO 在训练和预测过程中可以利用全图信息。Fast R-CNN 检测方法会错误的将背景中的斑块检测为目标，原因在于 Fast R-CNN 在检测中无法看到全局图像。相对于 Fast R-CNN，YOLO 背景预测错误率低一半。

三、YOLO 可以学习到目标的概括信息（generalizable representation），具有一定普适性。我们采用自然图片训练 YOLO，然后采用艺术图像来预测。YOLO 比其它目标检测方法（DPM 和 R-CNN）准确率高很多。

我们的项目中使用到的是 YOLOv5 算法，其在 YOLOv4 的基础上添加了一些新的改进，速度与精度都得到了极大的性能提升，其主要的改进思路如下：

- 输入端：在模型训练阶段，提出了一些改进思路，主要包括 Mosaic 数据增强、自适应锚框计算、自适应图片缩放；
- 基准网络：融合其它检测算法中的一些新思路，主要包括：Focus 结构与 CSP 结构；
- Neck 网络：目标检测网络在 BackBone 与最后的 Head 输出层之间往往会插入一些层，Yolov5 中添加了 FPN+PAN 结构；
- Head 输出层：输出层的锚框机制与 YOLOv4 相同，主要改进的是训练时的损失函数 GIOU_Loss，以及预测框筛选的 DIOU_nms。

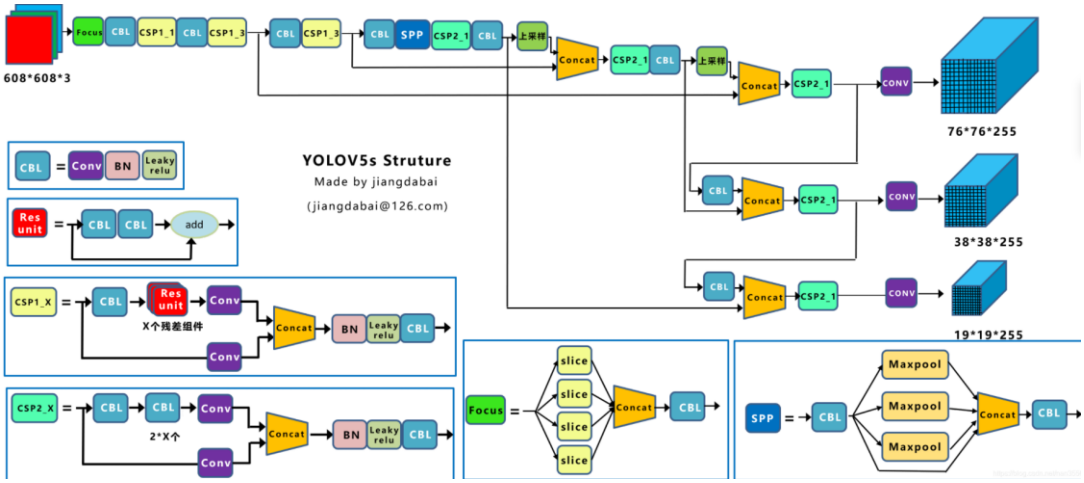


图 2.6 YOLO 结构示意图

2.3.3 卡尔曼滤波技术

卡尔曼滤波（Kalman filter）是一种高效率的递归滤波器（自回归滤波器），它能够从一系列的不完全及包含噪声的测量中，估计动态系统的状态。卡尔曼滤波会根据各测量量在不同时间下的值，考虑各时间下的联合分布，再产生对未知变数的估计，因此会比只以单一测量量为基础的估计方式要准。

卡尔曼滤波在技术领域有许多的应用。常见的有飞机及太空船的导引、导航及控制。卡尔曼滤波也广为使用在时间序列的分析中，例如信号处理及计量经济学中（在本项目中，也是用于分析情绪观测值的时间序列）。卡尔曼滤波也是机器人运动规划及控制的重要主题之一，有时也包括在轨迹最佳化。卡尔曼滤波也用在中轴神经系统运动控制的建模中。因为从给与运动命令到收到感觉神经的回

授之间有时间差，使用卡尔曼滤波有助于建立符合实际的系统，估计运动系统的目前状态，并且更新命令。

卡尔曼滤波的算法是二步骤的程序。在估计步骤中，卡尔曼滤波会产生有关目前状态的估计，其中也包括不确定性。只要观察到下一个量测（其中一定含有某种程度的误差，包括随机噪声）。会通过加权平均来更新估计值，而确定性越高的量测加权比重也越高。算法是迭代的，可以在实时控制系统中执行，只需要目前的输入量测、以往的计算值以及其不确定性矩阵，不需要其他以往的资讯。使用卡尔曼滤波不用假设误差是正态分布，不过若所有的误差都是正态分布，卡尔曼滤波可以得到正确的条件几率估计。

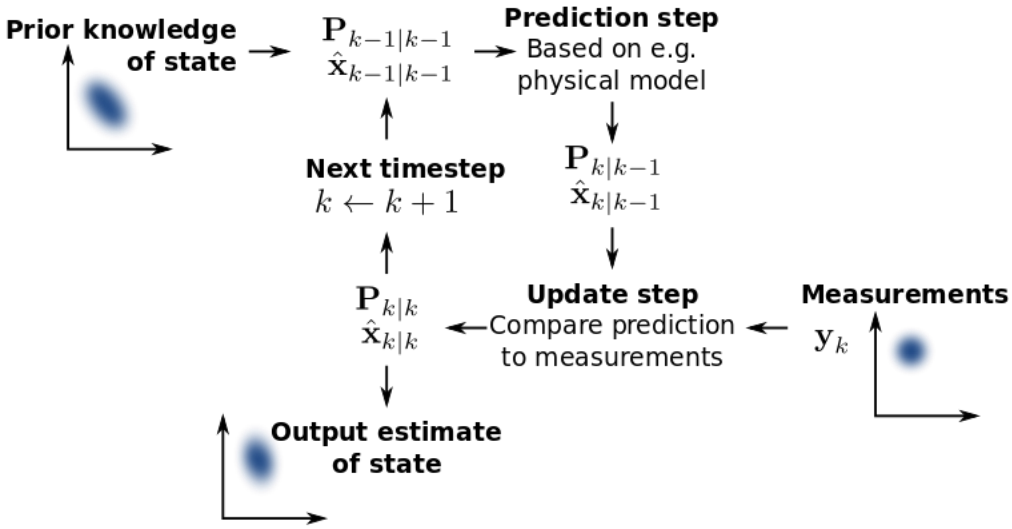


图 2.6 卡尔曼滤波器

在本项目中，我们的观测信号主要是人的面部表情和周围的环境信息，如果直接将观测到数据进行显示的话，会发现其中含有大量的噪声，并不符合人类情绪变化的一般规律（一般来说，情绪在很长的时间段中，应该是缓慢的变化）。因此，我们引入了卡尔曼滤波器对观测到的表情推测出的情绪信息做了推断，将信号的观测量作为含有噪声的系统输入，去估计当前的情绪变化状态，最后，能够将情绪观测值转换为最优估计的情绪状态值。

引入卡尔曼滤波进行数据处理，能够显著提高波形的平滑度，更加符合人类对于情绪变化的认知，也能够更加客观地反映出一段时间内人的情绪变化，提高学习状态评估的可靠性。综合来看，卡尔曼滤波器是一种有效的数据处理方式，能够改善噪声带来的影响，提高情绪分析的可靠性和稳定性。

2.3.2 语音处理技术

目前主流的语音特征提取技术有 MFCC 或 Filter bank，二者的计算步骤基本一致，但是 MFCC 额外进行了 IDFT。就技术内核而言，梅尔(Mel)频率谱是在已知信号频谱的基础上，基于人类听觉系统的感知特性，设计出的一种频谱分组方式。通过计算 Mel 频谱，将得到比原始傅里叶频谱更加具有区分性的频域紧凑表达，从而有利于精确地实现识别任务。Mel 频率尺度的值大体上对应于实际频率的对数分布关系。

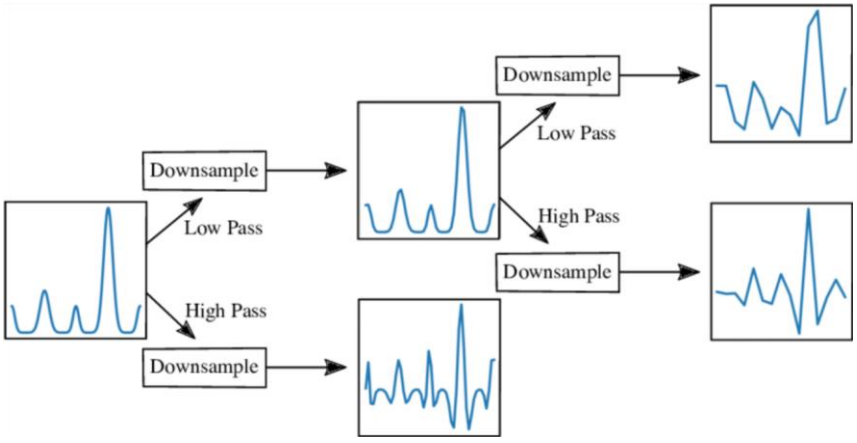


图 2.7 Fbank 特征提取

Fbank 与 MFCC 两者效果相比而言，有如下两点主要的不同：

- 计算量：MFCC 是在 FBank 的基础上进行的，所以 MFCC 的计算量更大。
- 特征区分度：FBank 特征相关性较高（相邻滤波器组有重叠），MFCC 具有更好的判别度，这也是在大多数语音识别论文中用的是 MFCC，而不是 FBank 的原因。

使用对角协方差矩阵的 GMM 由于忽略了不同特征维度的相关性，MFCC 更适合用来做特征。DNN/CNN 可以更好的利用这些相关性，使用 Fbank 特征可以更多地降低 WER。考虑到我们将采用深度学习网络模型，因此我们将使用 Fbank 算法，通过将音频数据加载为特征向量形式，然后将其输入深度学习模型进行抽取语音特征。最后采用 softmax 分类算法实现情感标签的分类任务。

2.3.6 Qt 界面

一般来说如果要使用 Python 开发跨平台的图形界面程序的话，主流的选择有以下几种：

- Tkinter：基于 Tk 的 Python 库，这是 Python 官方采用的标准库，优点是作为 Python 标准库、稳定、发布程序较小，缺点是控件相对较少。
- wxPython：基于 wxWidgets 的 Python 库，优点是控件比较丰富，缺点是稳定性相对差点、文档少、用户少。
- PySide2、PyQt5：基于 Qt 的 Python 库，优点是控件比较丰富、跨平台体验好、文档完善、用户多。缺点是库比较大，发布出来的程序比较大。

由于本项目功能相对较多，且界面比较复杂，因此我们使用到的 UI 组件是 PyQt5。

PyQt5 是 Digia 的一套 Qt5 应用框架与 python 的结合，Qt 库由 Riverbank Computing 开发，是最强大的 GUI 库之一 。其是由一系列 Python 模块组成。超过 620 个类，6000 函数和方法。能在诸如 Unix、Windows 和 Mac OS 等主流操作系统上运行。

2.4 实现功能

2.4.1 实时人脸裁剪和情感分析

通过部署的高清网络摄像头，对环境中的人的情绪进行识别和分析。整个流程为：首先通过 RetainFace 网络对图像中的人脸进行裁剪，然后裁剪得到人脸

信号被送入 EmoNet 网络中，进行情绪的分析和识别，输出为三个值，一个离散的情绪状态（Neutral、Happy、Sad、Surprise、Fear、Disgust、Anger、Contempt），两个连续的情绪指标（Valence、Arousal），我们将实时裁剪得到的人脸信号放置在右上角、将情绪指标放置在右下角进行显示（情绪的离散状态输出为字符串、将连续的两个情绪指标动态显示在坐标轴上）。

通过查看该界面的人脸裁剪和情感分析实况，可以及时的得到反馈，以便学习者进行调整学习状态。当然，在课堂教学等环境中，也可以在后台进行，节省资源的同时，以免造成对学习者的影响，因此，我们在这里做的实时人脸裁剪和情感分析展示界面是一个有多用途、可供选择的展示界面。该界面的运行资源占用在可接受范围内，能够满足实时性要求并在连续的一段学习时间中保持稳定的运行状态。

图 2.8 情绪评估界面

2.4.2 实时环境评估和物品检测

在学习质量评估中，另一项重要的指标就是学习环境的评估和检测，学习环境是影响学习状态的一个相关因素，因此，我们将其也纳入到我们的整体评估系统中。首先是考虑环境中的温湿度情况是否处于一个人体适宜的范围内（其中温度在 22 ~ 26℃为人体的适宜范围、湿度范围在 40 ~ 50%为人体的适宜范围），其次是传感器也能够检测到空气质量的情况，因此我们也将空气质量的信息也同时纳入到环境测评的信息栏中（空气质量应当在在 GB 3095—2012 的指定优良范围内）。

通过环境的温湿度、空气质量等信息的检测之后，我们同时也考虑到了学习环境内的电子产品的干扰因素，因此，我们引入了 Yolov5 物品检测算法，对环境中可能对学习造成的影响的手机等电子设备纳入到评分的体系中，当环境中出现手机等电子产品时，检测算法将相关物品及时提取出来贴出标签（包括人、手机和书籍的检测）。

同时，也在记录人员的是否在场等情况，用于考察学习人的学习态度。当检测中，出现手机等干扰物品或者是人员大部分时间不在场的情况下

最后，在针对在不同应用环境中的使用，本系统特别考虑到在家庭等隐私性要求较高的环境中使用场景，设计了隐私保护功能。针对采集的音视频信息会及时自动删除，只保留提取的情绪信息，并且进行必要的加密措施，访问这些数据需要采用账号、用户的登录方式。为了回溯情绪问题根源，系统可选择是否保留以前的情绪检测过程的报告等。

图 2.9 环境检测界面

2.4.3 学习评估分析报告

我们生成了学习评估报告，对学习时长、学习情绪、学习环境状态等指标进行统计和分析，并最终对学生学习状态进行评估并打分。同时，为了保证个人隐私，我们将对视频或音频数据进行及时的消除，生成的评估结果将避免出现引导性词汇，以减少使用者主观情绪的可能的影响。评估结果将以可视化图表形式展现，使得监护人可以直观了解到学生的学习状态。

在处理的学习状态图表信息中有三个图表展示，包括学习积极性的状态展示、情绪平和度的状态展示和环境指标变化展示。三个图表中的蓝色曲线是没有滤波处理的原始状态信息，而红色的曲线则是前面提到过的通过卡尔曼滤波得到的平滑之后的数据变化表示，容易看出有明显的改善和提高的效果，然后计算相关图标的方差、平均值等特征，用作最后评分所用的参考数据。

在经过上述的图表展示和预处理功能之后，我们还会生成最后的评分报告，包括对学习人的状态、态度等多方面进行定性的打分统计（由于目前对于学习状态没有统一的评判标准，因此我们自主定义了一套能够定性反应出学习人相对学习态度的评价标准，当情绪过激、态度不端正等清楚出现的时候，都会导致得分较低的情况发生，能够对学习人的相对学习情况的评判有所帮助），最终可以将整个报告放置在页面上用于学习人（或者监护人、教师等）时候的查阅和参考。

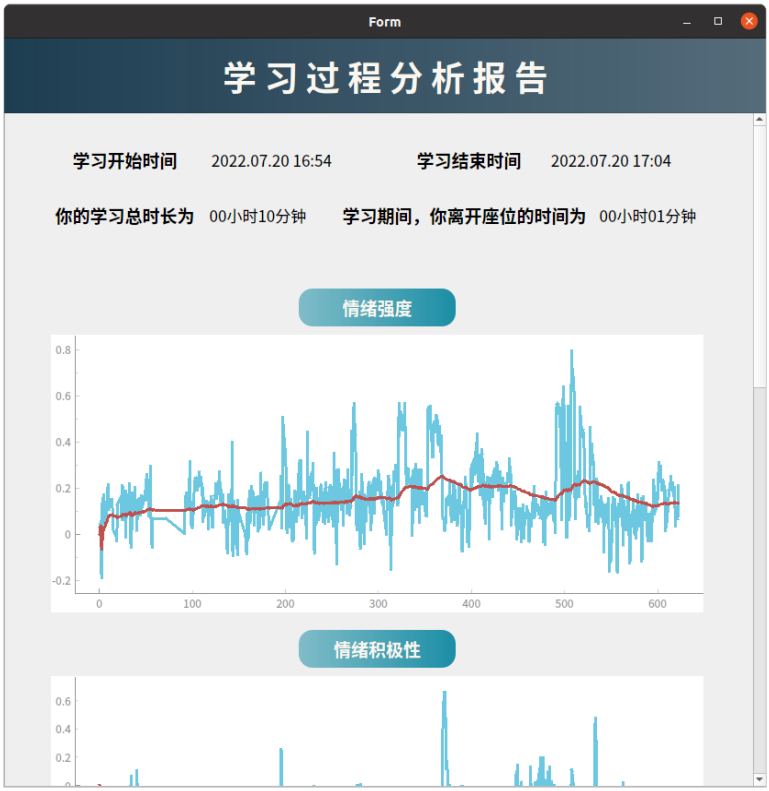


图 2.10 学习评估报告 - 1

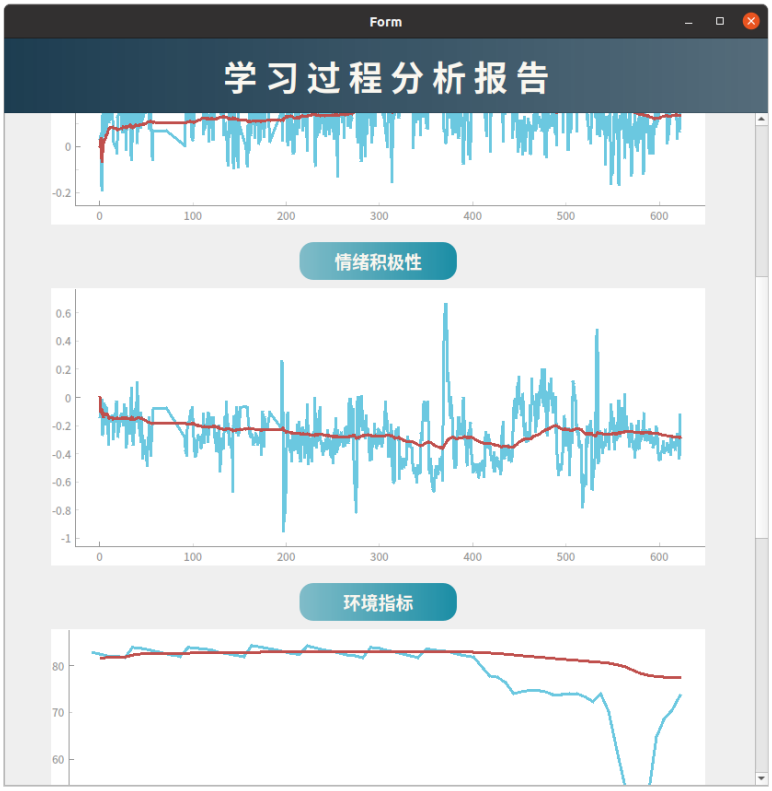


图 2.11 学习评估报告 - 2

图 2.12 学习评估报告 - 3

2.4.4 语音交互系统

为了实现更加智能的交互方式，我们在系统中嵌入了智能的语音助手（包括实时的语音唤醒功能、查询功能和连接到百度云智能语音助手上的语音交互功能），同时，我们设计了专门的语音交互界面，用户能够适时地查看到相关的语音交互的文字显示信息，以防没有合适的扬声设备或者环境不适合外放的情况。

其中，语音唤醒功能是全局打开的，在任何界面下，都能够通过呼叫“小特，小特”进行唤醒（我们经过了模型的训练，设定了“小特”进行唤醒，小特为比赛赞助方英特尔的简称）。在语音唤醒之后，小特能够实时地和用户进行对话和交流，以及提供一些有用的信息服务（例如天气预报）。在经过一段时间的语音信号检测后，如果没有检测到输入信号，则会进入休眠状态，重新等待被唤醒，降低了部分功耗和资源占用。

当用眼过度、情绪过激、不良坐姿、学习环境不适宜、长时负面情绪等情况时，进行语音提醒与交互能够帮助用户重新找到学习的状态。同时，使用者也可以通过语音能够得到更多维和立体的反馈信息，感觉到小特助手的人性化，体现出产品的人文情怀。

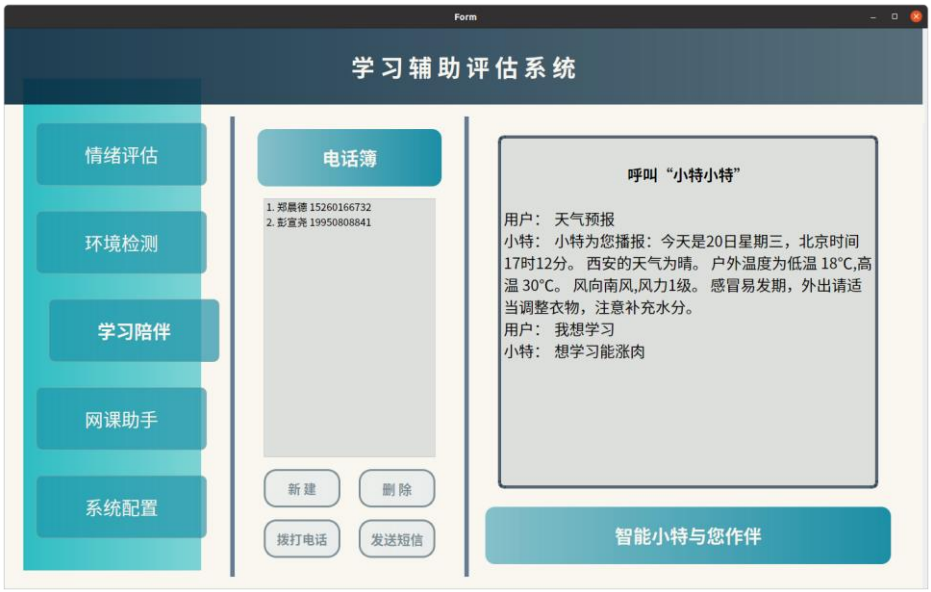


图 2.13 语音交互系统

2.4.5 电话通信系统

学习助手中，我们还设置了电话通信系统，用于进行 GSM 的电话通讯功能。我们将 GSM 模块嵌入到了整个系统中，并通过驱动调用到 python 环境中进行设置，通话系统需要有拨打电话和接听电话两个服务。我们进行拨打电话服务的时候是在主线程中，通过监听事件，调用了相关的槽函数进行电话的拨打。

同时，开启了一个 GSM 模块单独的线程，用于每 5s 钟进行一次电话事件的查询，如果查询到有电话拨打，就会弹出消息窗口进行提示接听或者挂断，实现了基础的电话接听和拨打的小功能。加入这一功能的主要考量是，考虑到在网课期间家长可能会外出，孩子在家可能会有远程联系的需要，增加这一功能后，孩子能够很方便的和家人取得远程的联系。

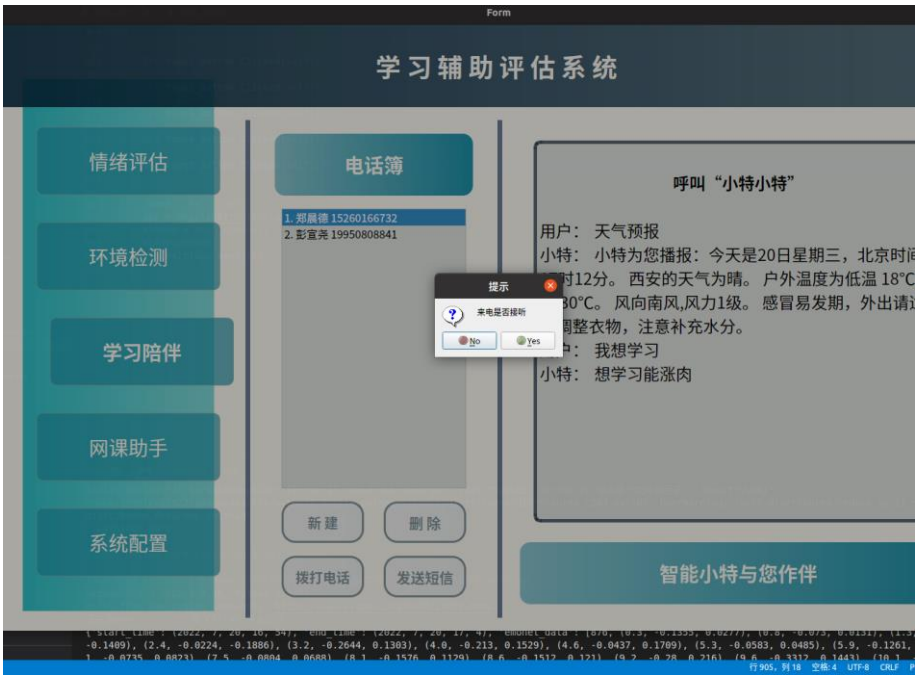


图 2.14 通话系统 - 1

图 2.13 通话系统 - 2

2.4.6 网课视频播放器

在实现以上基础的功能后，我们还实现了一个网课视频播放器的功能界面。在实际的网课学习中，大部分学生会需要对线上课堂的内容进行回放和复习。因此，我们在程序中增添了一个视频播放器的功能，能够从文件系统中读取到视频并进行播放、暂停、拖动等基础操作，以便及时的复习和巩固已经学习了的知识点。

综合上述，通过整合上述部分所描述的模块功能，得到了我们的整体的“小特智能学习助手”。小特智能学习助手是一个功能完备、交互友好、实时检测的基于多模态情感分析的智能学习交互助手。

2.5 实施方案

本次项目的预计实施方案如下：

2.5.1 市场调研与需求分析

预计实施时间：四月初至四月中下旬。

通过市场调研，确定产品的目标人群、市场规模与前景、并进行现有产品分析和优缺点比较，以确定当前目标人群需求。主要包含：功能需求、性能需求、可靠性需求、成本需求。并针对当前的技术水平、经济水平、政策规定完善功能设计，以满足可行性要求。

2.5.2 数据集搜集

预计实施时间：四月中旬至四月下旬。

常用的视频情感数据集有 AffectNet、LIRIS-ACCEDE（Audio-visual）、DEAP、HUMAINE（Audio+visual +gesture）、EMDB、MAHNOB-HCI、MMI（visual）等。

这些数据集大部分具备十万至百万的量级，足以支撑神经网络平台的训练效果。

同时，为了保证训练模型的实际应用效果，团队将通过平台自行采集数据，获取训练集，提升模型精度和适用性。

通过将模型在不同的数据集上进行训练，使得模型增强对于不同的环境及不同的人群泛化性，最后取得一个比较优秀的结果。

2.5.3 深度学习网络模型构建与训练

预计实施时间：五月初至五月中旬。

通过摄像头拍摄、开源数据集搜集等方式获取充分的样本供神经网络训练使用。深度学习是人工智能的一个重要组成部分，相较于传统机器学习方法具有精度高，操作简便等优势。通过对比当前流行的人脸情感分析模型与姿态检测模型，构建较优的网络模型，并通过不断的迭代实验进行模型的优化。

2.5.4 模型部署

预计实施时间：五月下旬至六月初。

本项目将在云服务器上进行网络模型构建及训练调优，随后通过 OpenVINO 软件或其他 AI 部署工具将模型部署在 GNS-V40 平台上，完成实时的推理计算。

2.5.5 系统测试与优化

预计实施时间：六月。

针对部署完毕的模型，布置场地进行实机训练与测试，主要针对：语音识别精确度、情绪判别准确度、系统反应灵敏度（运行速度）、测试结果可信度、系统稳定性等方面，对于暴露出的问题或不足将对系统进行修改调优。

第三部分 系统测试

3.1 测试设备

在本次项目中，我们模拟了真实的线上教学的情况，将整个套设备应用于一套独立的书桌上进行测试，我们将一个高清网络摄像头置于显示器的正上方，用于模拟线上教学中常见的前置摄像头的情况，然后再用另外一个摄像头置于身侧，用于检测环境或检测物品。

同时，我们验证了多人的情况，将两个摄像头布局在两个学习者周围，分别进行检测。通过切换账号能够切换到不同的学习者。多人的情况更加常见，例如在线上的教学中，往往涉及到对一个班级的学习状态进行评估，我们能够将这一套设备进一步地进行拓展，用于班级教学质量的评估。完成班级教学质量评估，切合了本次的边缘计算主题：通过在边缘端（教室）进行边缘化的班级教学质量评估，能够极大减少云计算中心的压力，将计算好的边缘端数据反馈回云平台（学校服务器）上，学校就能够通过获取到的数据进行学校整体的教学质量评估。

因此，我们进行了多人的测试方案，为整个系统的应用和发展拓宽了思路。

3.2 实时情感分析测试

实时情感分析测试中，我们进行的实时测试样例中，情绪识别结果中有大概92%符合的实际的离散情绪状态，我们选定的状态包括：自然、高兴、愤怒、惊讶、厌恶、恐惧、失落。其中的结果分析如下：

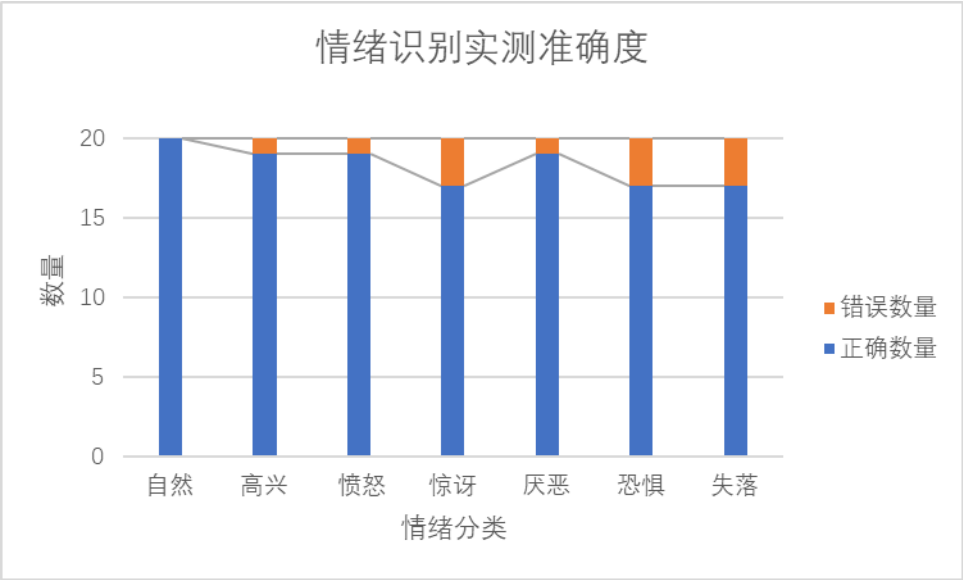


图 3.1 情绪识别准确度

其中，自然状态下，分类器的识别准确度最高，对于惊讶、恐惧等偏激情绪的识别正确度会稍微有一些偏低，可能是由于原始训练时所用到的数据集中的关

于愤怒的表情标签存在部分和显示中的实际状态不相符的情况，但在实际情况中，偏激情绪出现的频次将不会太高，不会太大影响到实际的运用。

同时，在帧率测试中，情感分析的帧率大概保持在 1 ~ 2 FPS 左右。

3.3 物品检测测试

物品检测时通过部署 Yolov5 算法在嵌入式平台上进行的，我们在物品检测的测试中，主要对环境中出现的人、书籍和手机进行了识别，因此，我们对这三者的准确度情况进行了测试。在测试过程中，我们通过对记录一段时间内，某种出现在环境中的物体的未检测到的次数进行记录，从而反推得到其检测成功率。结果如下：

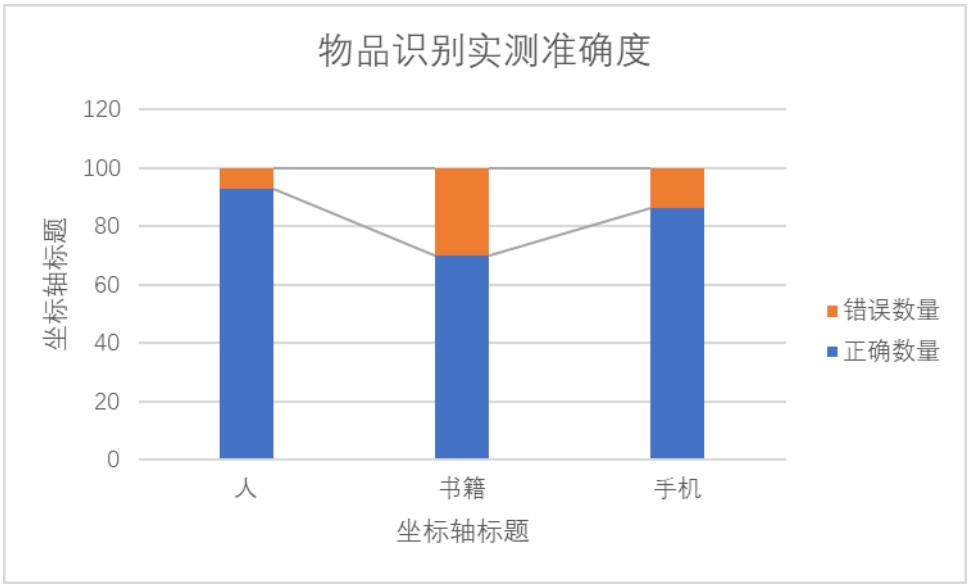


图 3.2 物品识别准确度

在实际测量的情况中，我们发现书籍的检测成功率偏低，可能原因是在数据集中的书籍情况和实际的书籍情况有所差别（实际测试中，只有当数据摆放位置较为远的时候，测出的准确路较高）；在其他情况下，人和手机在各种角度下测出率都比较高，满足了系统的实时性要求，能够在实际的系统中进行测试和使用。

在帧率测试中，稳定运行时的物品识别帧率维持在 3 ~ 4 FPS。

3.4 环境检测测试

关于环境的测试，我们在实际测试中中，用电子温湿度计对实际的环境信息进行了测试，并和传感器得到的温湿度信息进行对比。其温度测量值和变化范围符合合理的误差范围，由于室内的温度变化范围较小，且日常设备工作在正常室温下，因此未作极端环境条件测试。

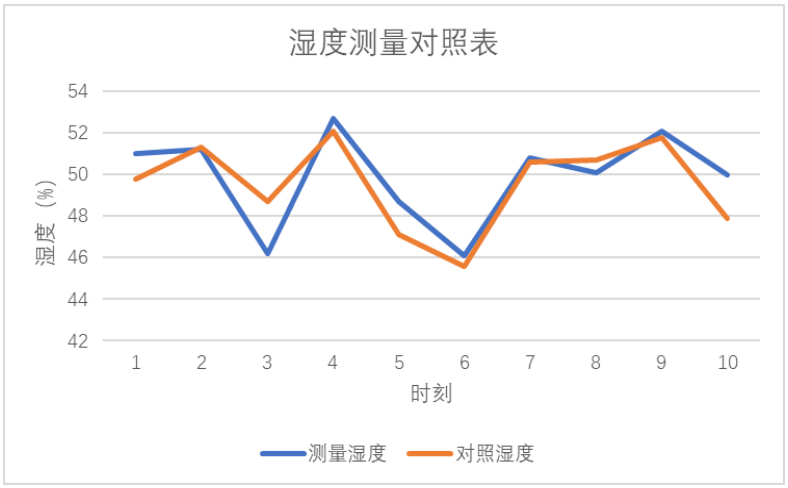


图 3.3 湿度测量对照

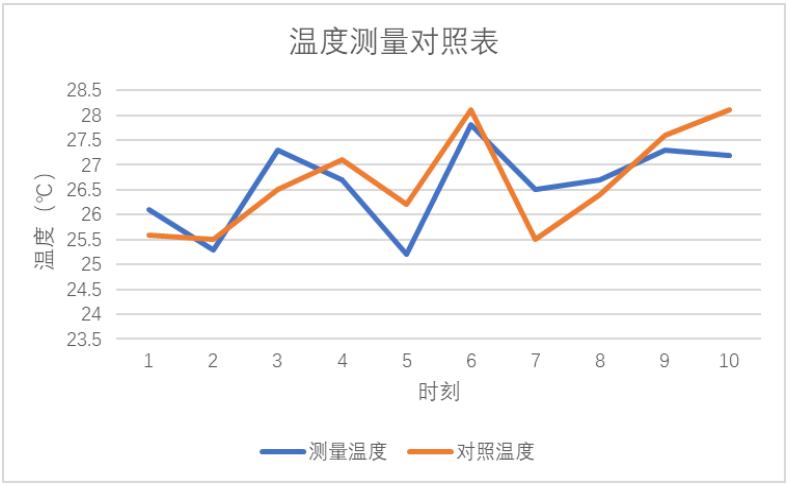


图 3.4 温度测量对照

由于设备条件限制，不能够对环境测试中的各项气体传感器测量值进行检测，因此，暂时未对其进行测量和标注。

3.5 语音交互系统测试

对于语音交互系统的测试，我们主要是针对唤醒成功率和识别的语音文字准确度来进行测试，在实际的测试中，系统表现十分灵敏，能够在 1s 的延迟内做到比较高准确度的唤醒。我们针对交互系统的测试结果如下：

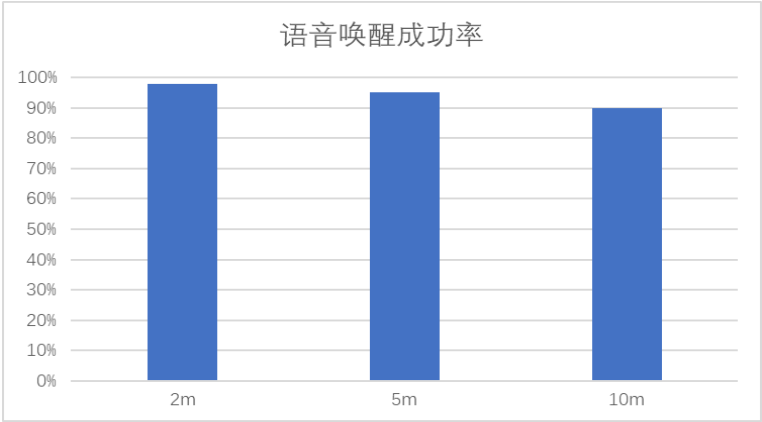


图 3.5 语音唤醒成功率

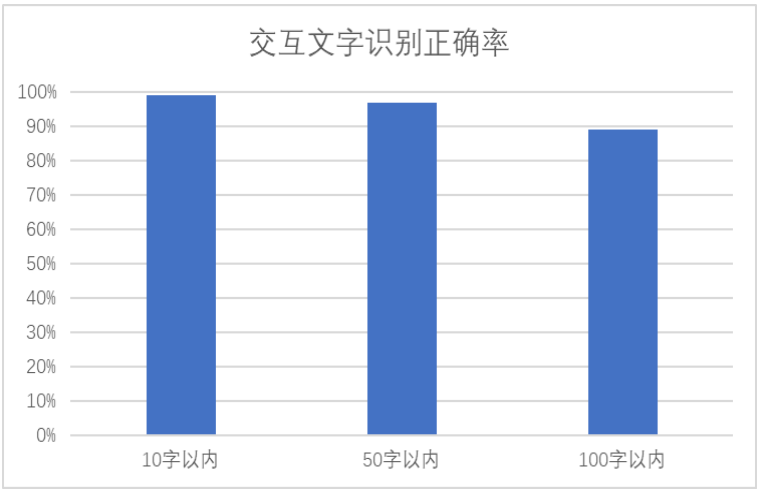


图 3.6 语音识别正确率

3.6 电话通信系统测试

实际测量电话通信系统的稳定性时，我们测试了多组通讯实验，目前通过率为 100%（包括接听电话和拨打电话），系统的电话通讯模块处于稳定的状态。但同时，由于电话无线通讯信号会对视频信号传输线有一定的干扰。因此，在实际的部署过程中，我们更换了抗干扰能力更强的视频传输线和显示器，问题得以解决，能够在通讯状态良好的同时，不会干扰到视频内容的显示。

第四部分 系统特色

4.1 多模态融合，稳定高效

系统集成了人体面部特征点提取、物品轮廓识别、目标检测、情感分析、语音识别、语义分析、环境监测和智能交互等技术，基于多模态信息拟合了学习评估模型，具备泛用性和稳定性。

4.2 连续情绪与学习评估系统

与常规的情感分类网络不同，系统开创性地采用了连续型指标用以衡量情绪状态，这类技术可以有效的处理样本中的负标签信息，同时能使学习评估模型拟合精确度提升。

4.3 充分挖掘系统资源，提高性能

系统使用了前沿的 Knowledge Distillation 技术，在复杂数据集上进行知识提取，完成了神经网络模型的压缩，大幅度提高了模型在 GNS-V40 边缘计算主机上的运行效率。同时，采用专门针对于嵌入式系统的 YOLOv5 目标检测算法，利用 OpenVINO 和 Intel VPU 计算单元实现了神经元加速，满足了系统的实时性要求。

4.4 系统可拓展性好，成本低

系统采用了成熟的外设模块，并且项目符合边缘计算的理念。情感识别，物品监测和学习评估均可在本机上完成，同时，系统具备了账号切换功能，通过增加摄像头即可实现用户数量的拓展。

4.5 交叉应用潜力大

项目充分发挥了边缘计算和深度学习的技术优势，除开线上教育家庭辅助，还可广泛应用于学校的教学质量评估系统。同时，借助于数据库的账号统计和云端计算，系统可以用于打造智能家庭-智能课堂双线教育平台。同时，项目中情感识别等关键技术亦可应用于其他的心理健康领域。

4.6 其他

总体上来看，我们的系统功能丰富，主要可以实现任务情感识别，环境智能检测，智能人机交互，学习状态评估等多个功能，并且各部分均较为完善，从各功能点来看：(1)由于本系统功能复杂，集成网络较多，因此对于算力的需求较大，在总算力有限的情况下，我们对各个网络进行了计算简化，在预测时间及准确率可接受的范围内最大程度的保证了多功能的并行。(2)为了确保各功能预测的时间，我们使用 OpenVINO 套件对神经网络计算进行加速，使系统能满足实时性的要求。(3)系统安全性较强，对于所有的数据处理均在本地进行，并且无云端备份，最大程度上保护了用户的隐私需求。(4)该项目符合边缘计算+AI 的理念，在项目中多次使用到目前最优秀的神经网络预测技术，包括用于情感分析的 EmoNet 网络，用于物体检测的 YOLOv5 网络等，在计算实现上，我们所有的计算都是基于由 Intel 提供的边缘计算平台来完成的，而无需在云端处理。

第五部分 团队组成

5.1 团队情况

本次项目团队成员包括：郑晨德、彭宣尧、霍嘉。各成员特长与分工如下：

郑晨德：擅长于语音数字信号处理、信号通信、嵌入式系统开发、计算机视觉处理。本次项目中主要负责语音情感分析模型、数据库处理、软硬件架构通信连接、学习状态评估模型建立等。并作为项目负责人保证团队工作的积极稳定进行。

彭宣尧：擅长于硬件架构处理、信号采集与处理、嵌入式系统开发、FPGA开发。本次项目中主要负责硬件选型、硬件架构搭建、硬件加速、系统级联、系统测试优化、交互界面开发等。

霍嘉：擅长于深度学习模型架构、模式识别算法、网络协议。本次项目中主要负责模型构建、学习状态评估模型建立、深度学习模型训练、模型部署等。

5.2 防疫安全保障

由于防疫安全问题，在项目实施过程中，小组将采取“非必要不线下”的原则。若无线下测试要求，则积极争取线上完成。为保证工作进度，定期开展线上会议，并作会议记录。线下使用实验室或前往实验场所时坚决佩戴口罩，并做好定期消毒工作。项目全程将完全、坚决服从学校的防疫政策，听从学校安排，保证防疫秩序。

参考文献

[1] 教育部光明网 https://m.gmw.cn/baijia/2020-05/14/1301222911.html
[2] 教育部 http://china.qianlong.com/2020/0515/4142082.shtml
[3] 唐晓波, & 刘广超. (2017). 细粒度情感分析研究综述. *图书情报工作*, *61*(5), 132.
[4] 刘继明, et al. "多模态的情感分析技术综述." *计算机科学与探索* 15.7 (2021): 1165-1182.
[5] 朱长水，丁勇，袁宝华，等. 融合 LBP 和 LPQ 的人脸识别[J]. 南京师大学报(自然科学版),, 2015 年(01 期).
[6] 中国国民心理健康发展报告 https://fjrb.fjdaily.com/pad/con/202104/13/content_65525.html
[7] *"Estimation of continuous valence and arousal levels from faces in naturalistic conditions"*, Antoine Toisoul, Jean Kossaifi, Adrian Bulat, Georgios Tzimiropoulos and Maja Pantic, published in Nature Machine Intelligence, January 2021
[8] * Geoffrey Hinton, Oriol Vinyals, Jeff Dean: “Distilling the Knowledge in a Neural Network”, 2015; [http://arxiv.org/abs/1503.02531 arXiv:1503.02531]
[9] * Xingkui Zhu, Shuchang Lyu, Xu Wang, Qi Zhao: “TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-captured Scenarios”, 2021; [http://arxiv.org/abs/2108.11539 arXiv:2108.11539].
[10] 彭丁聪. "卡尔曼滤波的基本原理及应用." *软件导刊* 8.11 (2009): 32-34.
[11] * Jiankang Deng, Jia Guo, Yuxiang Zhou, Jinke Yu, Irene Kotsia, Stefanos Zafeiriou: “RetinaFace: Single-stage Dense Face Localisation in the Wild”, 2019; [http://arxiv.org/abs/1905.00641 arXiv:1905.00641].
[12] * Karen Simonyan, Andrew Zisserman: “Very Deep Convolutional Networks for Large-Scale Image Recognition”, 2014; [http://arxiv.org/abs/1409.1556 arXiv:1409.1556].
[13] * Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger: “Densely Connected Convolutional Networks”, 2016; [http://arxiv.org/abs/1608.06993 arXiv:1608.06993].

附录

源代码

```
# -*- coding: utf-8 -*-

from hashlib import new
import os
import sys
import time
import cv2
import pickle
import serial
from functools import partial

import qdarkstyle
from PyQt5 import QtGui, QtWidgets, QtCore
from PyQt5.QtCore import *
from PyQt5.QtWidgets import *
from PyQt5.QtGui import *
from PyQt5.QtMultimedia import *
from sympy import im

import emonet_func
from UI import Main_GUI, NewContact_GUI, Changekey_GUI, Changeinfo_GUI

# from speech_wakeup import speech_wakeup_start

# from emonet import camera

dirname, py_filename = os.path.split(os.path.abspath(__file__))


def convert_table(data):
    with open(os.path.join(os.getcwd(), "UI", "TABLE.html"), mode="r",
encoding="UTF-8") as f:
        table_html = f.read()
    table_html = table_html.format(data[0], data[1], data[2], data[3],
data[4], data[5], data[6])
    return table_html
```

```
def save_variable(v, filename):
    f = open(filename, 'wb')
    pickle.dump(v, f)
    f.close()
    return filename


def load_variavle(filename):
    f = open(filename, 'rb')
    r = pickle.load(f)
    f.close()
    return r


class Camera_Thread(QThread):
    camera_signal = pyqtSignal(str)

    def __init__(self):
        super().__init__()

    def run(self):
        cap = cv2.VideoCapture(0)
        while True:
            rawpngpath = "results/picture_raw.png"
            ret, frame = cap.read()  # 读取摄像头画面
            cv2.imwrite(rawpngpath, frame)
            self.camera_signal.emit(rawpngpath)


class Emonet_Thread(QThread):
    rawfacepath_signal = pyqtSignal(str)
    facepngpath_signal = pyqtSignal(str, str, float, float)

    def __init__(self):
        super().__init__()
        self.retainFace = emonet_func.load_retainFace()
        self.emoNet = emonet_func.load_emoNet()
        self.run_sign = True

    def run(self):
        cap = cv2.VideoCapture(0)
        while self.run_sign == True:
            rawpngpath = "results/picture_raw.png"
            # 框出人脸的原画
            rawfacepath = "results/picture_raw_face_raw.png"
```

```
            crop_img_path = emonet_func.crop_face(self.retainFace,
rawpngpath)
            # 如果未检测到人脸，打印
            if crop_img_path is None:
                # print("未检测到人脸")
                pass
            # 如果检测到人脸就进行情绪识别
            else:
                expression, violence, arousal =
emonet_func.interface_emotion(crop_img_path, self.emoNet)
                self.rawfacepath_signal.emit(rawfacepath)
                self.facepngpath_signal.emit(crop_img_path, expression,
violence, arousal)

    def stop(self):
        self.run_sign = False


class snowboy_thread(QThread):
    snowboy_signal = pyqtSignal(tuple)

    def __init__(self) -> None:
        super().__init__()

    def run(self):
        # speech_wakeup_start.main(self.snowboy_signal)
        pass


class M702_Thread(QThread):
    M702_signal = pyqtSignal(tuple)

    def __init__(self, ser) -> None:
        super().__init__()
        self.ser = ser

    def run(self):
        data = [0] * 17
        while True:
            data[0] = self.ser.read()
            checksum = data[0][0]
            data[1] = self.ser.read()
            checksum += data[1][0]
            if data[0] == b'\x3c' and data[1] == b'\x02':
```

```
                for i in range(2, 16):
                    data[i] = self.ser.read()
                    checksum += data[i][0]
                data[16] = self.ser.read()
                checksum = checksum & 0xff
                if checksum == data[16][0]:
                    CO2 = data[2][0] * 128 + data[3][0]
                    eCH2O = data[4][0] * 128 + data[5][0]
                    TVOC = data[6][0] * 128 + data[7][0]
                    PM25 = data[8][0] * 128 + data[9][0]
                    PM10 = data[10][0] * 128 + data[11][0]
                    Temperature = float(str(data[12][0]) + '.' +
str(data[13][0]))
                    Humidity = float(str(data[14][0]) + '.' +
str(data[15][0]))
                    self.M702_signal.emit((CO2, eCH2O, TVOC, PM25, PM10,
Temperature, Humidity))
                    time.sleep(1)


class GSM_Thread(QThread):
    GSM_signal = pyqtSignal(str)

    def __init__(self, ser) -> None:
        super().__init__()
        self.ser = ser

    def run(self):
        self.ser.write("ATE0\n".encode('utf-8'))
        while True:
            check_data = self.ser.read()
            if check_data == b'R':
                check_data = self.ser.read(7)
                if check_data == b'ING\r\n\r\n':
                    self.GSM_signal.emit('ring')
                    time.sleep(5)


class Changeinfo_Window(Changeinfo_GUI.Ui_Form, QMainWindow):
    chengeinfo_signal = pyqtSignal(tuple)

    def __init__(self) -> None:
        super().__init__()
        self.setupUi(self)
```

```
        self.setWindowOpacity(0.97)
        self.Button_Confirm.clicked.connect(self.Confirm)
        self.Button_Cancel.clicked.connect(self.Cancel)

    def Confirm(self):
        name = self.Name.text()
        sex = self.Sex.text()
        grade = self.Grade.text()
        QMessageBox.information(self, '提示', '信息修改成功',
QMessageBox.Yes, QMessageBox.Yes)
        self.chengeinfo_signal.emit((name, sex, grade))

    def Cancel(self):
        self.chengeinfo_signal.emit(('cancel', '1', '2'))


class Changekey_Window(Changekey_GUI.Ui_Form, QMainWindow):
    changekey_signal = pyqtSignal(str)

    def __init__(self, userdata) -> None:
        super().__init__()
        self.setupUi(self)
        self.setWindowOpacity(0.97)
        self.userdata = userdata
        self.Button_Confirm.clicked.connect(self.Confirm)
        self.Button_Cancel.clicked.connect(self.Cancel)

    def Confirm(self):
        if self.Originkey.text() == self.userdata['key']:
            if self.Newkey.text() == self.Newkey_2.text():
                QMessageBox.information(self, '提示', '密码修改成功',
QMessageBox.Yes, QMessageBox.Yes)
                self.userdata['key'] = self.Newkey.text()
                self.changekey_signal.emit(self.Newkey.text())
            else:
                QMessageBox.information(self, '警告', '两次新密码输入不一致
', QMessageBox.Yes, QMessageBox.Yes)
        else:
            QMessageBox.information(self, '警告', '密码错误',
QMessageBox.Yes, QMessageBox.Yes)

    def Cancel(self):
        self.changekey_signal.emit('cancel')
```

```
class NewContact_Window(NewContact_GUI.Ui_Form, QMainWindow):
    newcontact_signal = pyqtSignal(tuple)

    def __init__(self) -> None:
        super().__init__()
        self.setupUi(self)
        self.setWindowOpacity(0.97)
        self.Button_Confirm.clicked.connect(self.Confirm)
        self.Button_Cancel.clicked.connect(self.Cancel)

    def Confirm(self):
        para = (self.Name.toPlainText(), self.Phonenum.toPlainText())
        self.newcontact_signal.emit(para)

    def Cancel(self):
        para = ('cancel', 'cancel')
        self.newcontact_signal.emit(para)


class Main_Window(Main_GUI.Ui_Form, QMainWindow):
    def __init__(self):
        super().__init__()
        self.setupUi(self)

        self.P5_Exit_Button.hide()  # 暂时不用

        self.emonet_flag = False
        self.emonet_data = [0]
        self.env_data = [0]

        # snowboy
        self.Snowboy_thread = snowboy_thread()
        self.Snowboy_thread.snowboy_signal.connect(self.Robotmsg_Show)
        self.Snowboy_thread.start()

        # 设置窗口透明
        self.setWindowOpacity(0.97)

        # 设置各个视频匹配窗口大小
        self.P1_Video_Raw.setScaledContents(True)
        self.P1_Video_Head.setScaledContents(True)
        self.P1_Mood.setScaledContents(True)
        self.P2_Video_Raw.setScaledContents(True)
```

```
        # 载入电话簿
        self.phonelist = load_variavle('userdata/phonelist.txt')
        for i in range(1, self.phonelist[0] + 1):
            self.P3_Phone_List.addItem(str(i) + '. ' + self.phonelist[i][0]
+ ' ' + self.phonelist[i][1])

        self.GSM_ser = serial.Serial("/dev/ttyUSB1", 115200, timeout=15)
        self.GSM_thread = GSM_Thread(self.GSM_ser)
        self.GSM_thread.GSM_signal.connect(self.Callin)
        self.GSM_thread.start()

        # 载入用户信息
        # results={'account':'intelcup','key':'123456','name':'彭晨嘉
','sex':'男','grade':'小学五年级','learningtimes':8,'learningtime':8000}
        self.userdata = load_variavle('userdata/intelcup.txt')
        self.P5_Account.setText(self.userdata['account'])
        self.P5_Name.setText(self.userdata['name'])
        self.P5_Sex.setText(self.userdata['sex'])
        self.P5_Grade.setText(self.userdata['grade'])
        self.P5_Learningtimes.setText(str(self.userdata['learningtimes']
))
        self.P5_Learningtime.setText(str(self.userdata['learningtime']))

        # 初始化子窗口与信号连接
        self.Child_NewContact = NewContact_Window()
        self.Child_NewContact.newcontact_signal.connect(self.AddContact)
        self.Child_Changekey = Changekey_Window(self.userdata)
        self.Child_Changekey.changekey_signal.connect(self.ChangeKey)
        self.Child_Changeinfo = Changeinfo_Window()
        self.Child_Changeinfo.chengeinfo_signal.connect(self.Changeinfo)
        self.Child_Changeinfo.Name.setText(self.userdata['name'])
        self.Child_Changeinfo.Sex.setText(self.userdata['sex'])
        self.Child_Changeinfo.Grade.setText(self.userdata['grade'])

        # 设置页面 1 的功能与信号链接
        self.Button_Emonet_Stop.hide()
        self.P1_Learning_Time.hide()
        self.Button_Emonet_Start.clicked.connect(self.Emonet_Start)
        self.Button_Emonet_Stop.clicked.connect(self.Emonet_Stop)
        self.P1_Emonet_Time = 0
        self.P1_Emonet_Timer = QTimer()
        self.camera_thread = Camera_Thread()
        self.camera_thread.camera_signal.connect(self.Video_Raw_Show)
        self.camera_thread.start()
```

```
        # 设置页面 2 的功能与信号连接
        self.M702_thread = M702_Thread(serial.Serial("/dev/ttyUSB0",
9600))
        self.M702_thread.M702_signal.connect(self.Env_Msg_Show)
        self.M702_thread.start()

        # 设置页面 3 的功能与信号连接
        self.P3_Newcontact_Button.clicked.connect(self.NewContact)
        self.P3_Deletecontact_Button.clicked.connect(self.DeleteContact)
        self.P3_Phonecall_Button.clicked.connect(self.PhoneCall)
        self.P3_Sendmsg_Button.clicked.connect(self.SendMsg)

        # 设置页面 4 的功能与信号链接
        self.P4_Video_Timer = QTimer()  # 视频进度条计时器
        self.P4_Bar_Maxnum = 1000  # 视频进度条最大值
        self.P4_Video_Player = QMediaPlayer()  # 视频播放变量
        self.P4_Video_Player.setVideoOutput(self.P4_Video)  # 设置窗口
        self.P4_Selectfile_Button.clicked.connect(self.openVideoFile)  #
选择视频文件按钮
        self.P4_Play_Button.clicked.connect(self.playVideo)  # 播放按钮
        self.P4_Susbend_Button.clicked.connect(self.pauseVideo)  # 暂停按
钮
        self.P4_Progress_Bar.sliderMoved.connect(self.slider_progress_mo
ved)  # 拖动进度条
        self.P4_Progress_Bar.sliderReleased.connect(self.slider_progress
_released)  # 拖动进度条后释放

        # 设置页面 5 的功能与信号链接
        self.P5_Changekey_Button.clicked.connect(self.changekey_pageshow
)
        self.P5_Changemsg_Button.clicked.connect(self.changeinfo_pagesho
w)

        # 切换页面
        self.Page1_Button.clicked.connect(self.Page1_Button_Clicked)
        self.Page2_Button.clicked.connect(self.Page2_Button_Clicked)
        self.Page3_Button.clicked.connect(self.Page3_Button_Clicked)
        self.Page4_Button.clicked.connect(self.Page4_Button_Clicked)
        self.Page5_Button.clicked.connect(self.Page5_Button_Clicked)

    def test(self, para):
        print(para)
```

```
    # Page1 Functionsd
    def Emonet_Start(self):
        print("Emonet_Start")
        self.emonet_flag = True
        self.Button_Emonet_Start.hide()
        self.Button_Emonet_Stop.show()
        self.emonet_data = [0]
        self.env_data = [0]
        self.P1_Learning_Time.show()
        self.emonet_thread = Emonet_Thread()
        self.emonet_thread.rawfacepath_signal.connect(self.Video_face_ra
w_Show)
        self.emonet_thread.facepngpath_signal.connect(self.Video_face_Sh
ow)
        self.emonet_thread.start()
        self.P1_Emonet_Time = 0
        self.P1_Emonet_Timer.setInterval(100)
        self.P1_Emonet_Timer.start()
        self.P1_Emonet_Timer.timeout.connect(self.P1_Emonet_Timeout)

    def P1_Emonet_Timeout(self):
        self.P1_Emonet_Time += 0.1
        min1, sec1 = divmod(self.P1_Emonet_Time, 60)
        hour1, min1 = divmod(min1, 60)
        self.P1_Learning_Time.setText("%02d:%02d:%02d" % (hour1, min1,
sec1))

    def Video_face_Show(self, facepngpath, expression, violence, arousal):
        # print(facepngpath)
        # 设置情绪识别结果的字体大小
        font = QtGui.QFont()
        font.setFamily("微软雅黑")
        font.setPointSize(10)
        self.P1_Mood.setFont(font)
        self.P1_Mood.setText(f'expression = {expression} \nviolence =
{violence} \narousal = {arousal}')
        self.P1_Video_Head.setPixmap(QtGui.QPixmap(facepngpath))
        self.Mood_Point.setGeometry(100 + violence * 100, 112 - arousal *
100, 51, 51)
        self.emonet_data[0] += 1
        self.emonet_data.append(
            (float(format(self.P1_Emonet_Time, '.1f')),
float(format(violence, '.4f')), float(format(arousal, '.4f'))))
```

```
    def Video_face_raw_Show(self, facerawpngpath):
        self.P2_Video_Raw.setPixmap(QtGui.QPixmap(facerawpngpath))

    def Video_Raw_Show(self, rawpngpath):
        # print(rawpngpath)
        self.P1_Video_Raw.setPixmap(QtGui.QPixmap(rawpngpath))

    def Emonet_Stop(self):
        self.emonet_flag = False
        self.emonet_thread.stop()
        self.P1_Emonet_Timer.stop()
        self.Button_Emonet_Start.show()
        self.Button_Emonet_Stop.hide()
        self.P1_Learning_Time.hide()
        print(self.emonet_data)
        print(self.env_data)

        save_variable(self.emonet_data, 'userdata/emonet_data.txt')
        save_variable(self.env_data, 'userdata/env_data.txt')

        pass

    # Page2 Functions
    def Env_Msg_Show(self, msg):
        _translate = QtCore.QCoreApplication.translate
        CO2, eCH2O, TVOC, PM25, PM10, Temperature, Humidity = msg
        mark = 83

        if self.emonet_flag == True:
            self.env_data[0] += 1
            self.env_data.append(
                (float(format(self.P1_Emonet_Time, '.1f')), CO2, eCH2O,
TVOC, PM25, PM10, Temperature, Humidity, mark))

        html_content = convert_table(msg)
        self.P2_Env_Msg.setHtml(_translate("Form",
                                           "<!DOCTYPE HTML PUBLIC
\"-//W3C//DTD HTML 4.0//EN\"
\"http://www.w3.org/TR/REC-html40/strict.dtd\">\n"
                                           "<html><head><meta
name=\"qrichtext\" content=\"1\" /><style type=\"text/css\">\n"
                                           "p, li { white-space: pre-wrap; }
\n"
```

```
                                        "</style></head><body style=\"
font-family:\'微软雅黑\'; font-size:8pt; font-weight:400;
font-style:normal;\">\n"
                                        "<p align=\"center\"
style=\"-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0;
text-indent:0px;\"><br /></p>\n"
                                        "<p align=\"center\" style=\"
margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px;
-qt-block-indent:0; text-indent:0px;\"><span style=\" font-size:20pt;
font-weight:600;\">智能小特为您服务</span></p>\n"
                                        "<p align=\"center\"
style=\"-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;
font-size:20pt; font-weight:600;\"><br /></p>\n"
                                        "<p style=\" margin-top:0px;
margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0;
text-indent:0px;\"><span style=\" font-size:14pt;\">    当前环境条件如下：
</span></p>\n"
                                        "<p
style=\"-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px;
font-size:16pt;\"><br /></p>\n"
                                        + html_content +
                                        "<p style=\" margin-top:0px;
margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0;
text-indent:0px;\"><span style=\" font-size:14pt;
vertical-align:super;\">    </span></p>\n"
                                        "<p style=\" margin-top:0px;
margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0;
text-indent:0px;\"><span style=\" font-size:14pt;\">    目前环境的总体评分
为：</span></p>\n"
                                        f"<p align=\"center\" style=\"
margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px;
-qt-block-indent:0; text-indent:0px;\"><span style=\" font-size:36pt;
font-weight:600;\">{mark}分</span></p>\n"
                                        "<p style=\" margin-top:0px;
margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0;
text-indent:0px;\"><span style=\" font-size:14pt;\">    </span></p>\n"
                                        "<p style=\" margin-top:0px;
margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0;
text-indent:0px;\"><span style=\" font-size:14pt;\">    现在的环境很适合学
习呢，请多多加油！</span></p>\n"
```

```
                                        "<p align=\"center\" style=\"
margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px;
-qt-block-indent:0; text-indent:0px;\"><span style=\"
font-weight:600;\">    </span></p></body></html>"))

    # Page3 Functions
    def NewContact(self):
        self.Child_NewContact.show()

    def AddContact(self, para):
        self.Child_NewContact.hide()
        if para != ('cancel', 'cancel'):
            self.phonelist[self.phonelist[0] + 1] = (para[0], para[1])
            self.phonelist[0] += 1
            save_variable(self.phonelist, 'userdata/phonelist.txt')
            self.P3_Phone_List.addItem(str(self.phonelist[0]) + '. ' +
para[0] + ' ' + para[1])
            self.Child_NewContact.Name.setText('')
            self.Child_NewContact.Phonenum.setText('')
        else:
            print('cancel')

    def DeleteContact(self):
        select = self.P3_Phone_List.currentRow()
        reply = QMessageBox.question(self, '警告', '是否删除联系人',
QMessageBox.Yes | QMessageBox.No, QMessageBox.No)
        if reply == QMessageBox.Yes:
            self.P3_Phone_List.clear()
            for i in range(select + 1, self.phonelist[0]):
                self.phonelist[i] = self.phonelist[i + 1]
            self.phonelist[self.phonelist[0]] = 0
            self.phonelist[0] -= 1
            for i in range(1, self.phonelist[0] + 1):
                self.P3_Phone_List.addItem(str(i) + '. ' +
self.phonelist[i][0] + ' ' + self.phonelist[i][1])
            save_variable(self.phonelist, 'userdata/phonelist.txt')

    def PhoneCall(self):
        select = self.P3_Phone_List.currentRow()
        phonenum = self.phonelist[select + 1][1]
        self.GSM_thread.terminate()
        head = "ATD" + str(phonenum) + ";\n"
        write_len = self.GSM_ser.write(head.encode('utf-8'))
        re_data = self.GSM_ser.read(write_len + 5)
```

```
        print(re_data)
        print(phonenum)  # 待修改
        self.GSM_thread.start()

    def SendMsg(self):
        select = self.P3_Phone_List.currentRow()
        phonenum = self.phonelist[select + 1][1]
        print(phonenum)  # 待修改
        # msg = keyboard
        # msg = video

    def Callin(self, para):
        self.GSM_thread.terminate()
        if para == 'ring':
            reply = QMessageBox.question(self, '提示', '来电是否接听',
QMessageBox.Yes | QMessageBox.No, QMessageBox.No)
            if reply == QMessageBox.Yes:
                write_len = self.GSM_ser.write("ATA\n".encode('utf-8'))
                re_data = self.GSM_ser.read(write_len + 1)  # ATA\r\n
                re_data = self.GSM_ser.read(4)  # OK\r\n
            else:
                write_len = self.GSM_ser.write("ATH\n".encode('utf-8'))
                re_data = self.GSM_ser.read(write_len + 1)  # ATA\r\n
                re_data = self.GSM_ser.read(4)  # OK\r\n
        self.GSM_thread.start()

    def Robotmsg_Show(self, msg):
        usemsg = '用户：\t' + msg[1]
        robmsg = '小特：\t' + msg[3]
        self.P3_Robotmsg.append(usemsg)
        self.P3_Robotmsg.append(robmsg)

    # Page4 Functions
    def P4_Video_Timeout(self):
        self.P4_Progress_Bar.setValue(
            round(self.P4_Video_Player.position() * self.P4_Bar_Maxnum /
(self.P4_Video_Player.duration())))
        m, s = divmod(self.P4_Video_Player.position() / 1000, 60)
        h, m = divmod(m, 60)
        self.P4_Time_Label.setText("%02d:%02d:%02d" % (h, m, s))

    def openVideoFile(self):
        url = QFileDialog.getOpenFileUrl()
        videopath = url[0]
```

```
        self.P4_Video_Player.setMedia(QMediaContent(videopath))
        self.P4_video_title.setText("{}".format(videopath.fileName()))
        self.P4_Video_Player.play()
        self.P4_Video_Timer.setInterval(1000)
        self.P4_Video_Timer.start()
        self.P4_Video_Timer.timeout.connect(self.P4_Video_Timeout)
        self.P4_Video_Player.pause()

    def playVideo(self):
        self.P4_Video_Player.play()

    def pauseVideo(self):
        self.P4_Video_Player.pause()

    def slider_progress_moved(self):
        self.P4_Video_Timer.stop()
        self.P4_Video_Player.setPosition(
            round(self.P4_Progress_Bar.value() *
self.P4_Video_Player.duration() / self.P4_Bar_Maxnum))

    def slider_progress_released(self):
        self.P4_Video_Timer.start()

    # Page5 Functions
    def changekey_pageshow(self):
        self.Child_Changekey.show()

    def changeinfo_pageshow(self):
        self.Child_Changeinfo.show()

    def ChangeKey(self, newkey):
        if newkey != 'cancel':
            self.userdata['key'] = newkey
            save_variable(self.userdata, 'userdata/intelcup.txt')
            self.Child_Changekey.Originkey.setText('')
            self.Child_Changekey.Newkey.setText('')
            self.Child_Changekey.Newkey_2.setText('')
        self.Child_Changekey.hide()

    def Changeinfo(self, info):
        if info[0] != 'cancel':
            self.userdata['name'] = info[0]
            self.userdata['sex'] = info[1]
            self.userdata['grade'] = info[2]
```

```
            self.Child_Changeinfo.Name.setText(self.userdata['name'])
            self.Child_Changeinfo.Sex.setText(self.userdata['sex'])
            self.Child_Changeinfo.Grade.setText(self.userdata['grade'])
            self.P5_Name.setText(self.userdata['name'])
            self.P5_Sex.setText(self.userdata['sex'])
            self.P5_Grade.setText(self.userdata['grade'])
        self.Child_Changeinfo.hide()

    # Page Switch Functions
    def Page_Button_Rise(self, Button, vertical_position):
        font = QtGui.QFont()
        font.setFamily("微软雅黑")
        font.setPointSize(22)
        font.setBold(True)
        font.setItalic(False)
        font.setWeight(75)
        font.setStrikeOut(False)
        Button.setGeometry(QtCore.QRect(70, vertical_position, 270, 100))
        Button.setFont(font)
        Button.setStyleSheet(
            "QPushButton{background-color: qlineargradient(spread:pad,
x1:0, y1:0, x2:1, y2:0, stop:0 rgba(23, 140, 164,150), stop:1 rgba(23, 140,
164,170));\n"
            "color: rgb(249, 247, 240);\n"
            "border:1px solid rgba(161, 163, 161,170);\n"
            "border-radius:10px;}")

    def Page_Button_Down(self, Button, vertical_position):
        font = QtGui.QFont()
        font.setFamily("微软雅黑")
        font.setPointSize(22)
        font.setBold(False)
        font.setWeight(50)
        Button.setGeometry(QtCore.QRect(50, vertical_position, 270, 100))
        Button.setFont(font)
        Button.setStyleSheet(
            "QPushButton:hover{background-color:
qlineargradient(spread:pad, x1:0, y1:0, x2:1, y2:0, stop:0 rgba(249, 247,
240,120), stop:1 rgba(249, 247, 240,140));\n"
            "color: rgb(23, 140, 164);\n"
            "border:1px solid rgba(161, 163, 161,170);\n"
            "border-radius:10px;}\n"
```

```
            "QPushButton{background-color: qlineargradient(spread:pad,
x1:0, y1:0, x2:1, y2:0, stop:0 rgba(23, 140, 164,140), stop:1 rgba(23, 140,
164,160));\n"
            "color: rgb(249, 247, 240);\n"
            "border:1px solid rgba(161, 163, 161,170);\n"
            "border-radius:10px;}")

    def Page1_Button_Clicked(self):
        self.stackedWidget.setCurrentIndex(0)
        self.Page_Button_Rise(self.Page1_Button, 150)
        self.Page_Button_Down(self.Page2_Button, 290)
        self.Page_Button_Down(self.Page3_Button, 430)
        self.Page_Button_Down(self.Page4_Button, 570)
        self.Page_Button_Down(self.Page5_Button, 710)

    def Page2_Button_Clicked(self):
        self.stackedWidget.setCurrentIndex(1)
        self.Page_Button_Down(self.Page1_Button, 150)
        self.Page_Button_Rise(self.Page2_Button, 290)
        self.Page_Button_Down(self.Page3_Button, 430)
        self.Page_Button_Down(self.Page4_Button, 570)
        self.Page_Button_Down(self.Page5_Button, 710)

    def Page3_Button_Clicked(self):
        self.stackedWidget.setCurrentIndex(2)
        self.Page_Button_Down(self.Page1_Button, 150)
        self.Page_Button_Down(self.Page2_Button, 290)
        self.Page_Button_Rise(self.Page3_Button, 430)
        self.Page_Button_Down(self.Page4_Button, 570)
        self.Page_Button_Down(self.Page5_Button, 710)

    def Page4_Button_Clicked(self):
        self.stackedWidget.setCurrentIndex(3)
        self.Page_Button_Down(self.Page1_Button, 150)
        self.Page_Button_Down(self.Page2_Button, 290)
        self.Page_Button_Down(self.Page3_Button, 430)
        self.Page_Button_Rise(self.Page4_Button, 570)
        self.Page_Button_Down(self.Page5_Button, 710)

    def Page5_Button_Clicked(self):
        self.stackedWidget.setCurrentIndex(4)
        self.Page_Button_Down(self.Page1_Button, 150)
        self.Page_Button_Down(self.Page2_Button, 290)
        self.Page_Button_Down(self.Page3_Button, 430)
```

```
        self.Page_Button_Down(self.Page4_Button, 570)
        self.Page_Button_Rise(self.Page5_Button, 710)


if __name__ == '__main__':
    app = QApplication(sys.argv)
    Window = Main_Window()
    Window.show()
    sys.exit(app.exec_())
```