学号____20154369____

密级_____

东北大学本科毕业论文

基于树莓派平台的实时人脸表情识别系统设计与实现

学 院 名 称 ：计算机科学与工程学院

专 业 名 称 ：计算机科学与技术

学 生 姓 名 ：闫润格

指 导 教 师 ：栗伟 副教授

二〇一九年六月

郑 重 声 明

本人呈交的学位论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本学位论文的知识产权归属于培养单位。

本人签名： 日期：

摘 要

随着计算机技术的发展，计算能力的提升和各类数据量的指数增长，近些年来，人工智能在许多研究领域出现了突破性的成果。这些成果进一步推动了更多研究资源的投入和具体应用的落地。在快速发展的图像识别方面，人脸表情识别成为人机交互技术发展的关键突破口之一。对于处理图像有着天然优势的卷积神经网络及其引出的深度学习方法成为各类解决方案的核心，不断有新的方案。除了大规模堆积计算能力训练出更好的模型外，目前，一些较为成熟的模型能够尝试解决实际生活场景中的问题，模型的部署成为了研究过程中亟待解决的问题。

本文提出了一种在树莓派上实现的人脸表情实时识别系统。以 OpenCV 视觉库和 TensorFlow 神经网络框架为核心，基于树莓派 Camera Module V2 完成实时捕获视频输入流，基于 haar 特征和 AdaBoost 级联分类器完成对人脸的识别和切割，并建立两种训练识别模型完成人脸表情的分类（快乐，恐惧，厌恶，中性，生气，惊讶，难过）：一是训练一个简化版的三层卷积神经网络，二是通过迁移学习对 Google Inception v3 的重新训练。基于柱状图和 emoji 表情替代可视化各类表情评分进行展示。本系统可应用于多种人与人、人机交互场景下的人脸表情识别任务。树莓派体积小，能耗低的特点为其部署提供了有利条件。

关键字：表情识别；树莓派卷积神经网络；迁移学习；实时图像；树莓派

ABSTRACT

With the development of computer science and technology, the improvement of computing power and the exponential growth of various types of data in recent years, breakthroughts were made in many research fields of artificial intelligence. These promising results have further promoted investment on research and the application of specific techniques. With the rapid development of image recognition, facial expression recognition(FER) has become one of the key directions among all. Having a natural advantage in image processing, convolutional neural networks(CNN) and deep learning methods were core solutions to such tasks. Large-scale computing devices was constantly stacked together to train a better model, in addition, some mature models are capable of solving problems in reality.

This paper proposed a real-time FER system implemented on Raspberry Pi, with Open Source Computer Vision(OpenCV) library and TensorFlow neural network framwork served as its core. Video is captured in real time using Raspberry Pi Camera Module v2. The face recognition and cutting are done based on the haar feature and the AdaBoost cascade classifier, and two training recognitions are established. The model completes the classification of facial expressions (happiness, fear, disgust, neutrality, anger, surprise, sadness): one is to train a simplified version of the three-layer convolutional neural network, and the other is to retrain the Google Inception v3 through transfer learning. The result of the recognition is diplayed with histogram of scores of 7 categories and an emoji corresponding to the category of the highest score. The proposed system is suitable for FER tasks in a variety of human-human, human-computer interaction scenarios. The credit-card size and low-energy consumption of Raspberry Pi makes it favorable in its deployment.

Keywords: FER; Raspberry Pi; CNN; Transfer learning;

目 录

1 绪论

1.1 课题背景

在计算机科学，心理学及人工智能领域，人脸表情识别都是重要的研究课题，常常被赋予很高的学术和商业潜力。面部情绪作为人类交流中的重要组成部分之一，有利于帮助我们理解他人的意图。通过不同的调查[1]可以看出，在人类沟通时，口语部分传达了全部信息的三分之一，而非语言部分传达了三分之二。在非语言部分中，面部表情是最为重要的信息交换渠道。因此，在过去的几十年中，对面部表情的研究逐渐引起了研究人员的兴趣。

随着近年来 CPU（中央处理器）计算能力大幅提升和 GPU（图像处理器）的强势崛起，呈现指数态势增长的各类数据推动了人工智能技术的快速发展。面部表情识别成为计算机视觉的重要识别任务之一，随着研究的深入，大型人脸表情数据库如雨后春笋般涌现，机器学习，深度学习中相关的解决方案更是百花齐放。人脸表情识别相关研究在计算机视觉与模式识别（CVPR）国际会议中始终占有一席之地，而随着人脸识别技术的成熟，人脸表情识别成为了人工智能发展路上的关键问题。

树莓派作为 2012 年才量产第一版的 ARM 计算机，很快证明了其强大的潜力。在机器人，智能家居，安保安防方面均有突出应用。而随着时间的推移，

1.2 国内外研究现状

各类功能强大的开源深度学习工具如 Theano, Torch, Caffe 等纷至沓来，大量深度学习应用如图像搜索，语音识别，语言翻译等出现在科研和市场中。2014 年，Devries 等人[2]采用卷积神经网络进行面部特征点标注和表情识别（图 1.1）在 ReLU 层后分出两个平行的用于进行表情识别的 Softmax 分类器和用于人脸特征点检测的预测层，并用于自然表情图像实验，结果表明相对于基本卷积网络 63.71%的准确率高出 3 个百分点；2015 年，Lopes[3]等人采用卷积神经网络进行特征学习，并针对图像数据进行数据增强，在 CK 表情库上实验表明卷积特征学习有利于表情分类，同时数据量的增大使得卷积网络性能更好;2016 年，Jung[4]等人把结合卷积特征模型和面部关键点分布模型，通过最终的 Fine-tuning 对两

个接收不同输入的网络进行结果的合并优化，利用微调策略在 CK 库上取得极高的识别率（图 1-2）。通过研究表明，利用深度学习方法可以有效提取分层特征，利用表情图像分类，比传统方法更好。

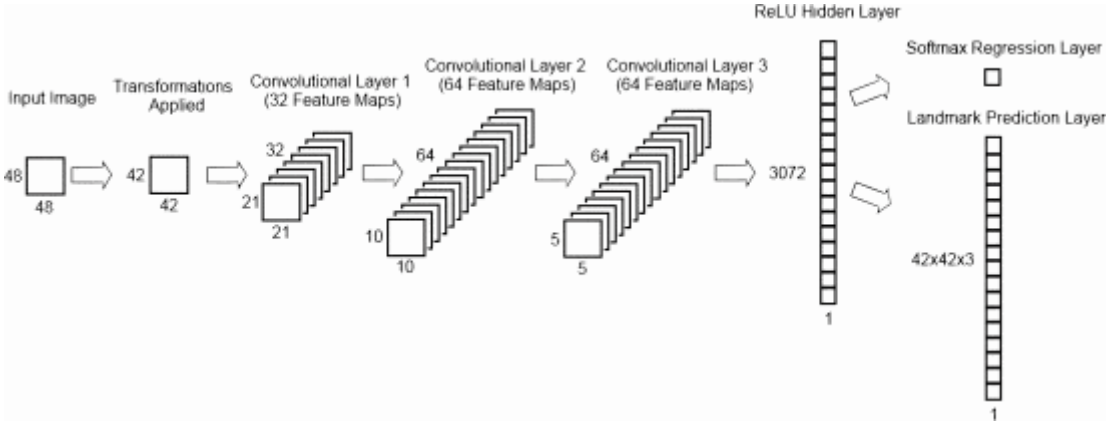


图 1.1 Devries 等人提出的网络结构

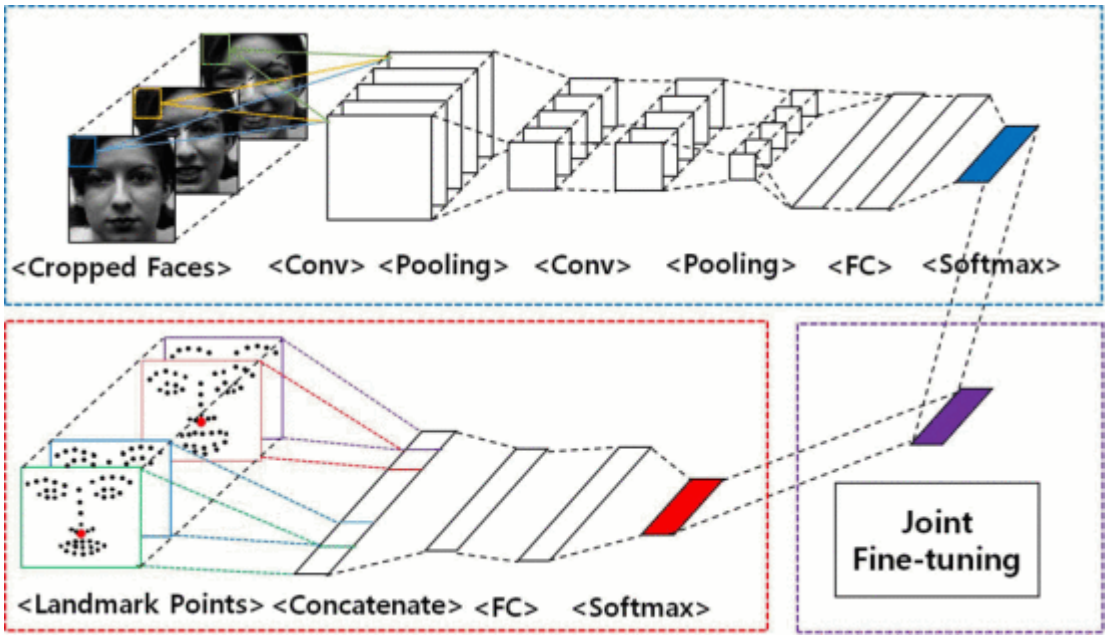


图 1.2 Jung 等人设计的网络结构，

而近几年来，对于人脸表情识别准确率的突破体现在两个方面：一是更加合理的网络结构，二是更加合理的损失函数（Loss Function）。人脸表情本身可以被理解为图像识别任务中人脸这一类别的多个内类（intra class），更好的损失函数能够使得每个内类更加紧密，从而更好地进行内类间的分类。

1.3 本文工作内容

本文针对树莓派软硬件特点，在树莓派上进行深度学习任务的尝试，并进行对所设计系统进行调整优化。

随着树莓派硬件水平逐代提升，其展示出一定计算能力，继而吸引一下研究人员在其上开展人工智能实验。本文在目前已有人脸识别任务成功的基础上，尝试面部表情识别这一复杂度更高的的任务。首先保证识别模型能够在树莓派上流畅运行，其次进行树莓派上模型训练的尝试。

本文着眼于树莓派人脸表情识别的潜在应用，期望验证其在实际生活场景中部署的可能性，旨在探索效率最高，开销最低，精度最高的系统安装和运行过程。

1.4 本文结构安排

本文重点介绍一种基于树莓派的实时表情识别系统的设计和实现。伴随着人脸表情识别技术的快速发展，认识到树莓派体积小，成本低等特点带来的发掘潜力，本文首先通过对已有最新技术的回顾进行可行性分析，并针对软件技术和硬件配置的现状进行较为详细，周全的系统设计。之后依次完成各模块的实现，组合出一个较为稳定的系统，达到了构思之初的主要考量指标，也为未来的继续研究和开发奠定了良好的基础。在完成设计，实现，测试等各个步骤后，开始了本文的撰写。

本文的正文分为六章：

第一章简要介绍了人脸表情识别技术的背景、国内外现状和本论文的创新之处。

第二章叙述了人脸表情识别技术的历史，原理，最新进展和未来发展趋势，介绍了搭载本文实现的系统的硬件环境——树莓派及其附件。

第三章设计了该系统的主要功能并讨论了几个最为重要的技术点。

第四章一步步讲解了系统的具体实现细节并讨论了关键问题的解决方案。

第五章展示了神经网络模型训练过程的数据和最优结果。

第六章总结了本系统的设计和实现过程并展望了本系统未来的发展方向。

2 相关技术综述

本章介绍了人脸表情识别技术和树莓派相关技术，为后续章节中的系统设计与实现做铺垫。在人脸表情识别技术方面，本章介绍了较为成熟的人脸识别技术以及以此为基础的各类表情识别研究。在树莓派相关技术方面，本章介绍了树莓派的发展，软硬件配置并列举了在树莓派上开展高强度计算任务的例子。

2.1 人脸表情识别技术

2.1.1 人脸识别技术

2.1.1.1 人脸识别接口

在人脸检测方面的研究目前已经较为成熟，市面上已出现许多高准确率的人脸检测应用，如 Microsoft Azure 人脸 API（图 2.1），Google Cloud Vision（图 2.2）等。由于使用此类功能类似于调用模型的服务，故大部分开发团队会将这项功能封装在一个轻松调用的 API 中。国内同样出现了许多轻量，识别快，精度高的 API，如旷视 Face++的 Detect API，百度智能云人脸识别 API，腾讯优图的人脸检测与分析 API 等。各类 API 普遍采用 json 数据格式（图 2.3）返回识别结果，方便开发人员调用。以上 API 多数需要在线使用，在一些较差的网络环境下会影响实时面部表情识别的性能。

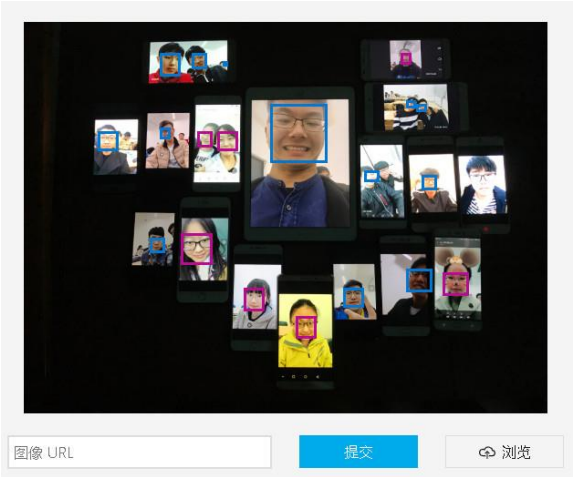


图 2.1 Microsoft Azure 人脸 API 的识别结果样例

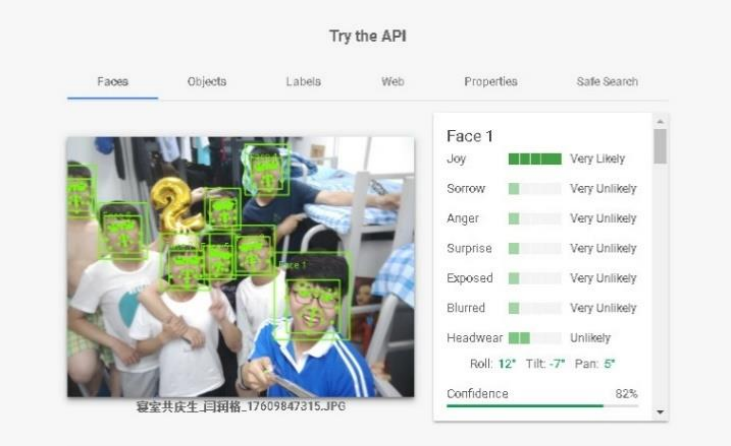


图 2.2 Google Cloud Vision API 的人脸识别，人脸特征点检测和人脸表情识别结果样例

图 2.3 Face++的人脸识别和 json 格式返回内容样例

2.1.1.2 haar 级联分类器

作为计算机视觉库的集大成者，OpenCV 提供了 haar 级联分类器。haar-cascade[5]是一个开展目标识别的机器学习算法，其通过 Paul Viola 提出的特征概念在图像和视频中进行目标识别。haar 特征（图 2.4）与卷积核类似。每个特征都是单一数值，通过黑色矩形下的像素和减去白色矩形下的像素和来获得。haar 特征尺寸较小，一张包含人脸的图片上往往需要进行超过 10^6 次特征计算。为了从中选出最佳特征，我们使用 AdaBoost（Adaptive Boosting）算法来提升效率。

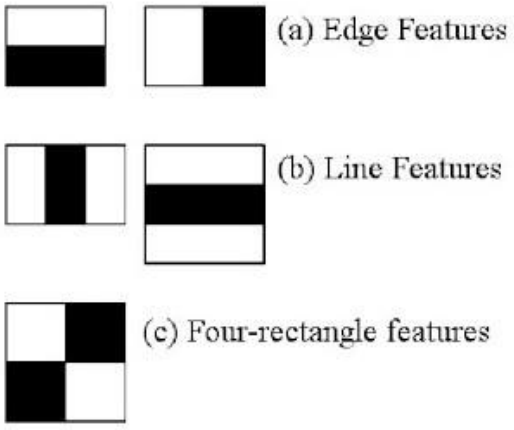


图 2.4 Haar 特征

AdaBoost 的适应性体现在：前一个分类器分类错误的样本会被用来训练下一个分类器。伴随 AdaBoost 使用的分类器可能很弱，但只要其能够做到比随机分类准确，就能够提升最终得到的模型。

$$F_T(\mathrm{x}) = \sum_{t=1}^{T} f_t(x)^{[6]} \qquad (2.1)$$

公式 2.1 中f_t是取x作为输入的弱学习者，返回值指明了目标的类别，绝对值则作为分类的置信度。同样的，T 分类器得到对正类样本得到正向结果，反之得到负向结果。

作为迭代的关键步骤，每一轮中都会新加入一个弱分类器，直到其达到某个预定的足够小的错误率。每个训练样本都有一个权重，表明其被选入训练器的概率（已正确分类——降低概率，未正确分类——提高概率），如此来提高算法对难以分类的样本的关注度。

$$E_t = \sum_i E[F_{t-1}(x_i) + a_t h(x_i)] \qquad (2.2)$$

公式 2.2 中，$F_{t-1}(x_i)$是已经加入到上一阶段训练中的加速分类器，$E(F)$是错误函数，$F_t(x) = a_t h(x)$是作为被考虑加入最终分类器的弱学习者。

在实际训练和分类过程中，一张图片的大部分区域往往都不包含人脸特征，因此，在 haar-cascade 中，级联分类器的概念被引入了。所有 haar 特征被分到不同阶段的分类器中并逐一应用。若第一阶段内没有找到人脸特征，则丢弃整个第一阶段。否则进入第二阶段。若该图像通过所有阶段的 haar 特征，则该图像含有人脸区域。

2.1.1.3 面部动作编码系统（Facial Action Coding System）

人脸识别中最为重要，使用率最高的特征是 AU（Action Units）。文献[7]提出了 Facial Action Coding System，试图使用基于解剖学的 23 个 Action Units 记录描述人脸肌肉运动情况。每一个 Action Units 对应一个或多个面部肌肉的收缩情况。如图所示（图 2.5），AU12 代表的是面部拉动唇角的肌肉，AU8 代表两唇角向中间靠近的肌肉，AU10 是上唇提升的肌肉，AU16 是下唇下压的肌肉，AU 可以单一表示面部表情，也可以组合表示面部表情。

图 2.5 Action Units 及其组合举例

FACS 还提供 AU 及其 AU 变化时间段（表情的发起，表情的完成，表情的结束，序数强度）的检测规则。这是由于面部的肌肉动作随时间变化，而该变化在表情解读中起到重要的作用。中性表情可以被看做是一个无表情的阶段，此时没有任何肌肉活动的迹象。表情的发起阶段，肌肉开始收缩并逐渐增加其强度；表情的完成像是一个高原期，此时肌肉的紧张程度达到最高并保持在较稳定的状态；表情的结束阶段通常伴随着肌肉的放松。各个阶段的顺序一般是中性表情-表情发起-表情完成-表情结束，但同样也有其他顺序组合的出现，如连续多个“表情完成”阶段的组合。作者一开始并没有将此系统用于情绪分析，而仅仅分析肌肉的变化情况。后来此系统被广泛用于表情分析和识别。

2.1.1.4 人脸识别模型比较

目前最常用的大型人脸检测数据集主要有 FDDB，WIDERFace，IJB-A，Caltech 10k Web Faces 等。马萨诸塞大学提出了 Labeled Face in the Wild(LFW)[8]数据集，并定期更新国际上表现出色的模型在 LFW 数据集上的测试结果，表 2.1 列举了其中一部分测试结果。

表 2.1 部分成熟模型在 LFW 数据集上的测试结果

Classifier/Model/API	Accuracy ±Standard Error of the Mean
Simile classifiers	0.8472 ±0.0041
Multiple LE + comp14	0.8445 ±0.0046
Associate-Predict	0.9057 ±0.0056
face.com r2011b	0.9130 ±0.0030
Face++	0.9950 ±0.0036
DeepFace-ensemble	0.9735 ±0.0025
ConvNet-RBM	0.9252 ±0.0038
POOF-gradhist	0.9313 ±0.0040
DeepID	0.9745 ±0.0026
DeepID2	0.9915 ±0.0013
DeepID2+	0.9947 ±0.0012
DeepID3	0.9953 ±0.0010
Uni-Ubi	0.9900 ±0.0032
Baidu	0.9977 ±0.0006
Faceall	0.9967 ±0.0007
Dahua-FaceImage	0.9978 ±0.0007
THU CV-AI Lab	0.9973 ±0.0008
YouTu Lab, Tencent	0.9980 ±0.0023
Yuntu WiseSight	0.9943 ±0.0045
PingAn AI Lab	0.9980 ±0.0016
Turing123	0.9940 ±0.0040
VisionLabs V2.0	0.9978 ±0.0007
yunshitu	0.9987 ±0.0012

续表 2.1 部分成熟模型在 LFW 数据集上的测试结果

Classifier/Model/API	Accuracy ±Standard Error of the Mean
yunshitu	0.9987 ±0.0012
Meiya Pico	0.9972 ±0.0008
CHTFace	0.9960 ±0.0025
YI+AI	0.9983 ±0.0024
IFLYTEK-CV	0.9980 ±0.0024
DRD, CTBC Bank	0.9981 ±0.0033

2.2 面部表情识别技术

2.2.1 传统识别方法

传统识别方法主要依托图像获取、人脸检测、特征提取、特征分类四部分组成[9]。其中图像获取是指从摄像头等视频输入设备获取视频或图像输入或是直接从文件读取；人脸检测是指根据图片特征对图片中的人脸有无进行判断，若有则对其坐标位置进行标记；特征提取是指对于能够用来判断图像的属性，如颜色的分布，像素值的大小等进行记录；特征分类是指根据特征的各类要素的差异对特征进行区分。

2.2.1.1 图像获取

图像获取主要通过实时采集和文件读取的形式开展。实时采集取决于后续三个步骤的完成速度，若最终结果反馈速度明显慢于视频输入速度，则系统的实时性就会大打折扣。因此实时图像采集往往配合识别速度较快的模型使用。图像文件读取分为全表情图片读取和随机图片读取两种情况。若训练集/测试集中有不含人脸（表情）的图片，则需要对人脸检测模块设定阈值，否则则不用。视频文件读取时，同样需要后续模块较快的反馈速度，否则将大幅降低识别速度，同时占用更多内存资源。

2.2.1.2 特征提取

人脸表情特征提取主要为了去除非表情噪声，将最主要的特征用于训练，因此也起到了降低图像维数的作用。人脸表情的产生十分复杂，直观体现为肌肉运动及其带来的面部纹理，轮廓等形体变化。而对于表情的记录，静态图像和动态图像又有不同的特点。因此，对于表情的特征提取主要有基于静态图像的特征提取方法和基于动态图像的特征提取方法。其中基于静态图像的特征提取方法主要有整体法和局部法，而基于动态图像的特征提取算法主要有光流法、模型法和几何法等。

整体法：

较为经典的算法有主元分析法（PCA），线性判别分析法（LDA）和独立分量分析法（ICA）。近年来提出的 FastICA[10-12]算法能够快速分离表情特征，SVDA 算法[13]能够有效提升表情的类间分离，还免去了 SVM（支持向量机）算法所需要的决策函数，目前其表情识别率已超过 PCA 和 LDA。Tsalakanidou 等人[14]借助二维离散余弦变化，将人脸图像映射到频域空间，再结合神经网络实现表情特征的分类。

局部法：

局部法着眼于面部肌肉的褶皱，纹理的细微变化，通过多个局部属性组合出对整体表情的判断。LBP（Local Binary Pattern）算子法和 Gabor 小波法是较为经典的局部算法，文献[15]结合 LBP 特征法和线性规划方法，在 JAFFE 数据集上实现了 93.8%的准确率。基于 LBP 提出的 CBP[16]（Central Binary Pattern）充分考虑了中心像素的作用，计算速度得到了明显提升，LDP[17]（Local Directional Pattern）则胜在其对噪声不敏感的带来的靠噪能力和梯度信息带来的稳定性。

光流法：

光流法是早期动态图像人脸表情识别算法中主要采取的特征提取方法，其主要作用是反映不同帧之间目标物体的灰度变化，能够突出人脸形变，较好的体现人脸运动趋势。Yacoob[18]利用光流场和梯度场表示每一帧的时空变化，记录表情变化带来的面部肌肉刚性和非刚性的变化方向，进而通过肌肉的变化对表情进行分类。

模型法：

模型法主要将动态图像中的表情信息以参数的形式进行描述，针对模型法的研究往往兼顾动态图像中的人脸二维和三维建模。较为常用的算法有主动形状模型法（ASM）和主动外观模型法（AAM）。ASM 主要反映图像的局部纹理信息，而 AAM 主要反映图像的全局纹理信息，两种算法都由形状模型和主观模型两部分组成。文献[19]提出了一种基于 ASM 的三维人脸特征跟踪方法，完成了对人脸 81 个特征点跟踪建模并实现了对部分复合动作单元的识别。文献[20]结合二维表现特征和三维形状特征，基于 AAM 算法扩展出一种基于视图实现表情识别的方法，在人脸位置发生偏移的情况下，实现了表情特征的提取。

几何法：

由于表情的产生和表达主要是通过面部器官及其周围的变化来反映的，在对于动态图像的分析中，研究者更加重视人脸的主要器官和面部褶皱部分的特征提取。几何法即在面部器官区域标记特征点，进而计算特征点间距离和特征点所在曲线曲率的方法。Irene Kotsia[21]使用 Candide 网格节点记录人脸表情，通过记录动态图像第一帧与该图像序列中最大表情强度帧之间的几何位移来表达特征，从而实现对表情的识别。

2.2.1.3 特征分类

特征分类的目标是将提取出的特征和表情类别对应起来，其主要方法有基于贝叶斯网络的分类方法和基于距离度量的分类方法。在人脸表情识别中，表情类别主要分为基本表情和动作单元（AU）。

基于贝叶斯网络的分类方法：

贝叶斯网络来源于贝叶斯公式，是一种基于概率推理的图形化网络。在人脸识别方面，概率推理即从已知表情信息中推断出位置表情的概率信息的过程。目前已发展出多种贝叶斯网络扩展出的分类算法和隐马尔科夫模型（Hidden Markov Model）算法等。Ira Cohen[22]等人在面部表情分类任务上测试了不同的贝叶斯网络分类器，特别使用了树增强器（TAN）来学习不同面部动作特征间的依赖，还提出了一种新的 HMM 架构。

基于距离度量的分类方法：

距离度量是最基础的分类方法之一，其通过计算样本间距来实现表情分类。最常见的算法有 K-近邻法和支持向量机（Support Vector Machine）算法。K-近邻法通过未知样本 x 和所有已知类别样本间的欧氏距离远近来进行决策；支持向量机算法则首先优化目标函数，通过寻找使得不同类别样本间距最大的分类决策超平面来实现较为准确的分类。K-近邻法由于过度依赖待分类样本的数量[23]，往往作为图像分类任务中的入门方法。在 SVM 的基础上，文献[24, 25]各自提出了改进方案：前者通过对 K-近邻法和支持向量机的结合，把近邻信息添加到支持向量机的构建中，完成了一种局部 SVM 分类器；后者通过对 SVM 和树型模块的结合，完成了 CSVMT 模型，有效降低了不同层次上分类子问题的复杂度。

2.2.2 深度学习方法

2.2.2.1 卷积神经网络简介

卷积神经网络是一种特殊的深度学习模型，有着较好的图像特征提取的能力。输入数据通过卷积层中的滤波器进行卷积从而产生特征图，再将特征图组合形成全链接网络进行非线性分类。此处的全连接层类似于多层感知机（Multi-Layer Perceptron），区别在于 dropout 等机制的加入，减弱了训练中出现过拟合的情况，而 softmax 则把守着 CNN 的最后一关，得出各个样本的得分分布。

自 Fukishima，LeCun 等人提出，完善卷积神经网络（Convolution neural networks）以来，不断有研究人员加入新的网络层，提出新的网络结构。伴随着 AlexNet 在 2012 年 ILSVRE（ImageNet Large Scale Visual Recognition Competition）上的惊艳表现，DCNN 时代到来了。后续几年参加比赛的 Inception，VGG，ResNet 等网络模型经过大型数据库 ImageNet 上的训练，展现出了越来越强的图像识别能力，成为了目标识别，图像分类任务中最为流行的网络模型。深度学习除预测以外，还完成了特征工程的任务，省去了人工提取特征的繁琐而能够更好地结合 AU（Action Units）进行表情的分类。

2.2.3.2 深度学习进展

在深度学习方面，除了卷积神经网络加深，网络结构的变化，新网络层的出现，传统方法与神经网络的结合也推动面部表情识别的准确率连年上升。Z. Yu[23]

等人首先糅合目前准确度最高的三个人脸检测器，并在其检测基础上对多个深度卷积神经网络的结果进行综合评估（公式 2.3）。他们首先在 fer2013 数据集上完成预训练，接着在 SFEW 数据集上进行微调（Fine-tuning）而达到更高准确率。

$$\mathcal{L} = -\sum_{i=1}^{N} logP(y_i|x_i) \qquad (2.3)$$

其中，N 是训练样本数，x_i是第 i 个训练样本，y_i是x_i的标签，$P(y|x_i)$是输入x_i时神经网络对第 y 个类别给出的相应。

2010 年以来，对人脸表情识别的研究受到广泛关注，研究人数增多，研究方向变宽，数据集也向着更大型，更细分类的方向发展。文献[26]全面而彻底地回顾了人脸表情的识别过程（图 2.6），近年来数据集的发展（图 2.7），各类识别技术的优劣，还关注了以上各点的相关问题。

随着研究的深入，在已有的用于图像识别的优秀神经网络结构支持下，更多专门用来识别表情的网络被设计出来。他们针对表情识别类似于内类（intra class）识别这一特点，对网络中各个细节进行了改进和优化，从而在这一特定任务上获得更高的准确率。

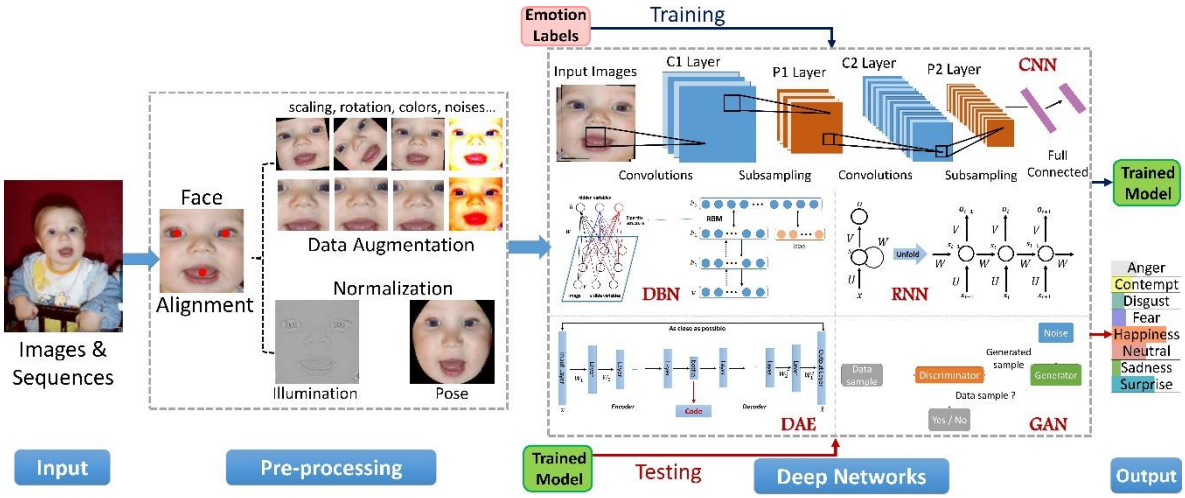


图 2.6 人脸表情识别全过程

TABLE 1
An overview of the facial expression datasets. P = posed; S = spontaneous; Condit. = Collection condition; Elicit. = Elicitation method.

Database	Samples	Subject	Condit.	Elicit.	Expression distribution	Access
CK+ [33]	593 image sequences	123	Lab	P & S	6 basic expressions plus contempt and neutral	http://www.consortium.ri.cmu.edu/ckagree/
MMI [34], [35]	740 images and 2,900 videos	25	Lab	P	6 basic expressions plus neutral	https://mmifacedb.eu/
JAFFE [36]	213 images	10	Lab	P	6 basic expressions plus neutral	http://www.kasrl.org/jaffe.html
TFD [37]	112,234 images	N/A	Lab	P	6 basic expressions plus neutral	josh@mplab.ucsd.edu
FER-2013 [21]	35,887 images	N/A	Web	P & S	6 basic expressions plus neutral	https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge
AFEW 7.0 [24]	1,809 videos	N/A	Movie	P & S	6 basic expressions plus neutral	https://sites.google.com/site/emotiwchallenge/
SFEW 2.0 [22]	1,766 images	N/A	Movie	P & S	6 basic expressions plus neutral	https://cs.anu.edu.au/few/emotiw2015.html
Multi-PIE [38]	755,370 images	337	Lab	P	Smile, surprised, squint, disgust, scream and neutral	http://www.flintbox.com/public/project/4742/
BU-3DFE [39]	2,500 images	100	Lab	P	6 basic expressions plus neutral	http://www.cs.binghamton.edu/~lijun/Research/3DFE/3DFE_Analysis.html
Oulu-CASIA [40]	2,880 image sequences	80	Lab	P	6 basic expressions	http://www.cse.oulu.fi/CMV/Downloads/Oulu-CASIA
RaFD [41]	1,608 images	67	Lab	P	6 basic expressions plus contempt and neutral	http://www.socsci.ru.nl:8180/RaFD2/RaFD
KDEF [42]	4,900 images	70	Lab	P	6 basic expressions plus neutral	http://www.emotionlab.se/kdef/
EmotioNet [43]	1,000,000 images	N/A	Web	P & S	23 basic expressions or compound expressions	http://cbcsl.ece.ohio-state.edu/dbform_emotionet.html
RAF-DB [44], [45]	29672 images	N/A	Web	P & S	6 basic expressions plus neutral and 12 compound expressions	http://www.whdeng.cn/RAF/model1.html
AffectNet [46]	450,000 images (labeled)	N/A	Web	P & S	6 basic expressions plus neutral	http://mohammadmahoor.com/databases-codes/
ExpW [47]	91,793 images	N/A	Web	P & S	6 basic expressions plus neutral	http://mmlab.ie.cuhk.edu.hk/projects/socialrelation/index.html

图 2.7 人脸表情识别主流数据集

TABLE 6
Comparison of different types of methods for static images in terms of data size requirement, variations* (head pose, illumination, occlusion and other environment factors), identity bias, computational efficiency, accuracy, and difficulty on network training.

Network type	data	variations*	identity bias	efficiency	accuracy	difficulty
Pre-train & Fine-tune	low	fair	vulnerable	high	fair	easy
Diverse input	low	good	vulnerable	low	fair	easy
Auxiliary layers	varies	good	varies	varies	good	varies
Network ensemble	low	good	fair	low	good	medium
Multitask network	high	varies	good	fair	varies	hard
Cascaded network	fair	good	fair	fair	fair	medium
GAN	fair	good	good	fair	good	hard

图 2.8 各类人脸表情识别技术的优劣

2.3 树莓派及其相关技术介绍

树莓派(Raspberry Pi)是一系列小型单板计算机（最新版如图 2.9），由英国树莓派基金会推动开发。该项目最初的目的是促进计算机基础知识在中小学校内和发展中国家的推广，但最终产品和平台都比预期更受欢迎。树莓派各类型产品及其周边配件吸引了大量的程序开发人员和无编程知识的业余爱好者[27]。

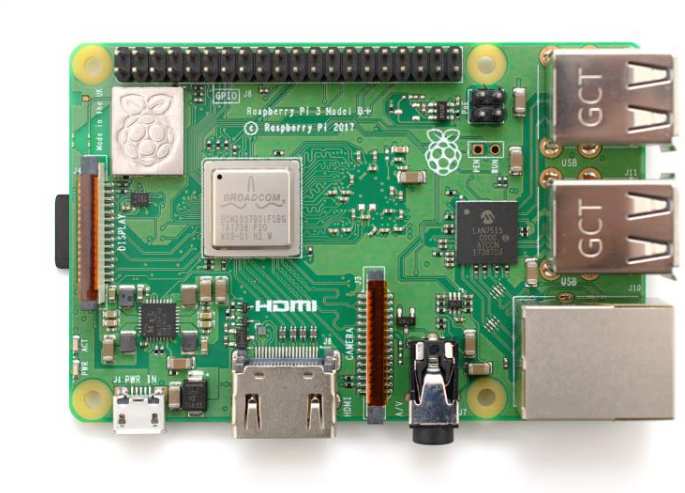


图 2.9 树莓派 3 Model B+（2017）实物图

截至 2015 年 2 月，树莓派基金会已售出超过五百万台树莓派，这也使得树莓派成为有史以来最畅销的在英国设计并制造的电脑。近年来用户对树莓派的需求强劲增长，截至 2018 年 3 月，树莓派的销售量已达 1900 万。

树莓派已发布数代产品（表 2.2）。所有型号的产品都具有博通（Broadcom）片上系统（SoC）

表 2.2 树莓派模型发布情况

家族	型号	构成	以太网	无线网	发布时间	是否停产
RPi 1	B	标准型	有	无	2012	是
	A	(85.60 × 56.5 mm)	无		2013	是
	B+		有		2014	
	A+	紧凑型	无		2014	
		(65 × 56.5 mm)				
RPi 2	B	标准型	有	无	2015	

RPi Zero	Zero W/WH	零型 (65 × 30 mm)	无	无 有	2015 2017
RPi 3	B	标准型	有	有	2016
	A+	紧凑型	无		2018
	B+	标准型	有		2018

2.3.1 树莓派软硬件介绍

2.3.1.2 树莓派硬件介绍

树莓派的硬件经历了数个版本的进化，其中主要的改进体现在主存容量和外围设备的支持。图 2.10 展示了 Model B 和 Model B+的架构；Model A，Model A+和 Pi Zero 的架构与之类似，仅从其中除去以太网和 USB 组件。其中以太网适配器直接内联到一个 USB 端口。而 Model A，Model A+和 Pi Zero 的 USB 端口直接连接到片商稀土。在 Model 1B+及之后的树莓派模型上，USB/以太网端口提供了一个五端口集线器，其中四个是可用的。而 Model 1B 仅提供两个可用 USB 端口。在 Pi Zero 上，一个微型 USB 端口直连到片上系统。

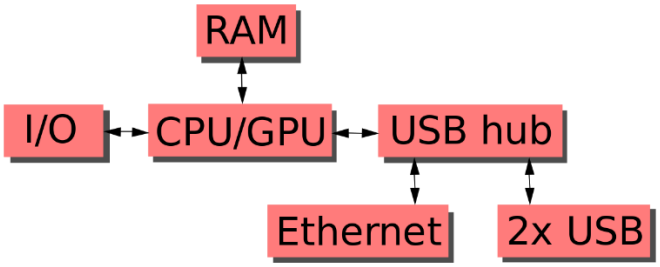


图 2.10 树莓派 Model B/Model B+的架构示意图

处理器：

第一代树莓派上所使用的博通 BCM2835 片上系统包含了一个主频为 700MHz 的 ARM11 76JZF-S 处理器，一个 VideoCore IV 型图形处理器和主存。其拥有 16kB 大小的一级缓存和 128kB 大小的二级缓存，其中二级缓存主要供图形处理器使用。从实物图上看，片上系统被覆盖在主存芯片下，所以肉眼仅能看到其边缘。

树莓派使用的处理器与初代 Iphone 手机的处理器相同，其时钟频率更高，因此也能与速度更快的图形处理器配合。

第二代树莓派早期模型使用主频为 900MHz 的 64 位 4 核 ARM Cortex-A7 处理器，其拥有 256kB 的二级高速缓存。改进版使用与树莓派 3B+相同的处理器——主频为 1.2GHz 的 64 位 4 核 ARM Cortex-A53 处理器，并对其做了降频处理，依然保持主频为 900MHz。

第三代树莓派（Model 3B+）使用博通 BCM2837B0 片上系统，主频提升至 1.4GHz，并将二级高缓提升到了 512kB 大小。

树莓派 Zero 和 ZeroW 依然使用第一代的片上系统，并把主频提升到了 1GHz。

性能：

第一代树莓派在 700MHz 运行时，其真实性能约为 0.041GFLOPS，在 CPU 层面上，其性能与奔腾二代处理器（1997-1999）相当。GPU 真实性能约为 1Gpixels，与 2001 版 Xbox 的图形计算能力相当。

第二代树莓派的 CPU 性能提升了约 4-6 倍，GPU 性能保持不变。在并行化基准测试中，第二代树莓派比第一代快 14 倍

第三代树莓派的性能较第一代提升了 10 倍，这一提升主要依赖于任务线程和指令集的使用。基准测试表明，在并行任务中，树莓派 3 比树莓派 2 快了 80%。

主存：

在旧版 beta Model B 上，128MB 默认被 GPU 调用，另外 128MB 留给 CPU。最新版本的 Model B 拥有 512MB 主存容量，此时新的标准主存分割文件被应用。16 至 256MB 内存可被 GPU 动态调用。树莓派第二代和第三代均拥有 1GB 主存容量，树莓派 Zero 和 Zero W 拥有 512MB 主存。

网络：

Model A, A+和 Zero 不包含以太网电路，一般通过用户外部 USB 以太网或 WiFi 适配器连接到网络。在 Model B 和 Model B+上，SMSC LAN9514 芯片为树莓

派提供内置 USB 以太网适配器。树莓派第三代和 Zero W 配有博通 BCM43438 FullMAC 芯片，从而提供 2.4GHz 的 WiFi（802.11n）和蓝牙 4.1.。树莓派 3 还有一个 10/100 Mbit/s 的以太网端口。最新版的树莓派 3 Model B+拥有双频带 IEEE 802.11b/g/n/ac/ WiFi, 蓝牙 4.2 和千兆以太网。

视频：

树莓派的视频控制器能够生成标准现代电视分辨率，如高清和全高清，以及更高或更低的显示器分辨率，以及较旧的 NTSC 或 PAL 标准 CRT 电视分辨率。

尽管树莓派 3 没有 H.265 视频解码硬件，但由于其更强大的 CPU，可以把这项工作放在软件站完成。相比于之前产品中 GPU250MHz 的运行频率，树莓派 3 的 GPU 运行在更高的 300-400MHz 区间。

通用输入输出（GPIO）连接器：

树莓派 1 Model A+和 B+，树莓派 2 Model B，树莓派 3 Model A+,B 和 B+，树莓派 Zero 和 Zero W 的 GPIO 共有 40 个引脚，树莓派 1 Models A 和 B 仅有前 26 个引脚（图 3.3）。

Model B 第二代还拥有一个含 8 个引脚的焊盘（在电路板上称为 P5，在原理图上称为 P6），可以访问额外的 4 个 GPIO 连接。各引脚功能如图 3.4 所示。

Model A 和 B 使用 GPIO 16 提供对 ACT 状态 LED 的 GPIO 访问。Model A +和 B +使用 GPIO 47 提供对 ACT 状态 LED 的 GPIO 访问，使用 GPIO 35 提供电源状态 LED。

配件：

Gertboard – 树莓派基金会认可设备，是专为教育目的而设计的。其扩展了树莓派的 GPIO 引脚，允许与 LED，开关，模拟信号，传感器和其他设备进行接口和控制。它还包括一个可选的 Arduino 兼容控制器，用于与树莓派连接。

相机 – 2013 年 5 月 14 日，基金会和分销商 RS Components＆Premier Farnell / Element 14 推出了树莓派相机以及固件更新以使其更好地和树莓派连接。相机板出厂时带有柔性扁平电缆，可插入位于以太网和 HDMI 端口之间的 CSI

连接器。在 Raspbian 操作系统中，用户必须通过运行 Raspi-config 并选择摄像头选项来启用摄像头。相机模块在欧洲的售价为 20 欧元（2013 年 9 月 9 日）。它可以提供 1080p，720p 和 640x480p 视频。相机的尺寸为 25 mm×20 mm×9 mm。2016 年 5 月，camera Module V2 面世（图 2.11），该摄像头的像素提升到了 800 万。

图 2.11 树莓派 Camera Module V2 与树莓派 3 Model B+连接示意图[28]

红外摄像机 - 2013 年 10 月，树莓派基金会宣布他们将开始生产一款没有红外滤光片的相机模块，称为 Pi NoIR。

官方显示屏 - 2015 年 9 月 8 日，树莓派基金会和经销商 RS Components& Premier Farnell / Element 14 推出了 Raspberry Pi Touch Display.

HAT（硬件连接在顶部）扩展板 -受 Arduino 屏蔽板的启发， Raspberry Pi Foundation 设计了 HAT 板的接口，与模型 B+配合。每个 HAT 板载一个小 EEPROM（通常为 CAT24C32WI-GT3），其包含了板的相关细节，从而使得树莓派的操作系统能够从 HAT 获取信息和技术细节。具体信息传递与树莓派所使用的操作系统有关。

2.3.2.2 树莓派软件介绍

操作系统：

树莓派基金会提供了 Raspbian，一个基于 Debian 的 Linux 发行版，同时还有第三方 Ubuntu，Windows 10 IoT Core， RISE OS 和专门的媒体中心发行版。树莓派使用 Python 和 Scratch 作为主要编程语言，同时支持其他多种语言。默认固件是闭源的，与此同时，非官方的开源固件也可被使用。

其他可用操作系统有：

基于 Linux：Android Things, Arch Linux ARM, openSUSE, SUSE Linux Enterprise Server 12 SP2/SP3, Gentoo Linux, Lubuntu, Xubuntu, Devuan, CentOS(树莓派 2 及更高版本), RedSleeve（树莓派 1）, Slackware ARM, Kali Linux, SolydXK, Ark OS, Sailfish OS（树莓派 2）, Tiny Core Linux, Alpine Linux, Void Linux, Fedora（树莓派 2 及更高版本）等。

非 Linux：Broadcom VCOS, FreeBSD, NetBSD, OpenBSD, Plan 9 from Bell labs and Inferno, Haiku, HelenOS 等。

驱动程序用户接口：

树莓派通过二进制 blob 文件和最初是封闭源的附加软件使用 VideoCore IV GPU，blob 文件在树莓派从 SD 卡启动时加载至 GPU 中。附加软件的代码也在之后被释出。然而，大多数实际的驱动器工作都是通过闭源 GPU 代码完成的。应用软件调用闭源运行时库（OpenMax, OpenGL ES 或 OpenVG），之后这些胡调用 Linux 内核中的开源驱动器，再通过此驱动器调用闭源的 VideoCore IV GPU 代码。内核驱动器的用户接口是为这些闭源库专门设计的。在树莓派上，视频应用使用 OpenMax 库，3D 应用使用 OpenGL ES 库，2D 应用使用 OpenVG 库，而这后两者轮流使用 EGL 库。OpenMax 库和 EGL 库房轮流使用开源内核驱动器。

固件：

官方固件是闭源的，可自由再发行的二进制 blob 文件。官方还提供了最小概念验证的开源固件，主要用于初始化和启动 ARM 内核以及执行 ARM 端所需的最小化启动。该固件同样能够启动一个很小的 Linux 内核，并提供补丁来消除其对邮箱接口的响应性以来。该固件在树莓派 1，2，3 上均可运行。

软件开发工具：

作为致力于提升学生和大众编程能力的微型电脑，树莓派提供多种语言的开发环境和容易上手的教程。主要工具有：

Arduino IDE-为 Arduino 开发板编程。

BlueJ-面向初学者教授 Java 语言。

Julia-互动式跨平台编程语言/环境。

Scratch-一个跨平台教育类型交互开发环境，使用像乐高积木一样的可视化块堆叠在一起进行编程。Scratch 最初由 MIT 的 Life Long Kindergarten group 开发。树莓派版本的 Scratch 针对其有限的计算资源经行了大量优化，并在 Squeak Smalltalk 系统中进行实现。

TensorFlow-谷歌开发的人工智能框架。树莓派基金会与谷歌合作，通过预先安装的二进制文件简化了安装流程。TensorFlow 也是本论文中最为重要的工具之一。

2.3.2 深度学习可行性分析

2.3.2.1 处理复杂任务

随着树莓派的更新换代，其硬件逐步得到优化，随着而来就是计算能力的存储能力的大幅提升。

自树莓派被发布以来，基于树莓派的应用百花齐放。树莓派功耗低，体积小的特点使其在智能家居等一些非工业化场景下大显身手。特别是其强大的图形处理能力和丰富的多媒体接口使其受到了广泛使用。小到门禁打卡系统，大到高速公路大型 LED 灯牌，都能见到树莓派的身影。

树莓派从面世以来，以其独特之处吸引了大批爱好者尝试在其上进行大规模的运算任务。他们通过大量树莓派单板的堆叠搭建运算丛集[29]：2012 年，博伊西州立大学的博士生 Joshua Kiepert 使用 32 个树莓派组成一个成本约为 1500 美元的运算丛集。在 2013 年，英国南安普顿大学的一名工程师，使用 64 个树莓派

和乐高积木，打造出超级计算机 Iridis-Pi，成本约为 4000 美元美元。在 2017 年，美国洛斯阿拉莫斯国家实验室更是设计了由 750 个树莓派组成的超级计算机，令人看到了树莓派的巨大潜力。该超算的运算能力直接进入全球百大超级计算机前 100 名。

2.3.2.2 处理人工智能任务

而随着近几年人工智能，特别是深度学习的研究进展，已经有不少业余爱好者尝试用神经网络来考验树莓派的性能。在一些能够快速搭建的应用中，树莓派有着不错的表现。在笔记本电脑，台式电脑或其他具有较为强大的计算能力的平台开展神经网络的训练，将训练好的网络模型安装在树莓派上，此时，树莓派展现出了流畅运行小型网络模型的能力。如谷歌 AIY（Do-it-yourself Artificial Intelligence）项目中的 Vision Kit[30](图 2.12, 图 2.13)和 Voice Kit， 都是以树莓派 Zero W 为主板，通过简单的外设安装（小喇叭，摄像头，显示屏）激发了树莓派在人工智能方面的潜力。目前 Vision Kit 上能够成熟运行的模型有人脸检测，狗/猫/人检测，菜品分类，图片分类等。这些较为基础的应用不会让树莓派 CPU 过热或是直接死机，树莓派已经拥有能够运行对应神经网络模型的性能。

图 2.12 谷歌 AIY Vision Kit 套件实物图-外部

图 2.13 谷歌 AIY Vision Kit 套件实物图-内部

可是即便如此，在人工智能各领域开展研究的过程中，无一不是通过对大规模数据集的学习得出一个较为可靠的模型，再将真实生活中的数据投入模型中进行分析和预测。虽然树莓派具有处理复杂任务的潜力，但目前最新版的树莓派的 CPU，GPU，RAM 与一般笔记本电脑仍有较大差距。由于其有限的 RAM 大小和处理器速度树莓派的硬件并不适合处理繁重的计算型线程，阻碍了树莓派在训练模型方面更进一步的应用。

表 2.3 树莓派 3 Model B+与一般笔记本电脑的硬件对比

硬件	Raspberry Pi 3 Model B+	HP 256 G3 Notebook PC
CPU	4*ARM Cortex-A7 @ 1.40GHz	4*Intel(R) Core™ i5-4210U CPU @ 1.70GHz
L1 Cache	32KB	128KB
L2 Cache	512KB	512KB
GPU	Broadcom VideoCore IV @ 250MHz	NVIDIA GeForce 820M @ 775MHz
RAM	1.00GB	4.00GB

3 系统设计

本章根据各类技术的最新发展，考虑树莓派的性能对系统的影响及流畅度的要求，对系统进行设计。在本章中，系统被分为多个模块逐一进行设计，在每一小节中对各模块进行具体说明。

3.1 系统总体结构设计

分类模型作为系统的核心部分，需要通过训练得到，具体设计步骤如图 3.1：

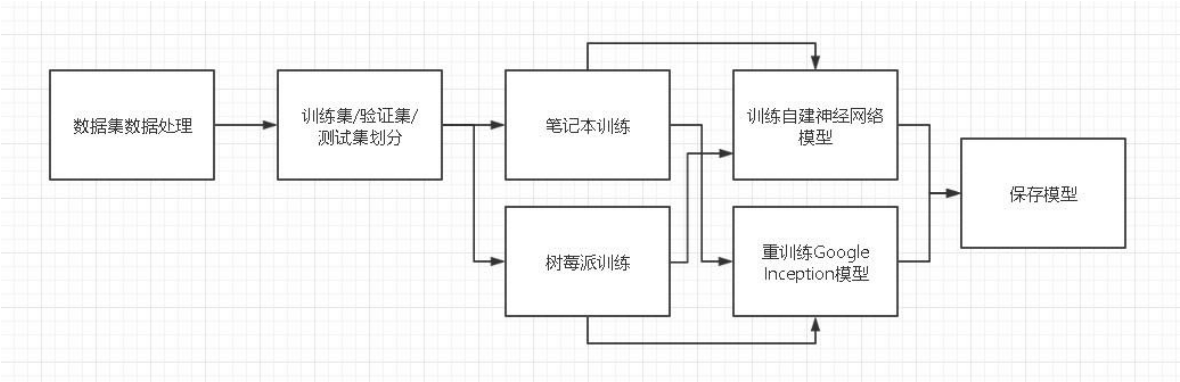


图 3.1 模型训练流程图

系统各个组成部分如图 3.2 所示，运行时顺序如下：

实时图像获取->检测人脸并切割图像->图像预处理->神经网络模型预测->结果展示

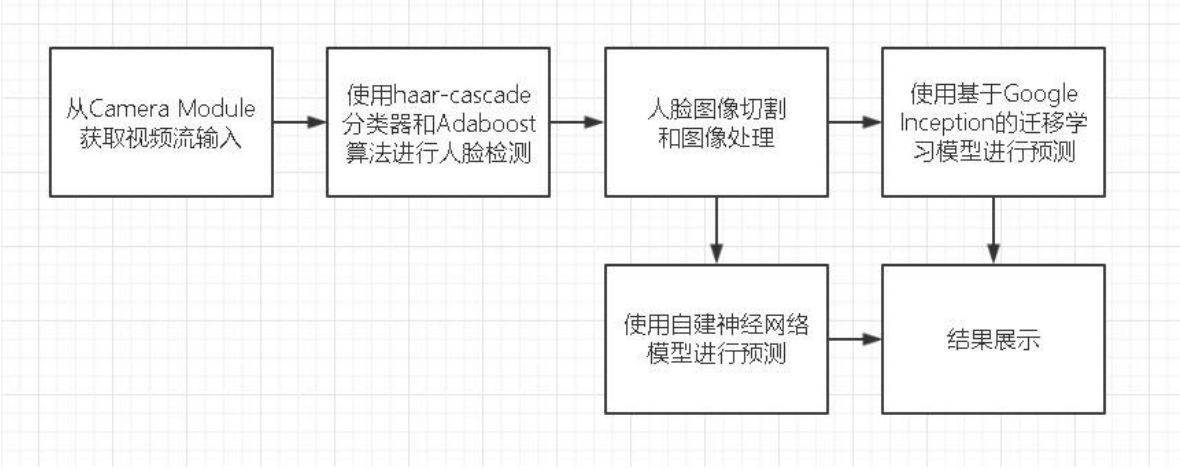


图 3.2 系统框图，各框均代表系统的一个模块

本系统中的关键技术点如下：

①数据集规模应最够大，其最主要的变量应符匹配本系统所识别的差异，数据应规整无坏值。

②数据集中图片应被进行合理而不过分的处理，使其最大程度保留表情属性，同时不会对神经网络造成太大负担。

③确定数据集的划分，保证数据得到充分的训练而无明显过拟合现象，保证数据被充分利用，提供最多样的特征。

④确保网络结构简洁无冗余，不会浪费计算资源，每层网络各司其职。

⑤训练过程中的数据应记录并可视化，确保整体训练次数足够多而各层的表现都足够好。

⑥人脸识别频率不应该过大或过小，以免造成后续表情漏判和误判。

⑦人脸切割不应过大或过小，以免增加无效特征或忽略重要特征。

⑧结果展示应直观而显著，从而让用户直接体验到识别的速度和准确率。

3.2 卷积神经网络设计

本文设计了一个卷积神经网络（图 3.3），其中包含两个卷积层，两个全连接层和一个线性层。每个卷积层除卷积操作外还有偏置操作，标准化操作和池化操作。整个网络的构建基于 tensorflow 框架实现，针对每个参数和细节进行微调。

3.3 其他部分设计

3.3.1 数据处理

fer2013 数据集中默认使用自然数 0-6 作为图片标签，在本系统中，由于离散数字不具有任何含义，故将标签编码形式改为 one-hot 编码，即通过一个形状为[7]，仅含有 0，1 的向量来实现。对于数据集，还应分批将（数据样本，数据标签）进行打乱。流程图如图 3.4。

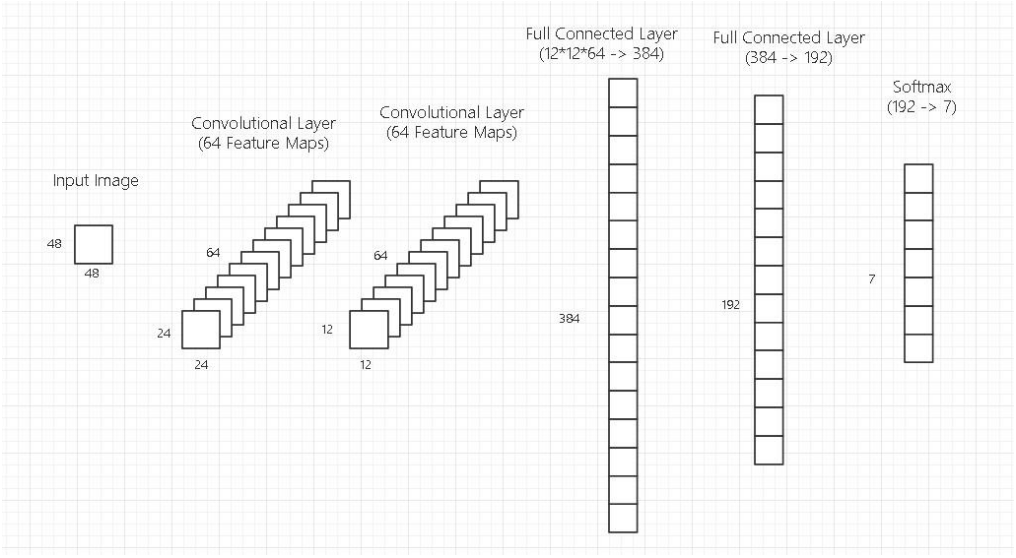


图 3.3 网络结构设计

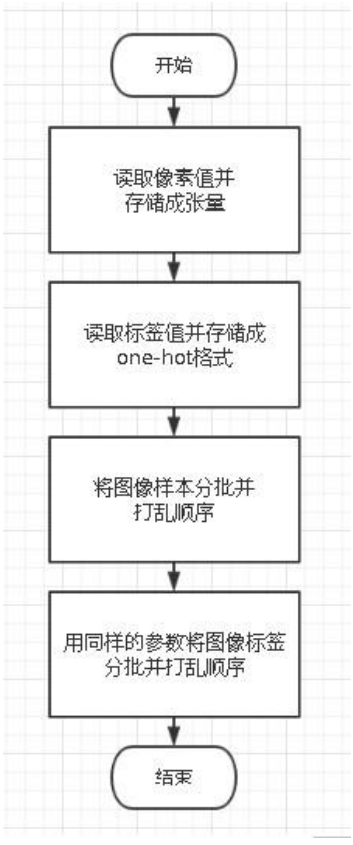


图 3.4 数据集操作流程图

3.3.2 结果展示

本文计划在视频输入直接作为用户界面背景，在其上进行人脸切割（以框圈出），表情识别的结果展示（如图 3.5）。

本文计划在用户界面左上角以动态柱状图的形式展示各分类识别得分情况。使用 OpenCV 中的文字处理函数对表情标签进行竖向输出，在每一标签右侧以矩形函数画出彩色矩形作为柱状图，其中矩形的宽度不变，长度与每一次面部表情的识别得分成正比，从而使得矩形面积与识别得分成正比。

同时，本文计划在用户界面右下角采用动态 emoji 替代的形式对表情识别结果进行直观显示。为保证 emoji 清晰显示，首先将 emoji 下覆盖的 frame 界面以 emoji 的像素反色显示，再在其上进行 emoji 的覆盖，使得显示的 emoji 始终对应识别得分最高的表情。

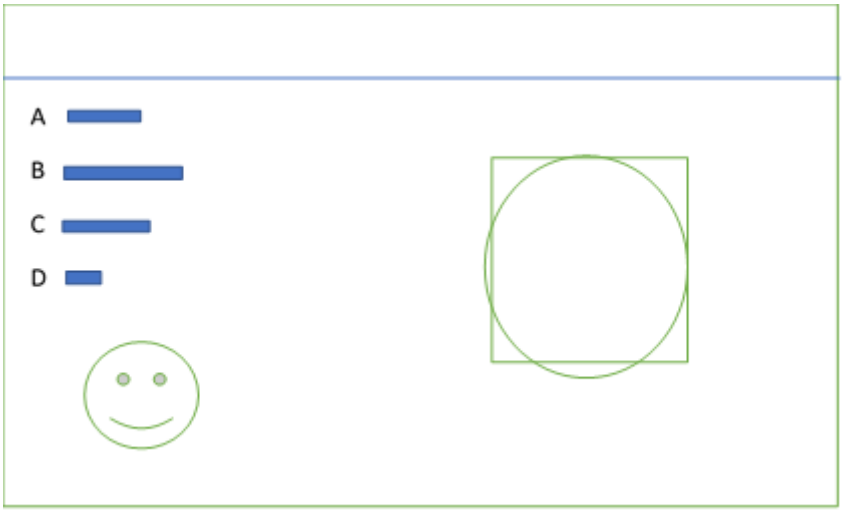


图 3.4 用户界面设计图

4 系统实现

针对设计过程中的细节，本章主要介绍系统的实现过程，小节顺序按照实际实现过程排列。实现过程主要包括环境搭建，神经网络搭建，模型训练，识别部分实现，整体组装等阶段。

4.1 系统环境搭建

4.1.1 笔记本电脑实验环境搭建

表 4.1 笔记本电脑实验环境参数

属性	参数
电脑型号	HP 256 G3 Notebook PC
CPU	4 核-Intel(R) Core™ i5-4210U @ 1.70GHz
GPU	NVIDIA GeForce 820M @ 775MHz
内存（RAM）	4.00GB
外存（ROM）	900GB
Python 版本	3.7

Windows 10 不是最好的开发环境，按部就班依照 Linux 系统下的配置流程进行实验环境搭建往往会出现各种各样难以调试解决的问题。本文直接选用 Anaconda 下的封闭 conda 完成实验环境搭建。封闭 conda 为带来了实验环境的灵活性，在设计实现过程中不必反复安装卸载同一个库的不同版本，取而代之的是每个特定 conda 为每个项目服务。为目前 Anaconda 已面向 Windows 用户提供图形用户界面，从安装到使用均实现了可视化（图 4.1）。

本论文首先创建一个封闭 conda 环境，命名为 tf-gpu，接着通过图形界面 Anaconda Navigator 向该 conda 添加实验必需工具（主要）：Python 3.7, numpy 1.16.3, pandas 0.24.2, OpenCV 3.4.2, TensorFlow 1.13.1（及其 python 依赖库）。

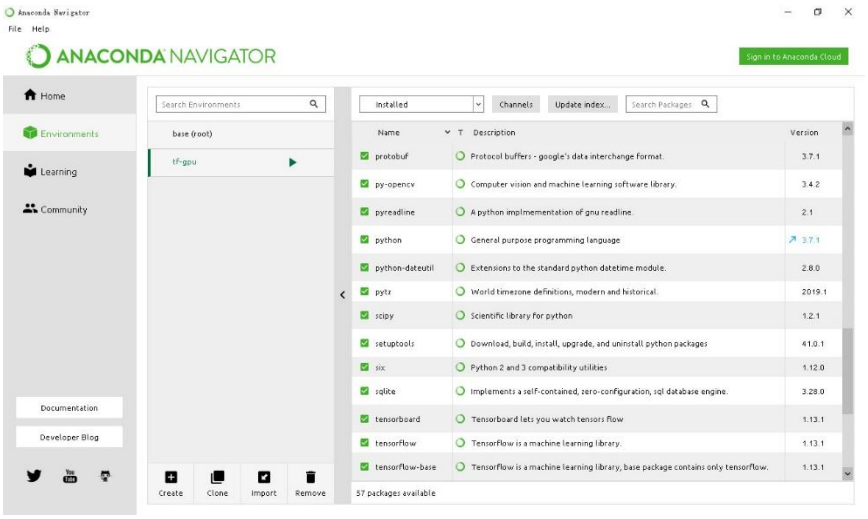


图 4.1 Anaconda Navigator——可视化 python 库管理工具

4.1.2 树莓派实验环境搭建

表 4.2 树莓派实验环境参数

属性	参数
型号	Raspberry Pi 3 Model B+
CPU	4 核-ARM Cortex-A7 @ 1.70GHz
GPU	Broadcom VideoCore IV @ 250MHz
内存（RAM）	1.00GB
外存（ROM）	32GB（Micro SD card）
摄像头	Raspberry pi Camera Module V2
显示屏	微雪 7inch HDMI LCD（C）（1024×600）

4.1.2.1 树莓派及配件组装

通过树莓派上的 HDMI 口连接显示屏的 HDMI 口完成显示功能，通过树莓派的一个 USB 口连接显示屏的 touch 口完成触控功能，使用网线连接树莓派和笔记本电脑的以太网口完成 VNC 连接，使用 5V 3A 电源适配器为树莓派供电，使用闪迪（SanDisk）32GB micro SD 卡作为树莓派外存，通过树莓派官方摄像头连接相机串行接口完成视频流输入功能，使用塑料散热片覆盖 CPU/GPU 和以太网控制器/USB 控制器确保系统平稳运行。

图 4.2 树莓派主板与摄像头的连接情况

图 4. 2 树莓派主板与显示屏的连接情况

4.1.2.2 树莓派环境搭建及软件安装

本论文选择官方推荐的 Raspbian 操作系统，其是一个基于 Debian 的免费操作系统，针对 Raspberry Pi 硬件进行了优化。同时，Raspbian 提供了超过 35000 个包和预编译软件并将它们以轻便的格式打包起来，极大简化了树莓派上软件的安装过程。Raspbian 具体安装过程如下：

①从树莓派官网下载 Raspbian 最新版本 Stretch 的镜像。

②在笔记本电脑上使用 SDFormatter 软件将 micro SD 卡格式化。

③使用 Win32DiskImage 软件将 Raspbian 镜像烧写至 micro SD 卡。

④将 micro SD 卡插入树莓派，上电开机。

4.1.2.3 远程控制连接

本文尝试了两种连接方式：SSH（Secure Shell）和 VNC（Virtual Network Computing）。其中 SSH 需要笔记本端的 PuTTy 软件，通过树莓派的 IP 地址连接，使用树莓派的用户名和密码登陆进行命令行界面的控制。VNC 则需要笔记本端的 VNC Viewer 和树莓派端的 VNC Server，同样是通过树莓派的 IP 地址连接，使用树莓派的用户名和密码登陆从而进行图形界面的控制。两种方法各有优缺点，如在树莓派上进行代码调试或 python 库安装时，使用简洁迅速的 SSH 更为高效；在树莓派上进行整体系统调试或图像视频操作时，使用具有图形界面的 VNC 更为清晰。

4.1.2.4 软件换源

无论是使用 Raspbian 系统的 apt-get 还是 python 中的 pip 来安装软件或是 python 库，均需要从 Raspbian 指定的软件源下载安装。而两个安装工具的默认源都是国外网站，下载速度不足 10B/s，故首先将两个工具的下载源配切换为国内源。本文使用的均为东北大学和清华大学源，极大提升了软件工具安装效率。

4.1.2.5 系统关键框架安装

虽然树莓派的潜力近几年才得到发掘，但计算机视觉库 OpenCV，人工智能框架 TensorFlow 等最受欢迎的相关学习工具均已宣布支持树莓派。在实际安装中，本论文仍碰到了许多雷区和障碍。

为了后续其他设计方案的尝试，本文首先使用的是 python 的 virtualenv 库，其与 Anaconda 类似，将不同项目所需的 python 库分别安装在相对独立的环境中，并在安装高级库或是具体运行程序时避免 python 库版本不同带来的依赖冲突。

在实际运用此方法过程中出现了一些问题。在独立的 env 环境中安装新的 python 库需要使用 pip 工具安装预编译的 wheel 文件，而实际使用 pip 安装中我们发现，这种方法对 TensorFlow 和 OpenCV 的安装支持较差。本文使用 pip 安装的 opencv-python 库无法被调用，而使用 pip 安装 TensorFlow 又遇到了版本冲突。TensorFlow python 库的依赖之一是 h5py（一个 HDF5 二进制数据格式面向 python 的接口）而 pip 中不提供树莓派所需的 h5py 的 wheel 文件，依赖无法安装导致 TensorFlow 安装失败。而若要先通过源码编译的方式安装好 h5py，又存在版本不匹配的问题，同样无法安装 TensorFlow。

最终本论文直接在当前用户 pi 目录下进行安装，这样操作的好处是各个工具可以通过最高效的方式被安装，但也伴随着一个缺点——整个过程中我们需要时刻关注可能冲突的 python 库版本。

OpenCV 安装：

OpenCV 源码编译时间较长，完全安装后需要较大的磁盘空间。在这里我们针对树莓派的特性进行了一些优化，这些优化对后续其他工具的安装，系统的设计和实现均有好处。

①扩展树莓派文件系统

默认情况下 Raspbian 根文件系统的大小为 2GB，这些空间甚至不足以放下 OpenCV 的所有文件，因此我们首先通过 raspi-config 完成对文件系统的扩展。

②卸载冗余软件

Raspbian 自带大量编程学习软件和编程工具，此处我们暂时将一些用不到的，占用较多外存空间的软件卸载掉，如：Libre Office，Wolfram engine 等。本文使用了 32GB 的 micro SD 卡，能够应付像 OpenCV 这样的大型工具，但对于一些追求最低配置批量搭建系统而使用 8GB 卡的情况，这一操作能够节省许多空间。（Libre Office 和 Wolfram engine 卸载后能够腾出约 1GB 空间）

③安装依赖库 Numpy，创建 build 文件，编译

由于树莓派内存和 CPU 的限制，笔记本电脑上一分钟就能用 pip 完成 numpy 的安装。在树莓派上，安装过程需要约 12 分钟。而在同时调用 4 个 CPU 时，OpenCV 的源码编译过程也需要约 2 小时的时间。

源码编译过程将消耗大量的计算资源和存储资源，而树莓派的一大瓶颈就是其 1GB 大小的 RAM。为了防止编译过程中树莓派因 RAM 资源枯竭导致反复编译失败，　　我们首先对交换空间进行调整。交换空间作为计算机的重要存储部分，在 RAM 用尽时能够给正在调用内存的应用一个喘息的空间。但由于内存与外村存取速度的显著差异，交换空间往往只能起到一点辅助性作用。在本文中，我们将默认的 100MB 大小交换空间暂时提升至 1GB 大小，从而减少编译过程中的挂起或中断。

TensorFlow 安装：

TensorFlow 对树莓派支持较好，从 pip 可以直接进行安装。由于 TensorFlow 官方将一部分模型和功能整合到了 TensorFlow-hub，故我们同样使用 pip 安装好 tf-hub。目前 TensorFlow 针对计算能力较弱的可移动设备推出了 TensorFlow-Mobile 和 TensorFlow-Lite，这两个版本均支持树莓派。考虑到三个版本间模型转换时可能带来的差错，本文依然使用最稳定，支持最彻底的 TensorFlow。

由于是从源码编译，此处我们可以放心使用最新版本的依赖。而 apt-get 安装的 python 库往往比 pip 安装的要新，故选用 apt-get 进行安装。

TensorFlow 的版本间差距较大，常常出现函数或类所在位置的更新带来的警告信息。为了避免过多警告对模型鲁棒性带来的影响，为之后的延伸开发提供较好的体验，本文使用最常用，也最受欢迎的稳定版本 TensorFlow 1.13。

4.2 视频输入及处理模块

本文使用树莓派官方发布的 Camera Module v2 进行实时视频采集。树莓派针对 Camera 提供了官方 python 库 picamera，考虑到其功能与 cv2 相比仍有不足，故使用 cv2 作为本文主要的 python 图像库。另外，本文用到的人脸识别技术同样需要调用 cv2 中的类和函数。

对于采集到树莓派的视频流中的每一帧，使用 cv2.cvtColor()进行处理，其中：

code=cv2.COLOR_BGR2GRAY

为了配合模型达到最佳识别效果，将图片颜色通道数由 3 修改为 1，与参与训练的 fer2013 数据集中图像样本的颜色通道数完全一致。

4.3 人脸识别及图像预处理模块

对于每一张灰度图片，使用 cv2 库中的 CascadeClassifier 类实例调用 detectMultiScale()函数，其中：

scaleFactor=1.3

相邻扫描窗口放缩倍数为 1.3。由于本系统主要考虑单人出现在镜头前（多人中最靠近摄像头的人）的情况，故对较为常见的 1.1 倍进行适当增大。

minNeighbors=5

含有监测目标相邻矩形的最小个数为 5。由于本系统主要功能为面部表情识别（人脸检测），故对常见的 3 个进行适当增大，确保识别准度。

此处本文对 detectMultiScale()函数的返回值 faces 进行处理。由于每个 face 对象是由[图像左上角 x 坐标, 图像左上角 y 坐标, 图像宽度, 图像高度]的列表组成，本文通过比较 faces 中每个 face 列表第三个值和第四个值的乘积来进行最大图像挑选。

对于截取到的人脸图片，使用 cv2.resize()函数进行处理，其中：

shape=(48, 48)

同样为了配合模型达到最佳识别效果，将图片像素尺寸修改为（48，48），与参与训练的 fer2013 数据集中图像样本的尺寸完全一致。

interpolation=cv2.INTER_CUBIC

图像插值方式选用了 4*4 像素邻域内的双立方插值。该插值法效率较低但效果较好，比较各种插值方法的使用对识别时间影响可忽略不计，故选用

INTER_CUBIC。resize 后的结果同样除以 255.0，将像素值范围改变至 0~1 的浮点数。

4.4 面部表情识别模块

4.4.1 自建卷积神经网络

本文提出了一种简易的卷积神经网络模型，通过 TensorFlow 框架实现。

第一个卷积层输入张量形状为[-1, 48, 48, 1]，其中-1 会被总体样本数量替代。第一个卷积层包含卷积，偏置，激活函数，池化，标准化五种操作：

(1)卷积功能通过 tf.nn.conv2d()函数实现，传入的参数是：

filter=[5, 5, 1, 64]

——卷积核大小为 5*5，输入频道数为 1，输出频道数为 64

strides=[1, 1, 1, 1]

——不跳过任何一个样本，也不跳过任何一个颜色通道（用来训练该网络的数据集 fer2013 本就是灰度图像，颜色通道仅一维），在宽度上每次移动一位，在高度上每次移动一位，即卷积核将以图像的每个像素点作为计算中心进行一次计算。

padding=’SAME’

——当卷积核来到图像边缘时，以’0’作为像素值填满图像外围需要被用来计算的“像素点”。由于 fer2013 数据集绝大多数图像都是人脸处于图像中心且人脸边缘与图像边缘几乎相切，故 SAME 模式能够更有效地保留图像边缘的人脸面部特征信息并将其加入计算中。

经过卷积，张量形状变为[-1, 48, 48, 64]

(2)偏置功能通过 tf.constant()函数实现，传入的参数是：

value=0.1

——初始偏置值赋予 0.1，作为比较常用的偏置值之一

shape 值由各层结合具体给出，等于各层卷积后的输出频道值。

经过偏置，张量形状不变[-1, 48, 48, 64]

(3)激活功能由 tf.nn.relu()函数实现，传入的参数是：

feature——上一步卷积后的张量与偏执参数张量的矩阵和

ReLU（Rectified Linear Unit，修正线性单元）[31]函数，如图 4.4 所示，是深度神经网络中一种较受欢迎的激活函数，一经提出便成为研究热点。

$$f(x) = x^{+} = \max(0, x) \tag{4.1}$$

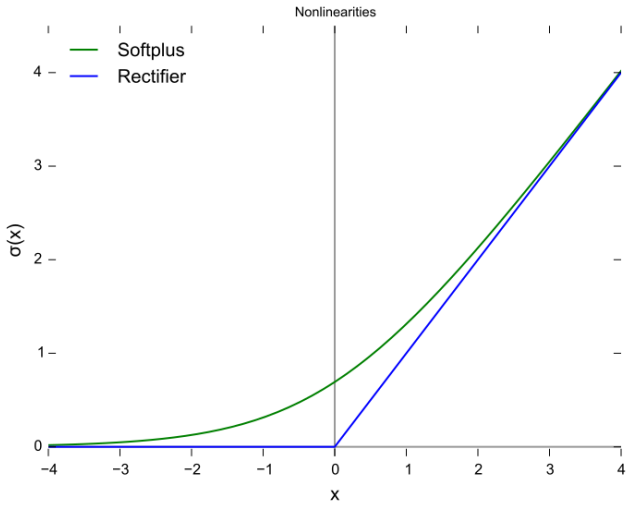


图 4.4 蓝色函数为 ReLU，绿色是其平滑近似得到的函数 softplus（softplus 的梯度为 softmax 函数）

使用 ReLU 作为激活函数的优点在于：

1.稀疏激活：在随机初始化的网络中，只有约 50%的隐藏单元被激活。

2.抑制梯度弥散：非负区间内梯度始终为常数，确保模型收敛速度较为稳定，从而更好地挖掘相关特征，拟合训练数据。

3.加快计算：ReLu 在程序中很好实现，可通过简单的 if-else 语句完成。

$$ReLU(x) = \begin{cases} x & if\ x > 0 \\ 0 & if\ x \le 0 \end{cases} \tag{4.2}$$

经过激活，张量形状不变[-1, 48, 48, 64]

(4)池化功能由 tf.nn.max_pool()函数实现，传入的参数是：

ksize=[1, 3, 3, 1]

——不对样本做池化，也不对颜色通道做池化，池化窗口的宽度为 3，高度也为 3。

strides=[1, 2, 2, 1]

——不跳过任何一个样本，也不跳过任何一个颜色通道，宽度上每次移动两位，高度上每次移动两位

padding=’SAME’

——与卷积类似，只是第一步池化时会把池化核放在图像内部左上角，当向右向下扫过图像发现像素点不足时，把图像周围需要用来做池化的像素点用’0’填充。

经过池化，张量形状变为[-1, 24, 24, 64]

(5)标准化功能本文使用了两种方法，分别由 tf.nn.lrn()函数（local response normalization）和 tf.layers.batch_normalization()函数实现，lrn()传入的参数是：

depth_radius=4——归一化半径，见公式 5.3 中的 2/n

bias=1——偏移量，见公式 5.3 中的 k

alpha=0.001/9.0——超参数，见公式 5.3 中的α

beta=0.75——超参数，见公式 5.3 中的β

LRN 在 2012 年在 AlexNet 的论文中被提出[32]，作者表明当 LRN 被用在 ReLU 前一层时，能够帮助模型更快速生成。其公式如 5.3：

$$b_{x,y}^{i} = a_{x,y}^{i}/(k + \alpha \sum_{j=\max\left(0,i-\frac{n}{2}\right)}^{\min\left(N-1,i+\frac{n}{2}\right)} \left(a_{x,y}^{i}\right)^{2})^{\beta} \qquad (4.3)$$

其中 i 代表像素值下标，j 代表平方累加索引，N 代表该层中的总核数，α，β，k，n 均是来自验证集的超参数。

batch_normalization()传入的参数是：

training=Ture——确保 TensorFlow 返回当前 batch 的归一化结果。同时，若将 training 参数设为 True，则需要在每个 session 中更新损失函数。

为了防止单纯的归一化对网络表达能力的破坏，BN 引入两个可以学习的参数γ和β，分别用来对归一化后的输出进行放缩和平移。BN 的公式如图 4.5 所示。

$$\mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{i,j}$$ Per-channel mean, shape is D

$$\sigma_j^2 = \frac{1}{N}\sum_{i=1}^{N}(x_{i,j} - \mu_j)^2$$ Per-channel var, shape is D

$$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}$$ Normalized x, Shape is N x D

$$y_{i,j} = \gamma_j \hat{x}_{i,j} + \beta_j$$ Output, Shape is N x D

图 4.5 BN 中各参数及输出计算公式[33]

其中μ_j代表每个 batch 的均值，σ_j代表每个 batch 的方差，$\hat{x}_{i,j}$代表归一化过程，$y_{i,j}$代表放缩，平移后的最终输出。

通过模型训练对上述两种方法进行比较发现，batch_normalization()比 lrn()能够更好更快地使训练精度提升。但在具体实施过程中，BN 层会消耗大量运算资源，仅能达到使用 LRN 层的网络训练速度的约三分之一。

从 LRN 的提出到 VGGNet 对 LRN 的否定，再到 BN 的提出，可以发现，LRN 主要是对于所有 feature map 上同一点也就是同一通道进行归一化，而 BN 是对每一张 feature map 进行归一化。在查阅资料过程中发现，如今 BN 的使用率更高，但 LRN 的作用不能被全盘否定。

第二个卷积层输入张量形状为[-1, 24, 24, 64]，其中-1 会被总体样本数量替代。第二个卷积层包含卷积，偏置，激活函数，池化，标准化五种操作：

(1)卷积功能通过 tf.nn.conv2d()函数实现，传入的参数是：

filter=[3, 3, 64, 64]

——卷积核大小为 3*3，输入频道数为 64，输出频道数为 64。与第一个卷积层相比，本层缩小了卷积核的尺寸，而不改变 feature map 的个数。

strides=[1, 1, 1, 1]

padding=’ SAME’

经过卷积，张量形状变为[-1, 24, 24, 64]

(2)偏置功能通过 tf.constant()函数实现，传入的参数是：

value=0.1

——初始偏置值赋予 0.1，作为比较常用的偏置值之一

shape——由各层结合具体给出，等于各层卷积后的输出频道值。

经过偏置，张量形状不变[-1, 24, 24, 64]

(3)激活功能由 tf.nn.relu()函数实现，传入的参数是：

feature——上一步卷积后的张量与偏执参数张量的矩阵和

经过激活，张量形状不变[-1, 48, 48, 64].

(4)池化功能由 tf.nn.max_pool()函数实现，传入的参数是：

ksize=[1, 3, 3, 1]

strides=[1, 2, 2, 1]

padding=’ SAME’

经过池化，张量形状变为[-1, 12, 12, 64].

(5)标准化功能由 tf.layers.batch_normalization()函数实现，传入的参数是：

training=Ture

第一个全连接层输入张量形状为[-1, 12, 12, 64]，其中-1 会被总体样本数量替代。第一个全连接层包含全连接，偏置和激活三种操作：

全连接可被视为一个全局卷积运算，将分布式特征表示映射到样本标记空间，从而减少特征位置对最终面部表情分类带来的影响。本文拟将表情分为 7 类，故设计了 384（全连接层）-192（全连接层）-7（线性层）的结构。

(1)全连接功能由 tf.matmul()函数实现，传入的参数是：

a——将上层网络的输出拉平至形状为[-1. 12 * 12 * 64]的张量。

b——各个数值以标准差为 0.1 进行初始化的形状为[12 * 12 * 64, 384]的卷积核。

经过全连接，张量形状变为[-1, 384]

(2)偏置功能通过 tf.constant()函数实现，传入的参数是：

value=0.1

经过偏置，张量形状不变[-1, 384]。

(3)激活功能由 tf.nn.relu()函数实现，传入的参数是：

feature——上一步卷积后的张量与偏执参数张量的矩阵和

经过激活，张量形状不变[-1, 384]

第二个全连接层输入张量形状为[-1, 384]，其中-1 会被总体样本数量替代。第二个全连接层包含全连接，偏置和激活三种操作：

(1)全连接功能由 tf.matmul()函数实现，传入的参数是：

a——上层网络的输出，形状为[-1. 384]的张量。

b——各个数值以标准差为 0.1 进行初始化的形状为[384，192]的卷积核。

经过全连接，张量形状变为[-1, 192]

(2)偏置功能通过 tf.constant()函数实现，传入的参数是：

value=0.1

经过偏置，张量形状不变[-1, 192]。

(3)激活功能由 tf.nn.relu()函数实现，传入的参数是：

feature——上一步卷积后的张量与偏执参数张量的矩阵和

经过激活，张量形状不变[-1, 192]

线性层输入张量形状为[-1, 192]，其中-1 会被总体样本数量替代。本文将线性层以与全连接层类似的方式进行具体实现，对整个张量进行全局卷积运算。线性层包含线性化，偏置两种操作。

(1)线性化功能由 tf.matmul()函数实现，传入的参数是：

a——上层网络的输出，形状为[-1. 192]的张量。

b——各个数值以标准差为 0.1 进行初始化的形状为[192, 7]的卷积核。

经过线性化，张量形状变为[-1, 7]

(2)偏置功能通过 tf.constant()函数实现，传入的参数是：

value=0.1

经过偏置，张量形状不变[-1, 7]。

4.4.2 迁移学习——Google Inception

本文还使用了 TensorFlow 中的 retrain.py 脚本，基于已训练好的 Google Inception 模型在 JAFFE 和 fer2013 上开展训练（具体训练参数如图 4.6）。在实际实现过程中，由于此方法重训练出的模型在树莓派上运行速度过慢（对每个切割出的人脸开展表情识别速度约为 3 秒/张），带来的实时体验较差，故暂时弃用此模型。

```
python retrain.py \
--bottleneck_dir=/retrained_data/bottlenecks \
--how_many_training_steps 4000 \
--model_dir=/retrained_data/inception \
--output_graph=/retrained_data/retrained_graph.pb \
--output_labels=/retrained_data/retrained_labels.txt \
--image_dir /retrained_data/dataset
```

图 4.6 retrain.py 脚本运行参数

4.4.3 识别过程

调用 deepnn()重建训练时的网络，并使用 tf.nn.softmax()对网络模型输出结果进行预测，其中：

logits=y_conv

——各表情类别得分情况

调用 tf.train.Saver()取出训练好的模型对人脸图像进行识别

对于优化算法，本文采用 Adam(Adaptive Moment Estimation)Optimizer。Adam 相比于其他优化算法有如下优点：

①不容易陷入局部最优解

②优化速度较快，收敛速度快，学习效果更明显

系统中使用 tf.train.AdamOptimizer 类实例实现，其中：

learning_rate=0.0001

——学习率α

损失函数 cross entropy 使用 tf.nn.softmax_cross_entropy_with_logits()函数实现，其中：

labels=y_

——全部图像样本标签以供验证

logits=y_conv

——全部图像样本预测标签以供验证

4.5 结果展示模块

系统实时显示区域左上角作为评分区域输出表情种类和各表情种类识别得分情况.

使用 cv2.putText()函数打印出其中表情标签名作为条形图刻度。

使用 cv2.rectangle()函数对各表情类别得分按比例进行条形图各条形绘制。

使用 numpy.argmax()函数对各表情类别评分取最大值，将其标签所对应的表情以 emoji（emoji 图）代替。

键盘等待’ ’键（空格键）状态，若’ ’键被按下，将系统当前界面的状态存为 PNG 格式图片，包括视频输入内容，haar 级联分类器识别出的人脸标记，当前人脸对应点识别条形图和表情所替换的 emoji。

键盘等待’Q’键状态，若’Q’键被按下，摄像头停止采集视频，系统退出。

4.6 关键技术点实现

①选取网络上规模较大，属性符合识别目标的数据集，随机抽取其中一部分数据通过人眼进行可行性判断。

图 4.7 fer2013 数据集中图像示例

表 4.3 目前网络上较为常见的人脸表情数据集

数据集	数据量	表情主体	表情数	图像大小	描述
CK+	593	123	7	640×490/ 640×480	数据中 327 个图像带有表情标签
MMI	326	32	6	未知(高)	人物带有配饰，数据中 213 个图像带有表情标签
JAFFE	213	10	7	256×256	均为日本女性
Fer2013	35887	未知	7	48×48	通过谷歌图片接口收集
AFEW	1806	未知	7	未知	一个图像中有多个主题，多个面部表情
Multi-PIE	755370	337	6	3072 x 2048	4 个场景，9 种光照条件下收集

所有图像的标签分类情况如表 4.4 所示：

表 4.4 fer2013 数据集中表情标签分类情况

标签	表情	图像数
0	愤怒的	4593
1	厌恶的	547
2	恐惧的	5121
3	快乐的	8989
4	难过的	6077
5	惊讶的	4002
6	中性的	6198

在 Kaggle 竞赛中，28709（80%）和 3589（10%）张图像被分享给参赛者，分别作为训练集和公开测试集，组委会将剩下的 3589（10%）张图片保留不公开，作为最终的测试集。在本次竞赛结束后，整个数据集都可以被访问。

从 Kaggle 官网获取的 fer2013 数据集存储在 csv 文件中，具体存储格式如图 4.8。该存储格式为研究人员和业余爱好者提供了较为轻松的数据获取方式——本文通过 python 的 pandas 库可以轻松完成对 csv 文件的操作。同时，数据集的排列格式完全按照当时比赛的情况还原，若实践过程中不改变数据集划分格式，则该格式为后续的训练结果提供了基准参照物。

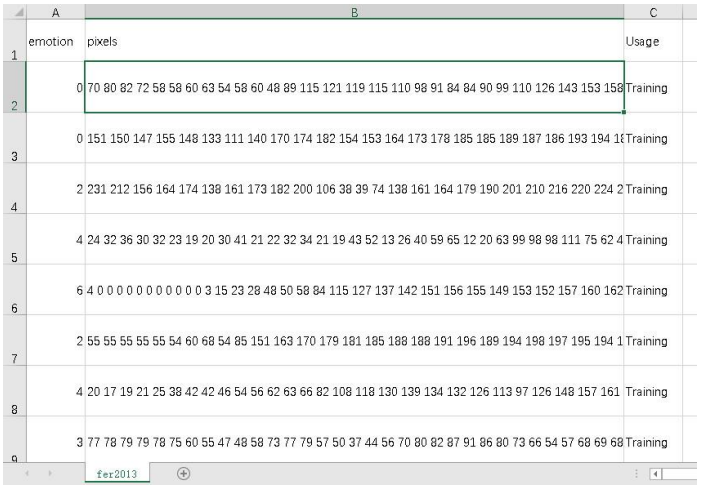


图 4.8 fer2013 数据集 csv 存储格式

②对 fer2013 数据集进行适当调整，尝试输入不同大小的图像进行训练。

由于 fer2013 数据集已是灰度图像，且图像尺寸已被统一到大部分神经网络可开展训练的大小，故本文首先尝试将原始图像投入神经网络中。单张图像的矩阵形状可表示为（48，48，1）其中三项数据分别代表图像宽度，图像高度和图像颜色通道值。

③对数据集进行划分，借鉴较为常见的处理方法并通过具体训练结果进行反馈。同时尝试对处理方法进行修改已达到最好效果。

在神经网络模型训练过程中，经过多种分割尝试，本文沿用了 80%-10%-10%的分割方法，也即留出法（holdout）。首先通过训练集开展训练，80%的数据量确保了对网络充分的训练，10%的验证集防止训练结果过拟合，另 10%的测试集确保训练后的神经网络模型准确而有说服力。

④本文在选取，设计网络模型过程中，首先采用了许多较为经典的网络模型开展训练，再对结果和训练效率进行比较，最终设计了一个对树莓派较为友好的网络模型。同时，在实现过程中，本文将迁移学习加入到系统当中，作为另一种

模型的训练手段，将自建神经网络模型和基于迁移学习重新训练的网络模型进行对比，从而提升系统效率。

本文使用 TensorFlow 在 ImageNet 上预训练好的模型 Inception v3。

⑤为深入观察训练过程中模型的变化情况，本文通过调用 xxx 库，使用图表的方式观察，分析模型状态，做出更好的调整。

⑥在人脸识别技术上，本文综合各因素考虑，最终选择使用 OpenCV 中的 Haar-cascade 人脸识别模块。

在默认情况下，依靠程序中的 while 循环对视频输入流进行识别，即完成一次识别，进行人脸切割，将切割后图片传递给下一步的神经网络模型。下一次识别尝试将在本次识别完成后立刻开始，从而使系统达到“实时”人脸表情识别。

在实际实验中，这一做法过于“实时”了。原因是面部表情在大多数时候都能维持一段时间，而不是转瞬即逝的。过于频繁的人脸识别会导致过于频繁的表情识别，从而将一些边缘表情（本文所研究的七种表情间的变化）切割下来传递给分类模型。表情识别的一个重要应用就是观察主体的表情变化，目前实现对表情变化的观察，主要是通过对一段时间内稳定表情（较短时间内无明显变化的表情）的记录和分析完成的。

而由于具体变化过程中的人脸是难以被识别和分类的（譬如一个嘴唇两端微微向下弯的表情，既可以被理解为从中性表情到难过表情的过渡，也可以被理解为从难过表情到中性表情的过渡），这类被快速捕捉到的边缘表情往往不能有效体现表情的变化。因此，当系统识别人脸频率过高时，会捕捉到很多边缘表情。这些边缘表情一方面拉低了整体面部表情识别的准确率，另一方面也会浪费不必要的计算资源。

⑦Haar 级联人脸分类器通过不同批次的特征检测来完成对人脸的识别，在实际应用过程中，其能够通过四个坐标（起点坐标 x，起点坐标 y，宽度，高度）来完成对人脸范围的大致确定，但由于不同观察主体的特征有尺寸上的变化，导致人脸范围有时排除掉了一部分下巴，有时排除掉一部分额头。

与一般的人脸识别不同，本系统要求识别人脸时主体面部特征不能被切割掉，否则会影响后续的面部表情识别。这是由于表情的形成往往调动整个面部的肌肉，在整个面部区域内都有其动态变化的体现，直接导致面部的线条，阴影等特征的变化。如果一部分特征信息随着面部图像的切割丢失，则会导致模型对表情的误判。

故此，本系统在实现过程中尝试对切割结果进行一些无差别补偿。（此处将宽和高各扩大 10 个像素）

⑧本系统通过图文结合的形式展现识别结果。

当系统开始运行时，摄像头捕捉到的实时视频流被输出在屏幕上。若检测到人脸，则用方框标出感兴趣区域（Region of Interest），将识别结果通过横向柱状图的形式展现在系统界面左上角，同时以 Apple Emojis 对应的匹配 emoji 将识别结果以表情形式展示在系统界面左下角。系统界面是用 xxx 实现的。

对数据及进行搭建一个人脸表情识别的神经网络模型，将成熟的模型作为系统的核心，对输入的视频流中截取的人脸进行识别，并给出一个识别结果。本系统的最终结果是面向用户的，因此很有必要保证其以一个较为友好的界面进行结果的展示。

5 实验结果

本章展示了神经网络模型训练的结果和系统实际运行的结果。将本文提出模型与其他模型进行对比，对比其各方面表现。对于整个实时人脸表情识别系统，本章展示了多个测试对象在系统上的表现，并进行结果分析。

5.1 模型训练结果

本文分别使用三种环境分别完成训练（表 5.1），其中模型预设最大训练步数（max_training_steps=40001）

表 5.1 训练过程

平台	时长	处理器
服务器	40001-12 分钟	GPU——2*GeForce GTX 1080 8G
笔记本电脑	~21000-8 小时	CPU——4*Intel(R) Core™ i5-4210U
树莓派	速度过慢，无法统计	CPU——4*ARM Cortex-A7

尽管树莓派搭载的 ARM CPU 和博通（Broadcom）GPU 赋予其一定的计算能力，使其能够运行一些复杂程序，并拥有了播放 1080P 视频的能力，但这与目前主流的计算能力水平仍差若干个数量级。由上表分析可得，相比于服务器和普通的笔记本电脑，树莓派的计算能力使它暂时无法承担训练神经网络模型这样繁重的任务。由于近年来计算机视觉方向的突破大多来自深度学习方法，特别是卷积神经网络和深度卷积神经网络的各类变形和演化推动的。可以得出结论，树莓派无法独自完成此类图像识别任务。

本文使用 tensorflow 的 Saver 类对参数进行存储，配合 checkpoint()函数构建模型。由于是从较为简单的网络结构训练而来，最终训练完成后模型大小仅为 41.2M。该文件大小有利于通过各种途径向树莓派进行传输。本文利用 Xshell 中的 Xftp 从服务器将模型下载到树莓派上，对于其他文件则是利用笔记本和树莓派之间的 VNC 连接进行传输。必需文件传递完毕后，树莓派可自行工作。在运行系统时，树莓派的 CPU 占用率保持在 50%左右，最高峰值达到约 75%。

5.1.2 优化结果

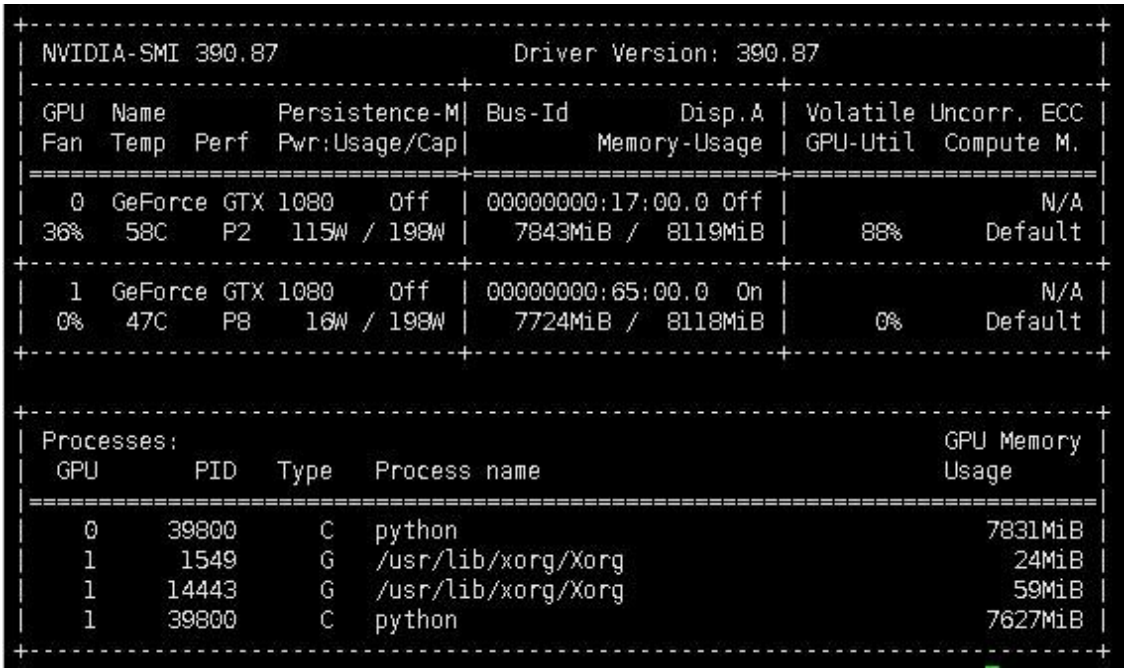


图 5.1 服务器 GPU 属性及神经网络训练时性能

5.2 系统预测结果

5.2.1 人脸切割优化

对 fer2013 数据集中的各数据样本进行观察可以发现，其中一半以上图片包含较为完整的人脸（刘海，头顶，双耳外侧，下巴尖）。在 haar 级联分类器的正面人脸识别中，面部区域内，最靠上的 haar 特征是眼睛，鼻梁，眉毛之间的差异性特征，最靠下的 haar 特征是嘴唇和下嘴唇下方的差异性特征。通过这些特征面部其他部分的特征所确定的人脸区域相比宽度近似，高度略小。故本文在 haar 级联分类器检测结果的基础上进行调整。将返回数据进行调整(如图 5.2，红色椭圆代表人脸范围，绿色方框为 haar 级联分类器返回结果，蓝色方框为调整后结果)：

[left_top_corner_x, left_top_corner_y, width, height]->

[left_top_corner_x, left_top_corner_y - 15, width, height + 15]

调整后识别准确率提升 5%。

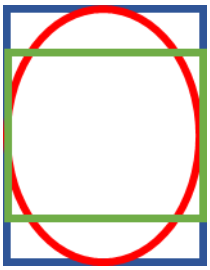


图 5.2 人脸切割部分调整示意图

5.2.2 识别准确率分析

表 5.2 模型训练结果

深度学习模型	识别准确率	模型运行时间	数据集	单张识别时间
本文提出的模型	59.3%	25s	fer2013	0.23
Inception_v3_retrained	54.2%	21s	fer2013	2.51
综述文献[26]引用的模型	75.2%	None	Fer2013	None

由于 fer2013 数据集中图片样本最少的类别厌恶的（disgusted）仅有 547 张样本，与图片样本最多（8989 张）的类别快乐的（happy）相比，仅占其 6.1%；与数据集全部图片样本（35587 张）相比，仅占其中 1.5%。故其在具体系统预测时，常被误识为其他类别，导致此类表情出现很多假负结果（False negative）

此外，在人类表情系统中，许多表情对、表情组之间往往没有清晰的界限，常常会被识别成近似的表情。因此，仅仅一个短暂的表情不能组成全部用于判断当前情感的要素。同时，许多表情常常成对，成组发生，一个表情常常引导出其他表情，或是相近间常常没有清晰的界限，

①厌恶和难过，在面部表情上有相似的特征，如眉毛下压，眼睛微闭，双唇微抿，唇角下压等等。无论是训练前打上表情标签的过程还是识别时匹配表情标签，这些相似点都对两种表情之前的区分增大了难度。

②快乐，作为一个涵盖较广的表情词汇，可以用来标记大部分积极性表情图像。同时，识别快乐表情的面部表情特征与识别其他表情的面部表情特征区别较

大，如伴随着快乐的牙齿露出，颧骨肌肉大幅度上扬及其联动的嘴角肌肉上扬等等。这些特征的独特性使得识别快乐表情比识别其他表情更轻松，具体体现在特征向量的明显差异。

③由于 fer2013 数据集所有图像均通过爬虫获取并对其中的人脸进行剪切，一小部分图像样本中出现了完全与人脸及表情特征完全无关的噪声（如图 5.3），对识别结果有一定影响。

图 5.3 无关噪声样本

④同时，fer 数据集中一部分图像中出现了人手遮挡面部的情况，其中大部分遮挡的是人脸下半部分，集中在嘴唇，嘴角，两颊和下巴区域（如图 5.4）。这些遮挡噪声对识别结果有一定影响。

图 5.4 人脸遮挡图像样本

5.2.3 表情识别偏误

对于标记为 disgusted 的表情，其可能发生的偏误有：

(a) (b) (c)

图 5.5 disgusted 表情标签偏误情况

①disgusted 可以被解读为 neutral 或是 angry

②disgusted 可以被解读为 sad

③disgusted 可以被解读为 sad 或是 fearful

对于标记为 sad 的表情，其可能发生的偏误有：

(a) (b) (c)

图 5.6 sad 表情标签偏误情况

①sad 可以被解读为 neutral

②sad 可以被解读为 surprised

③sad 可以被解读为 fearful

对于标记为 neutral 的表情，其可能发生的偏误有：

(a) (b)

图 5.7 neutral 表情标签偏误情况

①neutral 可以被解读为 sad

②neutral 可以被解读为 happy

对于标记为 angry 的表情，其可能发生的偏误有：

(a) (b) (c)

图 5.8 angry 表情标签偏误情况

①angry 可以被解读为 sad 或 surprised

②angry 可以被解读为 neutral

③angry 可以被解读为 disgusted

对于标记为 happy 的表情，其可能发生的偏误有：

图 5.9 happy 表情标签偏误情况

①happy 可以被解读为 surprised

对于标记为 surprised 的表情，其可能发生的偏误有：

(a) (b) (c)

图 5.10 angry 表情标签偏误情况

①surprised 可以被解读为 fearful

②surprised 可以被解读为 disgusted

③surprised 可以被解读为 happy

对于标记为 fearful 的表情，其可能发生的偏误有：

①fearful 可以被解读为 natural

②fearful 可以被解读为 angry

(a) (b)

图 5.11 fearful 表情标签偏误情况

同时，上述所有解读具有主观性，如对于某个表情的分析往往联系到该表情发生前的上个表情和发生后的下个表情，从而影响对于数据集标签的判断。对于

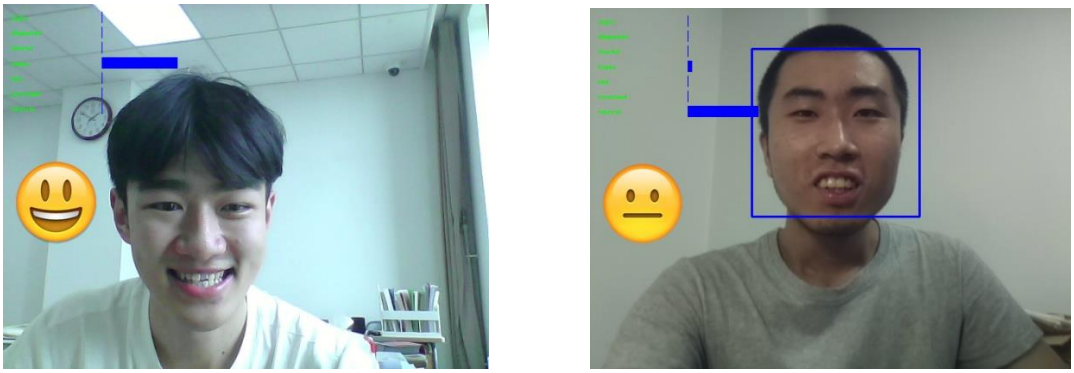
展示在面前的表情，每一个人都有不同的解读方式和结果，这些差异可能直接导致对于 fer2013 数据集以及表情分类准则的不同意见。

5.2.4 系统识别结果

通过在不同光照条件下的检测可以发现，本系统在弱光照条件下的识别效果较差，特别依赖白天较为均匀的日光。在夜间进行测试时，普通屋顶的 LED 照明灯会导致识别正确率骤降。在夜间配以台灯正面光照进行测试时，正确率会回升，但与白天的识别准确度比还有一定差距。在同等条件下，系统在笔记本电脑上的识别效果比在树莓派上的识别效果要好，这是因为笔记本电脑自带摄像头光源和来自屏幕的正面光照，而本系统中使用的树莓派 Camera Module V2 则表现较差。另一个导致识别结果差异的因素可能是摄像头的稳定性：笔记本电脑的摄像头内嵌在屏幕上方中央，几乎无晃动；而在树莓派上运行时，仍然是通过手持摄像头进行视频输入，大大小小的晃动难以避免。

在系统实际运行过程中，本文记录了识别过程中的一些正例（测试主体做出该表情被正确识别）和负例（测试主体做出该表情被错误识别），用以分析和改进：

happy：

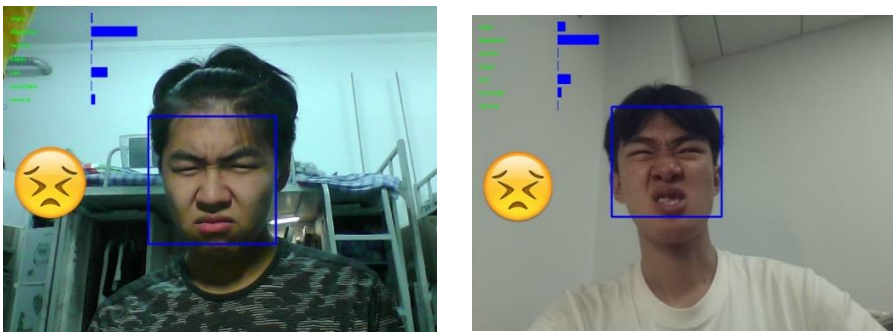


(a) (b)

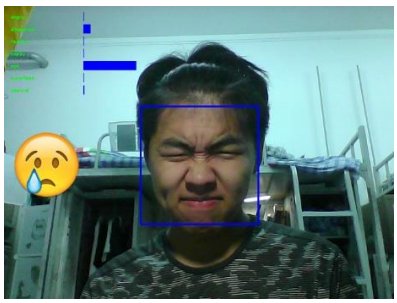
图 5.12 happy 正例与负例

disgusted：在尝试进行 disgusted 表情识别时，多位测试主体在所有尝试中完成了大量反例和极少量正例。这与 disgusted 主要具有的面部特征有关：眉头

禁皱，整个鼻子向上皱，嘴角用力。同时也与 disgusted 与 sad 较为相似的面部特征有关。



(a) (b)



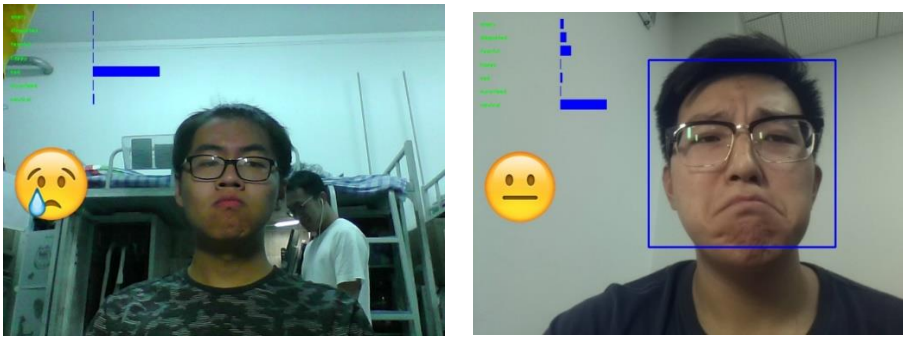
(c)

图 5.13 disgusted 的两个正例和一个反例

neutral：neutral 的特征较为清晰，嘴唇呈一条线，其他面部区域没有明显变化，故较容易正确识别。

图 5.14 neutral 的正例

sad：sad 的主要特征是嘴角下拉，嘴唇中心向上提，眉头有两种不同形态：禁皱和向两侧松弛。在识别过程中也有此现象，若眉头紧皱则容易与 disgusted 混淆。若两嘴唇紧紧抿住，则较细的唇线容易被忽略，或是连同嘴角肌肉被识别为 neutral。



(a) (b)

图 5.15 sad 的正例和反例

fearful：fearful 的嘴部特征与 surprised 类似，眉眼部特征与 sad 类似。而在实验中，试验主体在观察了 fer2013 数据集中的 fearful 样本后，仍表示难以作出可以被识别的 fearful 表情。

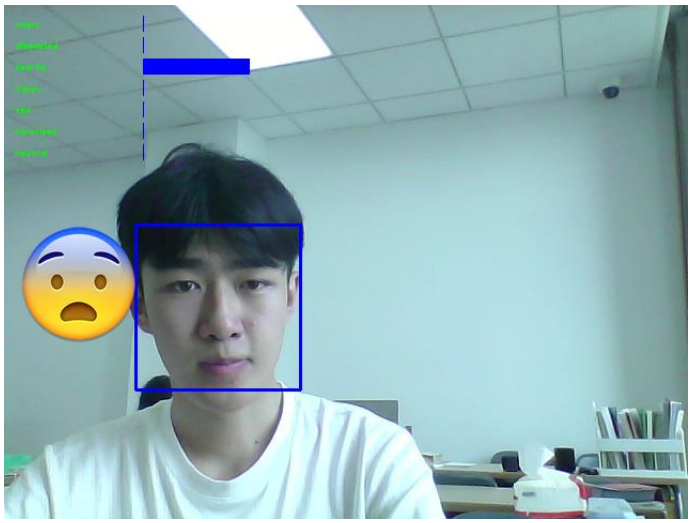
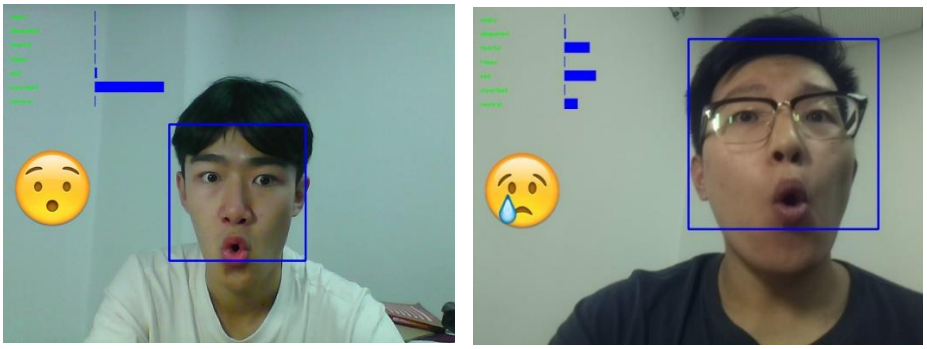


图 5.16 fearful 的一个反例

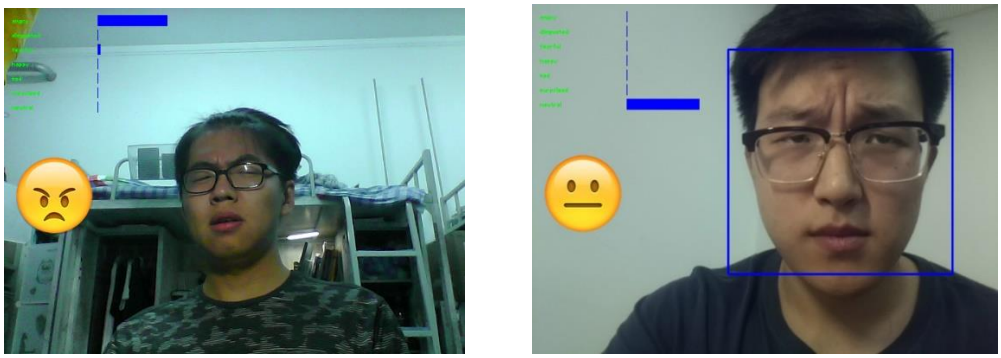
surprised：surprised 难以识别的情况主要有两部分构成：下颚下拉过低，导致本系统拉长后的人脸切割框仍无法捕捉到全部嘴唇信息；下颚下拉，上下唇呈’O’形时在下唇下侧造成了阴影。



(a) (b)

图 5.17 surprised 的正例和反例

angry：angry 的面部特征主要有：眉头紧皱，双唇紧抿。负例主要来源是双唇紧抿带来的 neutral 的误判和眉头禁皱带来的 disgusted 的误判。



(a) (b)

图 5.18 angry 的正例和反例

除了对具体样例的分析对未来特征提取方法的改进外，本系统仍有可以提升的空间：如过度实时性的改进，数据集的补充和改进，网络模型的提升等等。

以上，本文完成了对基于树莓派的实时人脸表情识别系统的设计和实现。

6 总结与展望

6.1 总结

本文从树莓派上开展人工智能研究的可行性出发，设计了一个简易的基于树莓派的人脸表情识别系统。在此过程中讨论比较了多种识别模型的优劣，针对需求设计了关键技术点，并针对树莓派的特性对模型进行调整和优化，最终系统能够基本完成识别任务。

本文完成的主要工作有：

（1）介绍了人脸表情识别和人脸识别的历史和最新发展情况。

（2）介绍了树莓派产品的发展情况和软硬件配置，简要分析了在树莓派平台上开展面部表情识别任务的可行性。

（3）训练一个小型卷积神经网络模型，尝试不同参数对结果带来的影响，根据模型在树莓派上的运行速度对模型进行优化。

（4）采用迁移学习的手段对已有图像识别成熟模型进行重训练，观察较大模型在树莓派上的运行速度。

（5）以直观的 emoji 代替文字对识别进行展示，同时考虑输出的美观性，将各类表情的评分一同展现。

但是，系统中同样存在一些缺陷，如光照不足的情况下表情识别率过低，以及对人脸正面角度及无遮挡的要求等等。同时，本文对潜在识别模型的探索还不够，尽管实时性能够基本满足，但最终结果仍待提升。其他可能的提升可以通过对数据集的增强，扩充，对于模型的改进等实现。

6.2 展望

树莓派的计算能力始终是其解决人工智能任务的首要难题，无论是模型训练还是模型的运行，对树莓派有限的处理能力和存储容量都是极大的考验。本文选取面部表情识别这一方向，在树莓派上尝试搭建一个简易的系统，得到了一些积极的成果，但更多未解决的问题需要进一步的研究。无论是传统机器学习方法还

是深度学习方法，都应进行广泛的尝试和改进，从而在有限计算能力的基础上达到较为满意的识别水平。

参考文献

[1] Mehrabian, A. Communication without words[J], Psychology Today, 1968, 2 (4), 53-55.

[2] T. Devries, K. Biswaranjan and G. W. Taylor, Multi-task Learning of Facial Landmarks and Expression[J], 2014 Canadian Conference on Computer and Robot Vision, Montreal, QC, 2014, pp. 98-103.

[3] A. T. Lopes, E. De Aguiar, and T. Oliveira-Santos. A Facial Expression Recognition System Using Convolutional Networks[J]. Brazilian Symposium of Computer Graphic and Image Processing, 2015, 2015-Octob:273–280.

[4] H. Jung, S. Lee, J. Yim, S. Park and J. Kim, Joint Fine-Tuning in Deep Neural Networks for Facial Expression Recognition[J], 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 2983-2991.

[5] P. Viola and M. Jones, Rapid object detection using a boosted cascade of simple features[J], 2001, Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, pp. I-I.

[6] Wikipedia. AdaBoost – Wikipedia[DB/OL]. https://en.wikipedia.org/wiki/AdaBoost, 2019

[7] P. Ekman, W. Friesen. Facial Action Coding System: A Technique for the Measurement of Facial Movement[J], 1978, Consulting Psychologists Press, Palo Alto.

[8] Computer Vision Lab, University of Massachusetts. Labeled Faces in the Wild – Results[DB/OL]. http://www.cs.umass.edu/lfw/results.html, 2019

[9] 龙鹏-言有三. [技术综述]人脸表情识别研究[DB/OL]. https://zhuanlan.zhihu.com/p/40572244, 2018

[10]周书仁. 人脸表情识别算法分析与研究[D]. 中南大学, 2009.

[11]周书仁, 梁昔明, 朱灿,等. 基于 ICA 与 HMM 的表情识别[J]. 中国图象图形学报, 2008, 13(12):2321-2328.

[12]何良华. 人脸表情识别中若干关键技术的研究[D]. 东南大学, 2005.

[13]应自炉, 唐京海, 李景文,等. 支持向量鉴别分析及在人脸表情识别中的应用[J]. 电子学报, 2008, 36(4):725-730.

[14]L. Ma, K. Khorasani. Facial expression recognition using constructive feedforward neural networks[J], in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 34, no. 3, pp. 1588-1595, June 2004.

[15]X.Feng, M.Pietikäinen, A. Hadid. Facial expression recognition with local binary patterns and linear programming[J]. Pattern Recognition and Image Analysis, 2005, 15, 546-548.

[16]付晓峰. 基于二元模式的人脸识别与表情识别研究[D]. 浙江大学,2008

[17]Jabid T, Kabir M H, Chae O. Local directional pattern (LDP) for face recognition[C]//2010 Digest of technical papers international conference on consumer electronics (ICCE). IEEE, 2010: 329-330.

[18]Y. Yacoob and L. S. Davis, Recognizing human facial expressions from long image sequences using optical flow[J], in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, no. 6, pp. 636-642, June 1996.

[19]F. Tsalakanidou, S. Malassiotis. Real-time 2D+3D facial action and expression recognition. Pattern Recognition. 43. 1763-1775. 10.1016/j.patcog.2009.12.009.

[20]J. Sung and D. Kim, Pose-Robust Facial Expression Recognition Using View-Based 2D+3D AAM[J], in IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 38, no. 4, pp. 852-866, July 2008.

[21]I. Kotsia and I. Pitas, Facial Expression Recognition in Image Sequences Using Geometric Deformation Features and Support Vector Machines[J], in IEEE Transactions on Image Processing, vol. 16, no. 1, pp. 172-187, Jan. 2007.

[22] Cohen I, Sebe N, Garg A, et al. Facial expression recognitionfrom video sequences: temporal and static modeling[J].Computer Vision & Image Understanding, 2003, 91(1–2):160-187.

[23] cs231n, Stanford University. K-Nearest Neighbors Demo[EB/OL]. http://vision.stanford.edu/teaching/cs231n-demos/knn/.2018

[24] 徐文晖,孙正兴.面向视频序列表情分类的 LSVM 算法[J].计算机辅助设计与图形学学报,2009,21(04):542-548+553.

[25] 徐琴珍,章品正,裴文江,杨绿溪,何振亚.基于混淆交叉支撑向量机树的自动面部表情分类方法[J].中国图象图形学报,2008(07):1329-1334.

[26] Li S, Deng W. Deep facial expression recognition: A survey[J]. arXiv preprint arXiv:1804.08348, 2018.

[27] Wikipedia. Raspberry Pi – Wikipedia. https://en.wikipedia.org/wiki/Raspberry_Pi[DB/OL], 2019

[28] Raspberry Pi Foundation. Camera Module V2. https://www.raspberrypi.org/products/camera-module-v2/[EB/OL], 2019

[29] Spoony. 3000 核！用 750 张树莓派挑战最经济超算. http://shumeipai.nxez.com/2017/11/22/750-raspberry-pi-challenge-the-most-economical-super-computing.html[DB/OL], 2017

[30] B. Alex, AIY Projects 2: Google’s AIY Projects Kits get an upgrade[EB/OL]. https://www.raspberrypi.org/blog/google-aiy-projects-2/, 2018

[31] Nair V, Hinton G E. Rectified linear units improve restricted boltzmann machines[C]//Proceedings of the 27th international conference on machine learning (ICML-10). 2010: 807-814.

[32] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.

[33] Stanford Vision and Learning Lab. cs231n[EB/OL].
http://vision.stanford.edu/teaching/cs231n/2017/, 2018

致谢

四年如一瞬，在东北大学的求学生涯即将画上句点。我懵懵懂懂踏入南湖公园旁的一栋栋教学楼，也常追着足球在五四五五奔跑整个下午。我满怀信心来到浑河之南的崭新校舍，体育馆，图书馆，一号楼，小南湖都留下了我的身影，有奋进，有拼搏，也有迷茫，有无助。时光匆匆远去，我成为了更好的我。

在大三结束的暑假里，我遇到了栗伟老师，冯朝路老师和他们的机器学习班，在这里我学到了计算机视觉的一些基本知识，和我的队友一起循序渐进地完成了图片分类的传统识别方法和深度学习方法。最终，我们的模型识别精度最高，取得了最好的成绩。那一个月的学习，每天与老师的沟通点燃了我对计算机视觉的兴趣，也让我找到了真正想要研究的方向。后来毕设论文选题时我主动找到栗伟老师，老师也很开心，积极引导我，帮助我把一个点子转化为了我毕业设计的题目。栗伟老师严谨治学的精神令我佩服，和蔼可亲的态度使我感动。在毕业设计开展期间，老师对我的指导和鼓励给了我很大的帮助，老师对我的信任使我深受鼓舞，在这里我要向栗伟老师深深地鞠一躬！借这个机会，我还要感谢李福亮老师。自打我入学，李老师就积极鼓励我参加科研，进入实验室磨练自己，他还多次指导我参加国家级的竞赛。虽然最后没能成为您的研究生，但我会始终记得您宝贵的意见和亲切的叮嘱。同样要感谢其他为我带过课的老师，不只在专业领域，你们真正打开了新世界的大门，并引领着我在其中徜徉。祝你们工作顺利！

感谢我的三位室友，我们一起玩，一起闹，一起学习，一起奋斗。不是兄弟，胜似兄弟。谢谢你们对我的照顾和帮助，我们未来的求学路上，是星辰和大海。

感谢我的爸爸妈妈对我一如既往的支持和鼓励。没有你们，我无法完成在这里的学业；没有你们，我的心将四处漂泊。

感谢李文惠同学四年来的陪伴。距离虽远，情意始终相连。

最后，我谨向所有关心和帮助过我的领导，老师，同学，家人致以最真诚的感谢。评阅本文的各位专家老师，你们辛苦了！