信息工程学院校企实训

分组项目报告

|  |  |
| --- | --- |
| 项目名称 | 基于Keras和TensorFlow的深度卷积神经网络识别人脸和情绪 |
| 项目负责人 | 刘晓飞、李坤、郑贺文 |
| 专业 | 计算机科学与技术1603 |
| 任课教师 | 胡英铭 |
| 编制时间 | 2019.7.11 |

西北农林科技大学信息工程学院

2019年6月制

项目成员及分工

（工作量总和为 100%）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 学号 | 姓名 | 工作内容 | 工作量（%） |
| 01 | 2016012854 | 郑贺文 | 实现情绪识别，答辩ppt编写 | 35% |
| 02 | 2016012863 | 刘晓飞 | 实现人脸识别，界面设计，答辩ppt编写 | 35% |
| 03 | 2016012861 | 李坤 | 接口设计，实训文档编写，项目整合测试 | 30% |
|  |  |  |  |  |

项目总打分表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 指标内容 | 分 值 | 指标内涵及评估标准 | | | | 得 分 |
| A | B | C | D |
| 构思（20 分） | | | | | |  |
| 选题意义 | 10 | 意义重大 | 意义较大 | 意义一般，属于简单开发 | 无意义 |  |
| 技术路线的可行度 | 10 | 合理可行，具体且有创新 | 合理可行,具体 | 基本合理可行 | 不够合理 |  |
| 设计（20 分） | | | | | |  |
| 设计内容 | 10 | 内容非常丰富 | 内容较丰富 | 内容一般 | 内容欠缺 |  |
| 解决的关键技术问题 | 10 | 准确，范围合 适，重点突出 | 基本准确 | 部分关键 | 未抓住关键 |  |
| 实现（20 分） | | | | | |  |
| 项目完成的技术水平 | 10 | 难度很大，达到较高水平 | 难度较大，超出一般水平 | 难度一般，达 到普通水平 | 难度小，很容易实现 |  |
| 团队精神 | 10 | 团队合作精神 强 | 合作情况良好 | 合作情况一般 | 合作不好 |  |
| 运作（20 分） | | | | | |  |
| 应用程度 | 20 | 产业化或实际 应用 | 模拟应用 | 可以应用但未应用 | 不能应用 |  |
| 文字表达及文档制作水平（10 分） | | | | | |  |
| 文字表达 | 5 | 文字表达非常 好 | 文字表达较好 | 文字表达一般 | 文字表达差 意思不明了 |  |
| 文档制作 | 5 | 制作非常专业 化 | 制作良好 | 制作一般 | 制作效果差 |  |
| 报告质量及口头表达能力（10 分） | | | | | |  |
| 报告质量 | 5 | 报告非常完整 | 报告比较完整 | 完整程度一般 | 报告不完整 |  |
| 口头表达能力 | 5 | 整体效果很好 | 整体效果良好 | 整体效果一般 | 整体效果差 |  |
| 总分： | | | | | |  |
| 评语： | | | | | | |

个人期末项目成绩表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 学号 | 年级 | 专业 | 姓名 | 成绩 |
| 01 | 2016012854 | 2016 | 计算机科学与技术 | 郑贺文 |  |
| 02 | 2016012863 | 2016 | 计算机科学与技术 | 刘晓飞 |  |
| 03 | 2016012861 | 2016 | 计算机科学与技术 | 李坤 |  |
|  |  |  |  |  |  |

说明：

1. 第二阶段实训任务及报告完成情况＝个人期末项目成绩＝小组总成绩×小组人数×个人工作量＋教师调控（范围－10～＋10）
2. 个人学期总成绩＝劳动纪律和实训态度×20%＋第一阶段实训任务及报告完成情况×40%＋第二阶段实训任务及报告完成情况×40%

项目报告应当分为以下模块进行编写

# 项目背景

人脸识别部分：

1. **人脸识别定义**

人脸识别，是基于人的脸部特征信息进行身份识别的一种生物识别技术。通常采用摄像机或摄像头采集含有人脸的图像或视频流，并自动在图像中检测和跟踪人脸。

1. **人脸识别优势**

**非侵扰性：**人脸识别无需干扰人们的正常行为就能较好地达到识别效果，无需担心被识别者是否愿意将手放在指纹采集设备上，他们的眼睛是否能够对准虹膜扫描装置等等。只要在摄像机前自然地停留片刻，用户的身份就会被正确识别。

**便捷性：**采集设备简单，使用快捷。一般来说，常见的摄像头就可以用来进行人脸图像的采集， 不需特别复杂的专用设备。图像采集在数秒内即可完成。

**友好性：**通过人脸识别身份的方法与人类的习惯一致，人和机器都可以使用人脸图片进行识别。而指纹，虹膜等方法没有这个特点，一个没有经过特殊训练的人，无法利用指纹和虹膜图像对其他人进行身份识别。

**非接触性：**人脸图像信息的采集不同于指纹信息的采集，利用指纹采集信息需要用手指接触到采集设备，既不卫生，也容易引起使用者的反感，而人脸图像采集，用户不需要与设备直接接触。

**可扩展性：**在人脸识别后，下一步数据的处理和应用，决定着人脸识别设备的实际应用，如应用在出入门禁控制、人脸图片搜索、上下班刷卡、恐怖分子识别等各个领域，可扩展性强。

**3.人脸识别应用领域**

**金融领域。**人脸识别当前在金融领域的应用最为广泛，当前国内金融领域监管要求严格，金融相关产品都需要实名认证，并且具有较高的安全性要求

**安保领域。**目前大量的企业，住宅，社区，学校等安全管理越来越普及，人脸门禁系统已经成为非常普及的一种安保方式。

**通行领域。**很多城市的火车站已经安装了人脸识别通行设备，进行人证对比过检，有些城市的地铁站也可以通过人脸识别的方式进行地铁进出站通行。

**泛娱乐领域。**现在市场上火爆的美颜相机，网络直播，短视频等都是建立在人脸识别的基础上对人脸进行美颜和特效处理。

**公安，司法领域。**公安系统在追捕逃犯时也会利用人脸识别系统对逃犯进行定位，监狱系统目前也会对服刑人员通过人脸识别系统进行报警和安防。

自助服务设备。如银行的自动提款机，无人超市等。

1. **人脸识别发展现状**

根据中国报告网发布《2018 年中国生物识别市场分析报告-行业深度分析与发展前景预测》中内容，2017 年生物识别技术全球市场规模上升到了 172 亿美元，到 2020 年，预计全世界的生物识别市场规模有可能达到 240 亿美元。自 2015 年到 2020 年，人脸识别市场规模增长了 166.6%，在众多生物识别技术中增幅居于首位，预计到 2020 年人脸识别技术市场规模将上升至 24 亿美元。

1. **人脸识别难点**

（1）人脸表情复杂，人脸具有多样的变化能力，人的脸上分布着五十多块面部肌肉，这些肌肉的运动导致不同面部表情的出现，会造成人脸特征的显著改变；

（2）人脸随年龄而改变，随着年龄的增长，皱纹的出现和面部肌肉的松弛使得人脸的结构和纹理都将发生改变；

（3）人脸有易变化的附加物，例如改变发型，蓄留胡须或者佩戴帽子和眼镜等饰物；

（4）人脸特征遮掩，人脸全部、部分遮掩将会造成错误识别；

(5)人脸图像畸变，由于光照、视角、摄取角度不同，可能会造成图像的灰度畸变、角度旋转等，降低了图像质量，增大了识别难度。

**5.人脸识别发展前景**

（1）基于大数据的大规模人脸搜索是人脸识别技术未来发展的重要方向。

（2）随着三维测量技术的发展，基于3D的人脸识别算法能够弥补2D投影造成有效识别信息丢失的问题，对于人脸旋转、遮挡、极度相似的传统难点具有很好的解决方式。

（3）随着我国城市化进程的加速，社会稳定、城市安全等问题逐渐显现，而人脸识别技术是实现安全、安心城市的关键技术。

（4）智能视频分析将是大安防市场未来的方向之一，而人脸识别是其中非常重要的技术和应用。

（5）智能家居与人脸识别技术的融合是未来发展的重点方向。

情绪识别部分：

1. **发展现状**

国外，美国、日本、英国、德国、荷兰、法国、印度、新加坡都有专门的研究组进行这方面的研究。其中MIT、CMU、Maryland 大学、Standford 大学、日本城蹊大学、东京大学、ATR 研究所的贡献尤为突出。Pantic 等通过提取正面和侧面的人脸轮廓特征点，获得 32个肌肉运动单元的信息，识别6 类表情。Mase等使用光流法来跟踪运动单元，采用K近邻方法进行分类，对4 类表情的识别率为80% 。Shan等用局部二值模式描述人脸特征，结合SVM在低分辨率下识别面部表情，获得了较好的效果。

国内清华大学、哈尔滨工业大学、中科院、中国科技大学、南京理工大学、北方交通大学。如哈尔滨工业大学的金辉、高文基于动态图像序列中的光流信息，通过隐马尔可夫模型识别任意时序长度的、复杂的混合表情图像序列，对7种表情的总体识别率达到96.9%。北京科技大学的杨国亮，王志良等在标准光流法的基础上提出了非刚体光流算法，构建 HMM 与BP神经网络混合分类器。对CMU库中愤怒、厌恶、悲伤、惊奇四种表情的识别率超过 80% 。

1. **情绪识别应用领域**

**人机交互：**在现今网络社会里，计算机已经不是仅仅被视为用于计算的机器了，它还应该提供各种使人们相互交流的功能。其中，影响的实时传输就是一个很重要的功能，当传输速度很慢时可以用摄像机里的表情分析器对用户表情进行分析，然后只传输分析结果，同样能达到好的交流效果。

**安全领域：**表情识别可以用于核电站的管理和长途汽车司机等着重强调安全的工作岗位。在岗者一旦出现疲劳。瞌睡的征兆，识别系统及时发出警报避免险情发生；还可以用与公安机关的案件侦破中。

**智能机器人的研制：**我国成功研制出能模仿人类40多种表情的机器人。在表情的具体实现上，使用了11个可以自由控制的节点，分别分布在眉毛、眼皮、眼珠和嘴上。通过11个自由节点的自由组合，在通过智能控制，它就能把各种表情展现出来。

**医疗领域：**表情识别还可以用于机器人手术操作和电子护士的护理。可根据患者面部表情变化及时发现其身体状况的变化，避免悲剧发生。

**远程教育：**远程授课时老师普遍认为很难掌握学生们对课程的理解程度，这也就需要一种表情识别其把学生们的表情规定为对课程的掌握程度并返回给袁成的教师，老师们就可以做出相应的反应了。。

**电脑游戏：**这可能是脸部表情识别和值作为有应用价值的一个领域了。如果一款游戏可以根据游戏这的喜怒哀乐来做出实时的反应，那么这款游戏肯定比那些传统与游戏更加吸引人。

1. **情绪识别难点**

（1）每种表情都是面部不同肌肉组合的结果，无法用一个固定的模型描述每种表情对应的面部肌肉组合。

（2）人脸面部表情变化时，表现为脸部特征点的运动，目前计算机不能精确定位特征点的具体位置。

（3）即使是同一种表情，不同人面部表现形式也不一样，计算机精确划分每种表情很困难。

**4.情绪识别难点**

（1）识别表情类别有待增加。

（2）人脸检测算法的鲁棒性有提升空间。

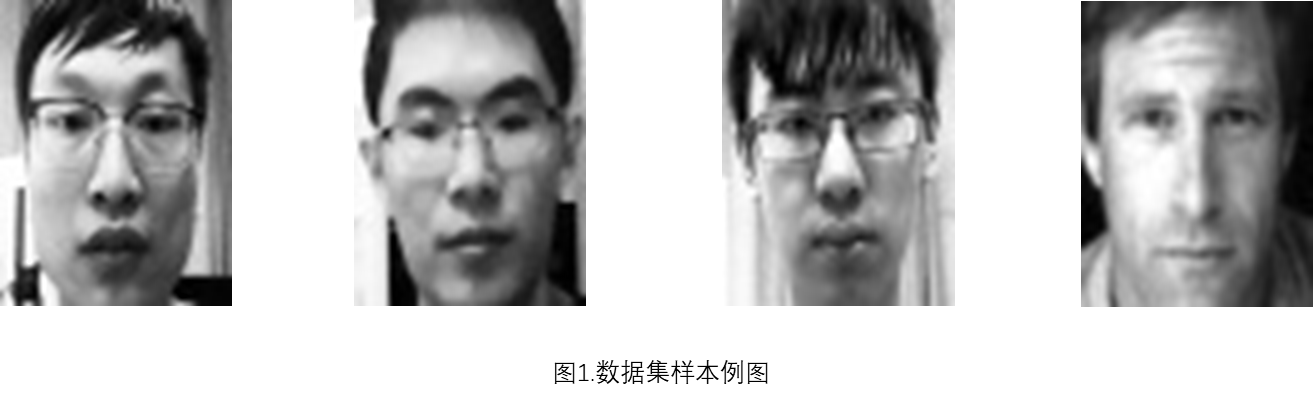
（3）表情识别计算量较大。

# 项目概述

本项目名称为《基于Keras和TensorFlow的深度卷积神经网络识别人脸和情绪》，实时人脸识别和情感分类，使用FER2013数据集和Keras CNN模型和OpenCV。

人脸识别部分：

人脸识别使用自己创建的数据集，准确率为99%，测试准确率为98.5%



**人脸识别的步骤：**

1.用OpenCV打开摄像头

2.利用OpenCV自带的人脸探测器探测人脸

3.采集不同人的人脸并准备数据集

4.对数据集进行预处理

5.用自己的数据集训练神经网络模型用于人脸识别

6.用训练好的数据集实现人脸识别

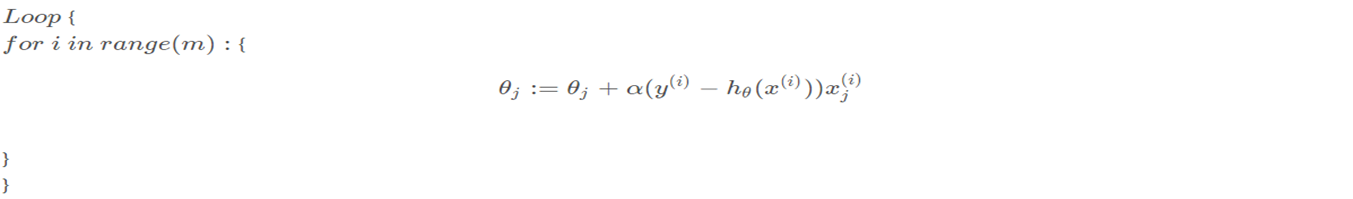
**使用的人脸识别算法：特征脸法（Eigenface）**

首先把一批人脸图像转换成一个特征向量集，称为“Eigenfaces”，即“特征脸”，它们是最初训练图像集的基本组件。识别的过程是把一副新的图像投影到特征脸子空间，并通过它的投影点在子空间的位置以及投影线的长度来进行判定和识别。

**人脸识别优化算法：SGD优化算法**

随机梯度下降算法（Stochastic gradient descent，SGD）在神经网络模型训练中，是一种很常见的优化算法。这种算法是基于梯度下降算法产生的。

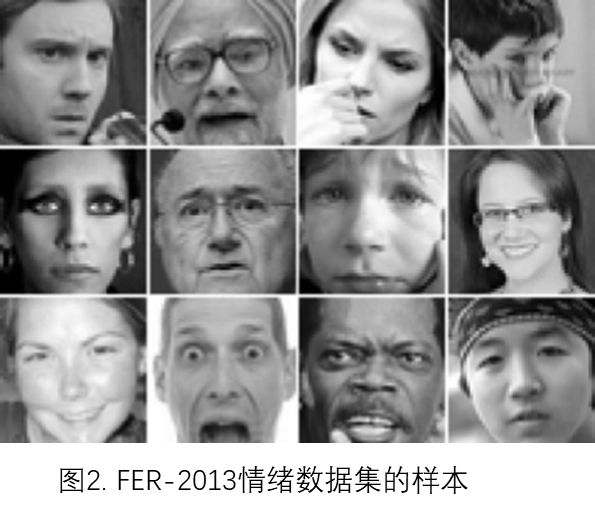
这个算法的流程就是在每次更新的时候使用一个样本进行梯度下降，所谓的随机二字，就是说我们可以随机用一个样本来表示所有的样本，来调整超参数 θ \thetaθ ，算法的公式如下所示：



因为这个样本是随机的，所以每次迭代没有办法得到一个准确的梯度，这样一来虽然每一次迭代得到的损失函数不一定是朝着全局最优方向，但是大体的方向还是朝着全局最优解的方向靠近，直到最后，得到的结果通常就会在全局最优解的附近。这种算法相比普通的梯度下降算法，收敛的速度更快，所以在一般神经网络模型训练中，随机梯度下降算法 SGD 是一种非常常见的优化算法。

情绪识别部分：

-FERD2013情绪分类训练集准确率：74.5%，测试集准确率：67%



**情绪识别步骤**

1.卷积神经网络模型的训练

（1）获取数据集

（2）搭建卷积神经网络模型

（3）训练模型

2.人脸情绪识别模块

（1）图像预处理

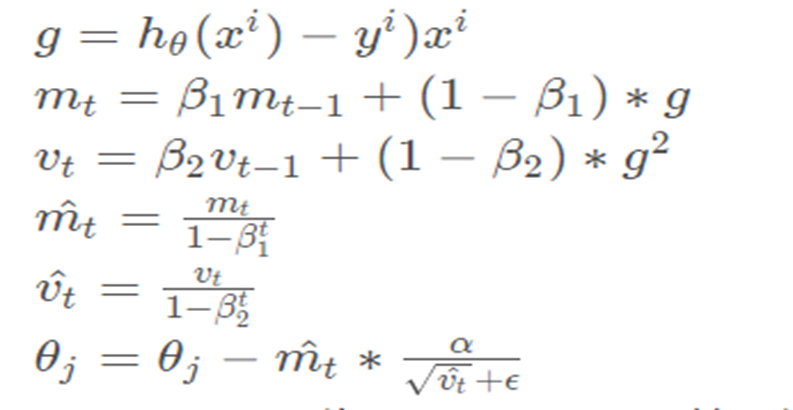
（2）人脸表情识别

**表情识别优化算法：Adam算法**

Adam是一种自适应学习率的方法，在Momentum一阶矩估计的基础上加入了二阶矩估计，也是在Adadelta的基础上加了一阶矩。它利用梯度的一阶矩估计和二阶矩估计动态调整每个参数的学习率。为了解决Adagrad算法出现学习率趋向0的问题，还加入了偏置校正，这样每一次迭代学习率都有个确定范围，使得参数比较平稳。

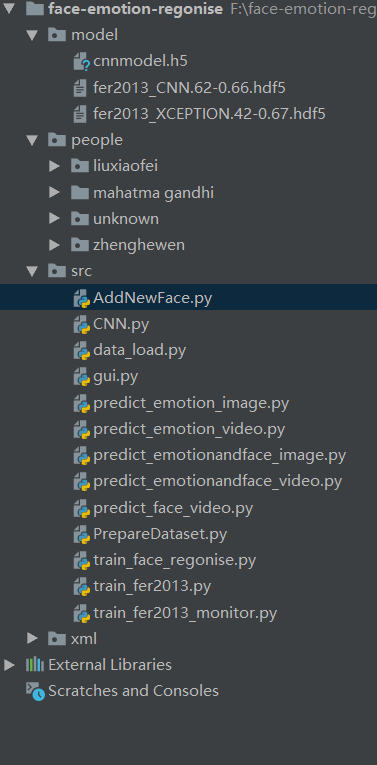
它的优劣点：迭代速度快，效果也好，但可能不收敛。

迭代公式为：



总体设计部分：

**文件概述**



1.Addnewpace.py

添加人脸，从摄像头捕获照片并保存

2.cnn.py

情绪识别神经网络模型

3.data-load.py

处理人脸表情数据集fer2013

4.gui.py

界面

5.predict\_emotionandface\_image.py

识别图片人脸和情绪

6. predict\_emotionandface\_video.py

通过摄像头实时识别人脸和情绪

7.prepareDataset.py

定义一些图像处理的函数集合，包括检测人脸，直方图均衡化处理等

8.train\_face\_regonise.py

训练人脸识别

9.train\_fer2013.py

训练情绪识别

10.train\_fer2013\_monitor

训练情绪识别的自定义回调函数

**接口设计**



无参数

无返回值

实现的功能：

打开摄像头，获取图像，识别图像中的人脸和情绪



传入参数：image\_path图片路径

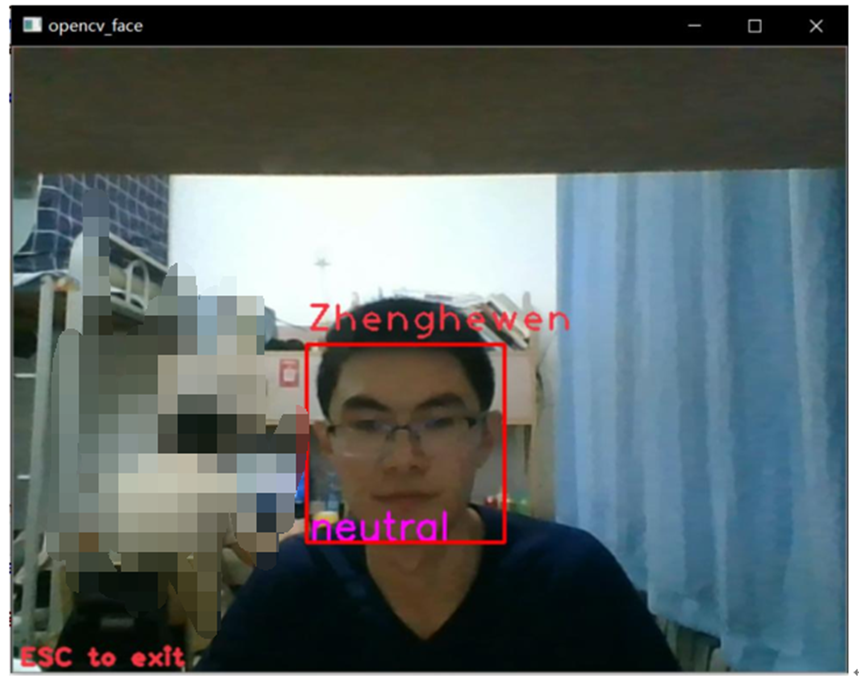
返回值：返回处理后的图像，返回模型预测后的结果

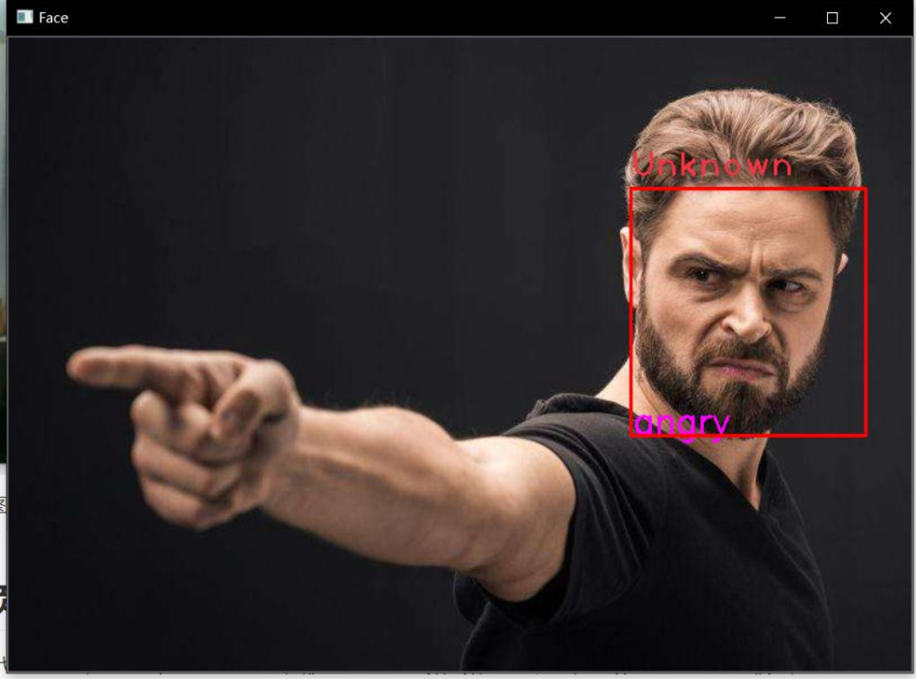
实现功能：

根据传入的图片识别人脸和情绪

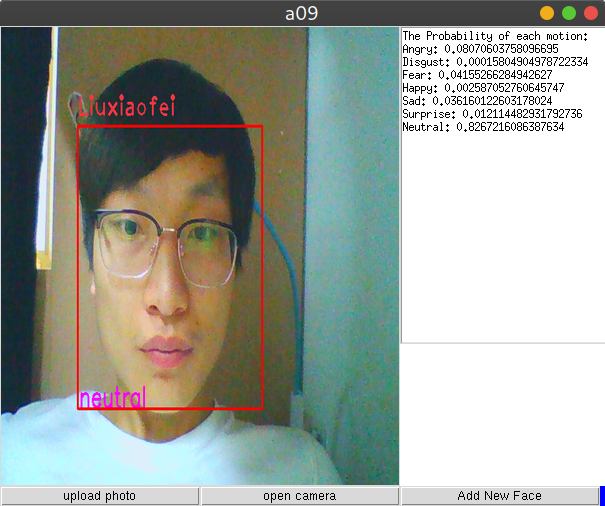
**项目整合**

情绪识别与人脸识别代码进行整合测试结果：



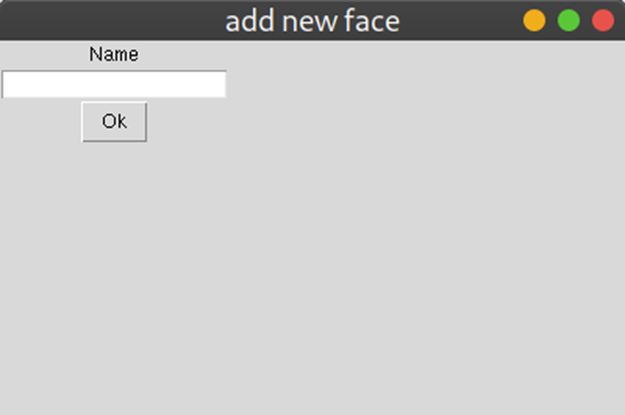


**项目使用**



1.点击“upload photo”可以从计算机中选择一张照片导入，若检测出人脸则会显示人脸姓名和情绪，右边显示各个情绪的概率，输出概率最大的情绪。

2.点击“open camera”可以打开摄像头实时检测人脸做出人脸识别和情绪识别。



3.点击“Add New Face”会出现此弹窗，输入想添加的人脸姓名，再点击“ok”，会打开摄像头自动捕捉人脸并训练。

# 四、 环境配置

Python==3.7.3 keras==2.2.4 TensorFlow==1.13.1 numpy==1.16.2

Opencv-python==4.1.0 pandas==0.24.2

# 五、 项目实现

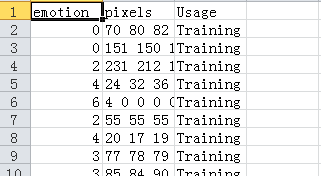
情绪识别部分：

**1.获取与处理数据集**

查阅资料，Fer2013、CK+是用于表情识别的数据集，这里选用常用的Fer2013。

Fer2013人脸表情数据集由35886张人脸表情图片组成，其中，测试图28708张，公共验证图和私有验证图各3589张，训练集与测试集比例为8：2。每张图片是由大小固定为48×48的灰度图像组成，共有7种表情，分别对应于数字标签0-6，具体表情对应的标签和中英文如下：0 anger 生气； 1 disgust 厌恶； 2 fear 恐惧； 3 happy 开心； 4 sad 伤心；5 surprised 惊讶； 6 normal 中性。

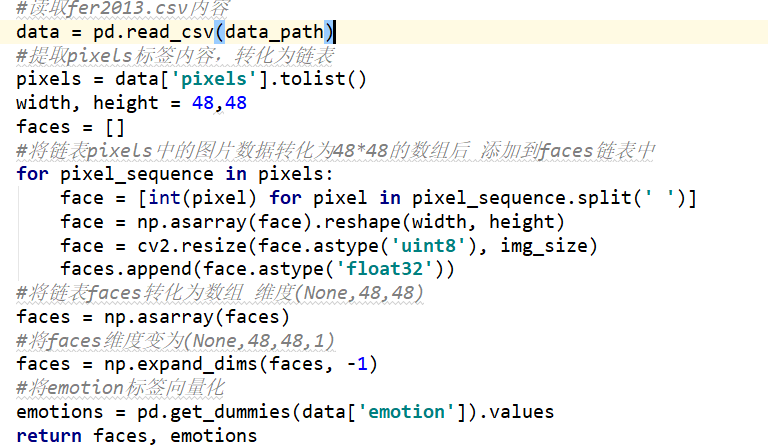
数据集将表情、图片数据、用途的数据保存到csv文件中，如下所示：



**2.解析与处理数据集**

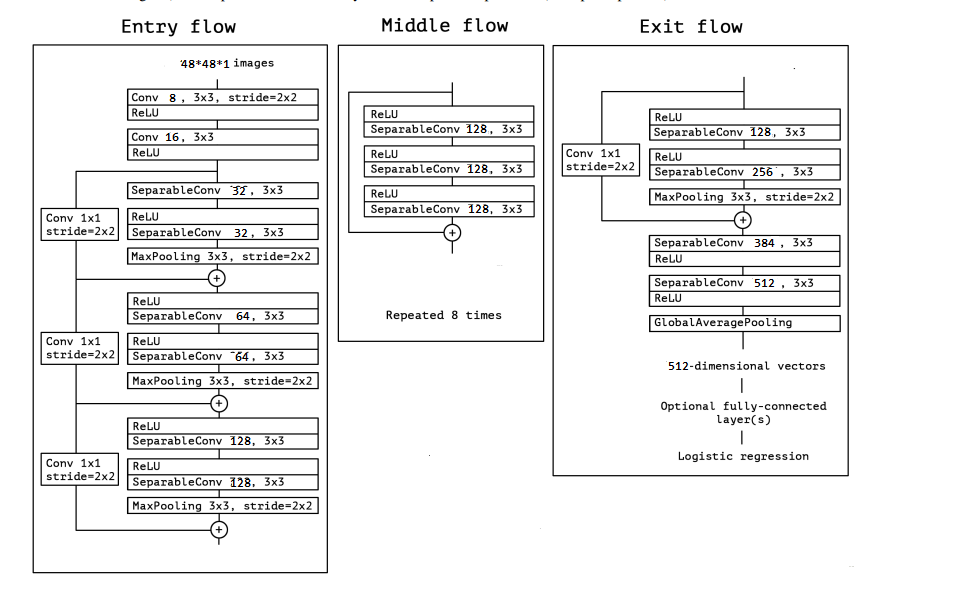
使用pandas解析数据集，根据标签提取内容，对提取的内容进行格式转换，与神经网络模型输入输出形状一致。

划分前80%为训练集，20%为测试集。



**3.Xception神经网络**

按照下图搭建神经网络模型



**4.模型配置**

优化器： 'adam',

损失函数：'categorical\_crossentropy'

数据增强：使用ImageDataGenerator对数据集旋转、平移、缩放、垂直翻转。

Batch-size : 32

Epoch: 200

回调函数：

EarlyStopping：当测试集损失函数训练50轮不优化停止训练。

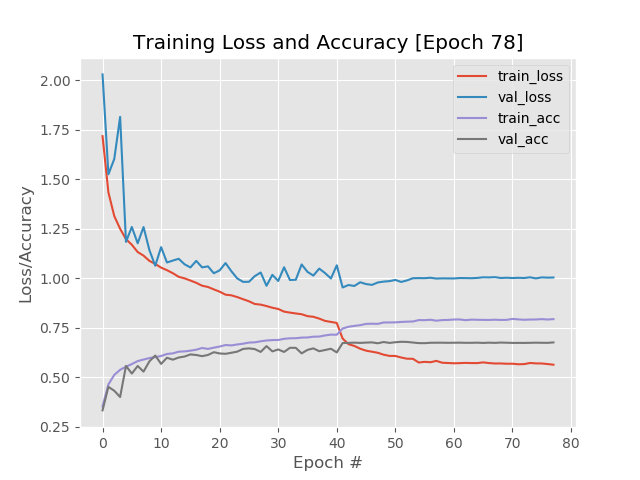
ReduceLROnPlateau：当测试集损失函数训练10轮不优化降低学习率至原来的10%。

CSVLogger：保存训练日志。

ModelCheckpoint：测试集损失函数优化时保存训练模型。

TrainingMonitor：自定义回调函数，实现训练过程可视化，每个epoch后绘制损失函数与准确率图像。

**5.训练结果**



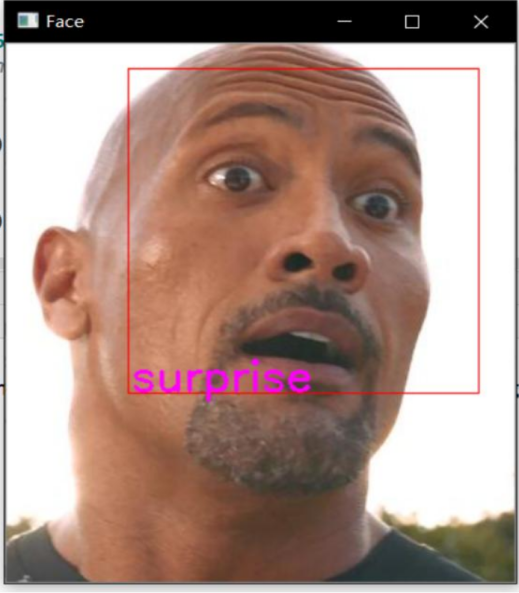
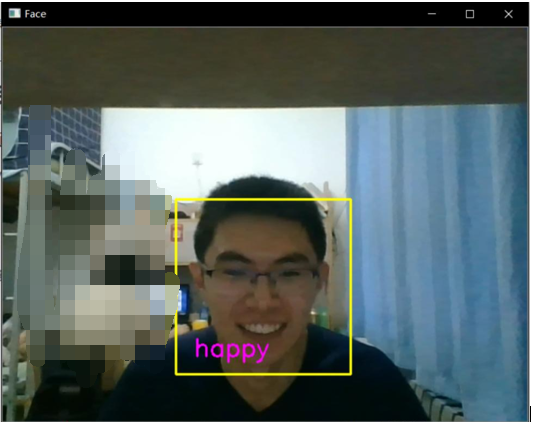
从图中可以看到，训练集损失函数逐步下降，准确率逐步上升，测试集则不断波动。第41轮学习率降低后，第42轮测试集损失函数达到最低，此后虽然训练集仍在优化，但测试集数据不再提升。第78轮手动停止训练，读取保存的模型重新开始训练仍无提升。

保存第42轮的训练模型，训练集准确率：74.5%、损失函数：0.69，测试集准确率：67% 、损失函数：0.95。

**6.测试**

使用Opencv获取图像数据，使用人脸分类器检测人脸位置，处理检测到的人脸图片，读取训练模型进行预测。实现流程如下：

* 1.VideoCapture对象获取图像信息。
* 2.使用cv2.cvtColor函数灰度化处理图像信息。
* 3.分类器的detectMultiScale函数检测人脸位置。
* 4.使用人脸位置坐标裁剪出人脸灰度图像。
* 5.处理人脸灰度图像进行预测。
* 6.使用cv2.rectangle函数框出人脸位置，cv2.putText函数写上表情标签。
* 7.检测本地图片差别在于第一步使用cv2.imread读取图像信息。



人脸识别部分：

**1.数据集的获取和处理**

使用摄像头进行人脸捕获和保存为 62x47 的人脸数据作为测试集和训练集，比例为 2:8。

数据集读入，读入用户名和图片信息，确定识别人脸的数量，用于最后的全连接层。

def collect\_dataset():

'''

@brief 提去相片信息，并贴上标签

:return:

'''

people = [person for person in os.listdir("../people/")]

counter = 0

labels\_dic = {}

images = []

labels = []

for i, person in enumerate(people):

labels\_dic[i] = person

for image in os.listdir("../people/" + person):

if image.endswith('.jpg'):

images.append(cv2.imread("../people/" + person + '/' + image, 0))

labels.append(i)

counter += 1

return images, np.array(labels), labels\_dic, counter

数据集的处理，将标签转化为向量 one-hot 形式。

images, labels, labels\_dic, counter = collect\_dataset()

train = np.asarray(images)

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(train, labels, test\_size=0.2, random\_state=random.randint(0, 100))

*# 格式化和标准化*

X\_train = X\_train.astype('float32')

X\_test = X\_test.astype('float32')

X\_train = X\_train.reshape(len(X\_train), 62, 47, 1) / 255

X\_test = X\_test.reshape(len(X\_test), 62, 47, 1) / 255

*# 将标签转化为二进制分类矩阵*

Y\_train = np\_utils.to\_categorical(Y\_train)

Y\_test = np\_utils.to\_categorical(Y\_test)

数据集的获取，主要代码如下所示。用户按下键盘 c 键，摄像头开始捕获人脸数据，将人脸数据处理后保存在对应的人名目录下。保存的图片格式为 jpg 格式，大小为 62x47。

while True:

ret, frame = cam.read()

faces\_coord = detect\_face(frame)

if len(faces\_coord):

faces = normalize\_faces(frame, faces\_coord)

cv2.imwrite(folder + '/' + str(sample) + '.jpg', faces[0])

if flag\_start\_capturing:

sample += 1

draw\_rectangle(frame, faces\_coord)

cv2.imshow('Face', frame)

keypress = cv2.waitKey(1)

if keypress == ord('c'):

if not flag\_start\_capturing:

flag\_start\_capturing = True

if sample > 150:

break

**2.搭建训练模型**

搭建 CNN 神经网络模型，主要代码如下：

*# 建立一个基于 CNN 的人脸识别模型*

model = Sequential()

*#建立一个CNN模型，一层卷积、一层池化、一层卷积、一层池化、抹平之后进行全链接、最后进行分类*

*# 卷积层*

model.add(Convolution2D(filters=32, kernel\_size=(3, 3), padding='same', input\_shape=X\_train.shape[1:]))

*# 激活层*

model.add(Activation('relu'))

*# 最大池化层*

model.add(MaxPooling2D(pool\_size=(2, 2), strides=(2, 2), padding='same'))

model.add(

Convolution2D(filters=64, kernel\_size=(3, 3), padding='same')

)

model.add(Activation('relu'))

model.add(

MaxPooling2D(pool\_size=(2, 2), strides=(2, 2), padding='same')

)

model.add(Flatten())

model.add(Dense(256))

model.add(Activation('relu'))

model.add(Dense(counter))

model.add(Activation('softmax'))

model.summary()

*# training model*

model.compile(optimizer='sgd', loss='mean\_squared\_error', metrics=['accuracy'])

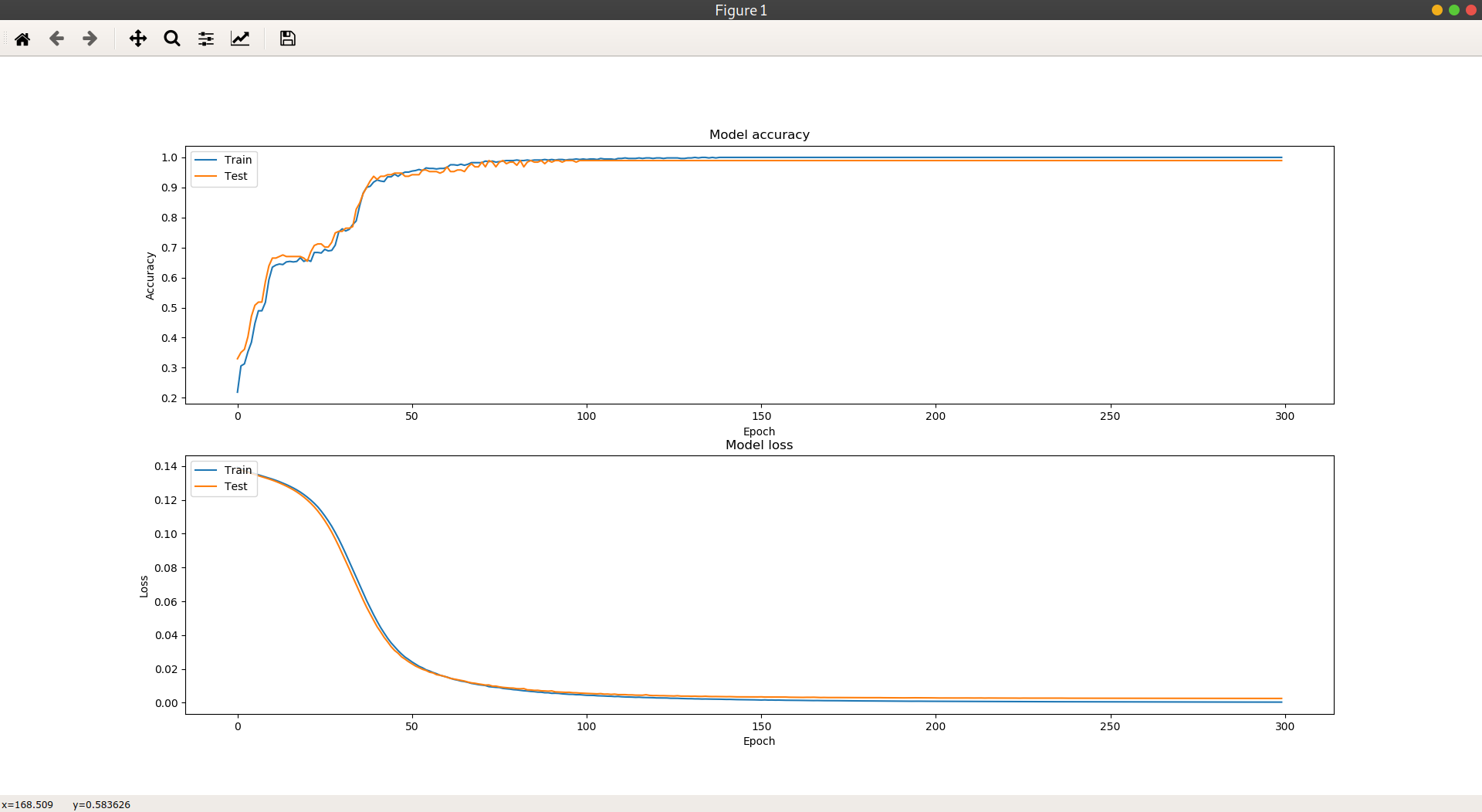
*# model.fit(X\_train, Y\_train, epochs=1, batch\_size=50)*

history = model.fit(X\_train, Y\_train, validation\_split=0.25, epochs=300, batch\_size=32, verbose=1)

filepath = '../model/cnnmodel.h5'

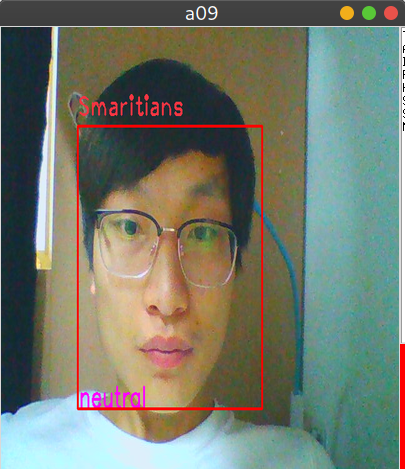
model.save(filepath)

将训练结果进行可视化处理，如下图所示：

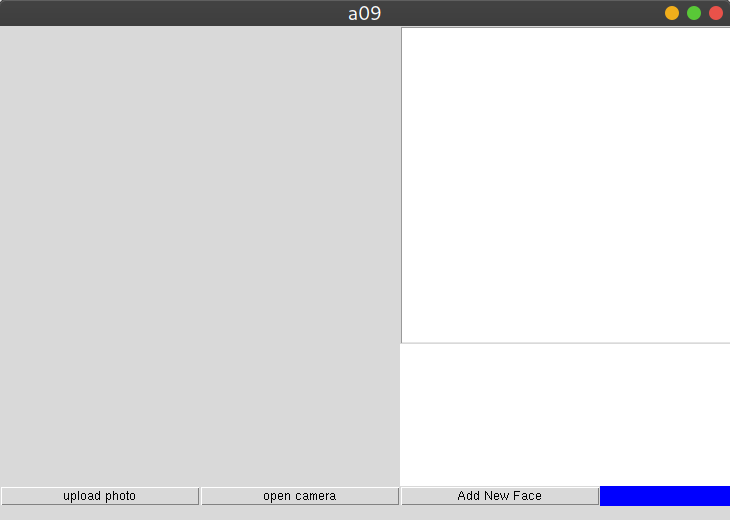


**3.预测**

预测结果如下所示，右侧是通过传入图片进行识别的结果，左侧是通过摄像头进行实时识别的结果。



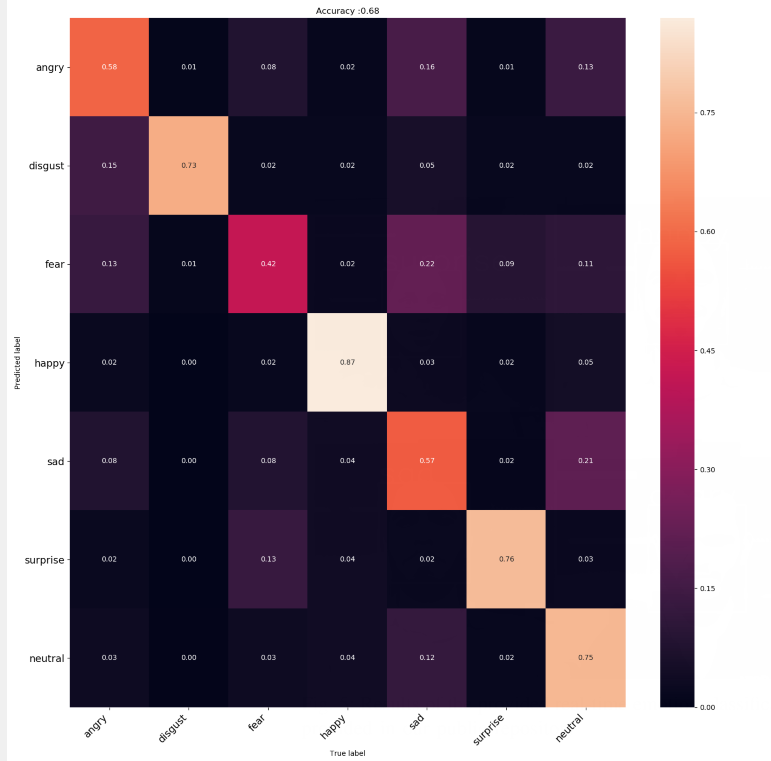
界面实现部分：



使用 tkinter 进行界面的实现，设计的界面较为简陋，代码也比较简单。主要分为三个部分，上传图片显示的部分（左侧），结果打印的部分（右侧）， 和功能按键部分（底部）。其效果

# 结果分析

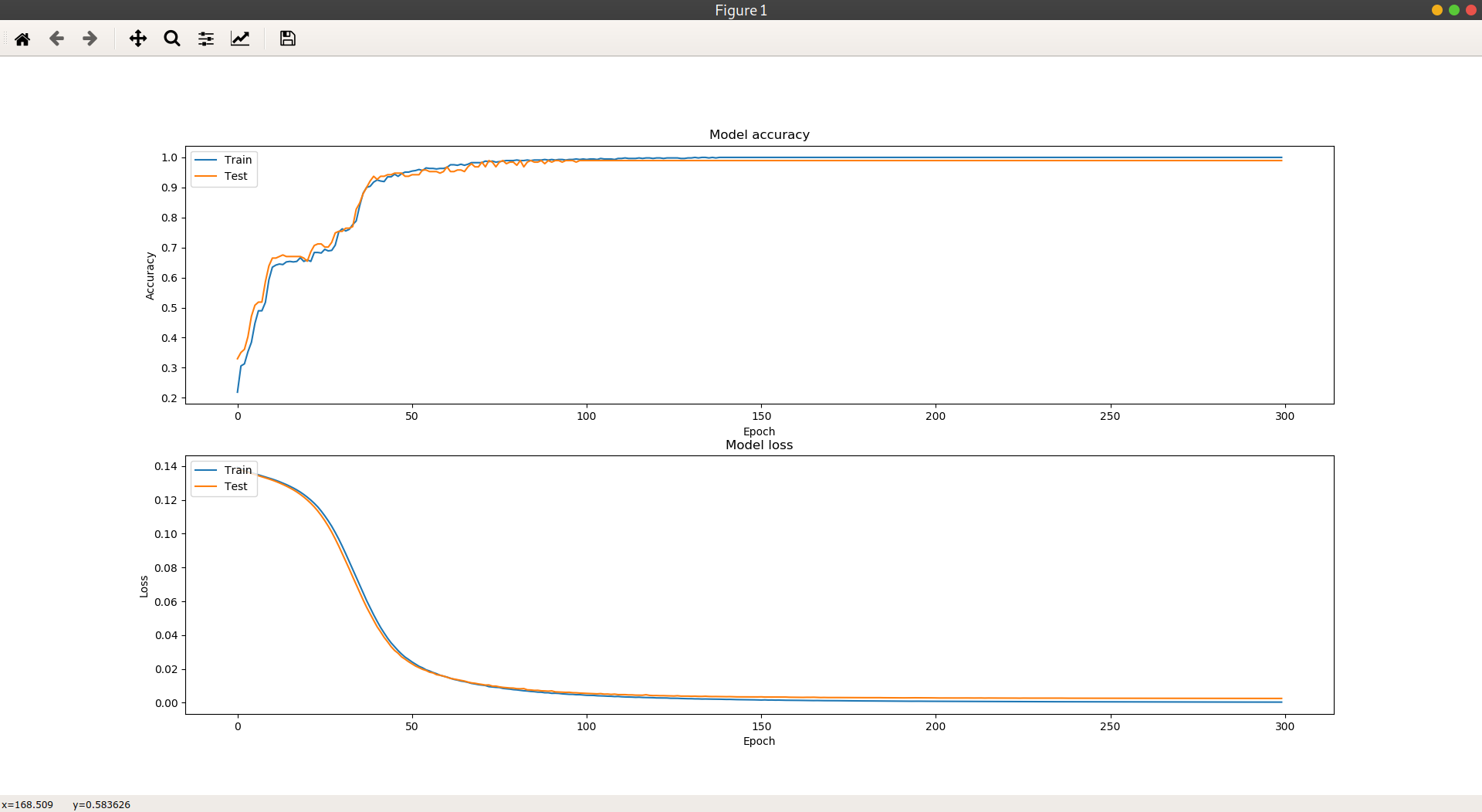
情绪识别部分：



使用训练模型对测试集进行预测画出混淆矩阵，测试集准确率为68%，通过矩阵可以看出该训练模型预测准确率从低到高依次为开心、惊讶、中立、厌恶、生气、伤心、害怕。

通过实际测试，开心、惊讶、中立、生气容易识别，厌恶、伤心、害怕容易被错误预测。

人脸识别部分：



人脸识别模型一共训练了 300 epoches, 由上图可以看出训练精度和测试精度在大约50轮后开始逐渐稳定且两者差别非常小，没有出现欠拟和和过拟合问题，在大约100 轮后已经非常稳定了，准确率达到了 99% 。如果录入的数据集越大，其精度的变化抖动越明显，稳定出现的越晚。在录入数据时如果包含人的脸部的各种姿势，低头，偏头，测头，仰头等各个角度，在预测时的正确率就会越高。

# 总结

时间过得真快，转眼间为期六周的实训已经结束。经过这六周的实训练习让我们学到了许多知识，回头想想实训这六周我们确实是有很大收获的。

　　在这六周中我们学到了很多东西，并且接触了从没有学习过的语言和一些企业的消息。这六周我们在学习的同时，也进一步懂得了实习的重要性，实训过程中自己有很多的不懂，很多的问题，都是通过老师或同学的帮助完成的，这次实训让我们明白了实训的主要目的是让我们通过不断的实习来积累经验，进而才能把书本的知识转换为技能。实践出真理，在这六周的实训确实有些累，不知从哪开始入手，但是累的有价值。学海无涯，有很多很多的东西在向我们招手，等待我们去努力的学习。在以后的工作、生活和学习中，发展自己的优势，弥补自己的不足和缺陷。

　　我们兴高采烈的进入到我们实训的计算机机房，打开各自的实训电脑，老师发来我们当天的实训内容。在操作中才知道自己会的只有书本上的知识，到实际操作时什么都不会。自己需要认真听老师的讲解，再结合自己的实践才能慢慢地搞懂，并完成当天的内容。

　　就这样，我们实训了六周，但是收获不小，在实训中改正自己操作中的不足之处，让我知道实际操作并不是那么简单，实际操作是需要牢固的基础知识，两者是不可分开的。

　　和我们一同参与实训的还有胡英铭老师和梁老师，在实训中也分别给予了我们指导，并且做详细的讲解，这次实训活动十分成功，除了各位同学的充分准备，还要感谢学院和泰克教育的各位领导及专业老师。

　　社会的竞争是激烈的，我想我们应该好好把握住大学学习的时间，充实、完善自我，全面发展，攻破电脑，争取做一名出色的现代人! 在此再次感谢学院领导和泰克教育的各位领导及专业老师给我们这次实训的机会，我们会再接再励，更努力的学习!

# 致谢

衷心感谢华为公司能够给我们这么一个宝贵的实习机会，也感谢学校为我们提供了舒适的实习环境，也由衷的感谢实习指导老师的辛勤付出，使我们在学习的过程中进行这次的实训，让我们从人工智能领域的小白成长为能够合作完成人工智能项目的初学者，这次实训让我们初步掌握了python这门新语言，提高了我们对人工智能的认识，实习也因此得以顺利完成。

# 参考文献

[1] Franc¸ois Chollet. Xception: Deep learning with depthwise separable convolutions. CoRR, abs/1610.02357, 2016.

[2] Andrew G. Howard et al. Mobilenets: Efﬁcient convolutional neural networks for mobile vision applications. CoRR, abs/1704.04861, 2017.

[3] Dario Amodei et al. Deep speech 2: End-to-end speech recognition in english and mandarin. CoRR, abs/1512.02595, 2015.

[4] Ian Goodfellow et al. Challenges in Representation Learning: A report on three machine learning contests, 2013.

[5] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectiﬁer neural networks. In Proceedings of the Fourteenth International Conference on Artiﬁcial Intelligence and Statistics, pages 315–323, 2011.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.

[7] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International Conference on Machine Learning, pages 448–456, 2015.

[8] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[9] Rasmus Rothe, Radu Timofte, and Luc Van Gool. Deep expectation of real and apparent age from a single image without facial landmarks. International Journal of Computer Vision (IJCV), July 2016.

[10] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[11] JostTobiasSpringenberg,AlexeyDosovitskiy,ThomasBrox,andMartin Riedmiller. Striving for simplicity: The all convolutional net. arXiv preprint arXiv:1412.6806, 2014.

[12] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2818–2826, 2016.

[13] Yichuan Tang. Deep learning using linear support vector machines. arXiv preprint arXiv:1306.0239, 2013.