# 人脸情绪分析

# 1、需求分析

## 1.1背景

随着计算能力的提升、神经网络研究的步步深入，人工智能在机器翻译、行为识别、语音识别以及图像分割、分类、检测等任务中取得了重大突破，推进了这些领域的发展。配以图像、语音识别的人工智能已经逐渐能够“看见”、“听见”，然而人机交互的过程中，让机器真正看得懂、听得懂情感丰富的人类的情绪状态，依然是一个很大的挑战。

未来想让计算机实现真正的人工智能并服务于人，并与人类产生自然的智能化的人机交互，计算机必须具备情绪识别和表达能力，需要具备情感。情绪识别是与人工智能相关的领域，它可以帮助计算机智能的识别人类情感。随着情绪识别领域的不断发展，人们对情绪识别的研究也越来越多，并在人机交流等不同的应用领域中占有重要的地位 。由于人脸表情是最容易获取且最直观反映人的情绪状态的模式，因此在所有情绪识别研究的分支中，基于人脸表情的情绪识别是最早也是最热门的一个分支。

## 1.2总体要求

### 1.2.1功能需求

1、基于yolov5的人脸检测情绪分析需要先完成目标检测，能够对识别人脸

2、识别人脸之后需要对目标检测所用的坐标点（嘴巴、眼睛、鼻子等）进行分析，从而得出不同的情绪，例如嘴角上扬一般为开心，眉毛皱一般为忧愁

3、需要大量不同情绪的人脸数据集，从而为模型提供数据训练

### 1.2.2性能需求

1、准确率作为评分标准要求训练模型结果识别准确率高

2、训练速度尽可能快，能够处理大量的人脸情绪图片数据集

3、训练完的模型要求能够保存下来，以便后续使用不需要再次进行大量数据集的训练，可以直接使用模型进行识别预测

### 1.2.3可靠性和可用性需求

人脸检测识别情绪项目界面清楚简单，易操作，web端上传图片即可进行分析，可以有效控制相应权限的访问资格，在正常网络下能正常使用，支持网页端使用。

### 1.2.4运行环境需求

AI模块：

需要装有python3、yolov5，还需要在Anaconda环境下安装pytorch，有GPU的可选择GPU训练模型，没有则选择CPU训练模型

Web模块：

Web 前端使用 Vue + ElementUi

后端模块：

后端框架使用SpringBoot + Mybatis进行开发

数据库使用Mysql

# 2、可行性分析

## 2.1内容范畴

该项目可行性研究报告是对在经济、政治、技术、市场、操作等不同视角及因素的研究分析说明，对于项目的内容、经济效益、费用支出、条件限制、最终性能、设备技术关系等多方面统筹概述，在此基础上，保证项目在建设上的必要性，在经济上的合理性，在政治上的合法性、在技术上的先进性以及在社会上的适应性，为项目的开发提供科学依据及基础。

## 2.2项目背景

### 2.2.1技术背景

目前人工智能技术已经发展成熟且实用性大大提升，在各领域都发挥着重要作用，不管是机器学习方法还是深度学习方法，对于模型的训练准确度都达到合格标准；前端技术用到了vue，后端使用springboot，实现了前后端分离

### 2.2.2市场背景

2014年以来，人脸识别技术尝试在各类行业中实现应用落地，行业应用场景拓展，有利于推动人脸识别市场规模增长。使用场景开发将成为人脸识别企业的重点发展方向，交通、零售、医疗等行业在数据积累、AI应用基础方面有一定积累，均为进一步与人脸识别应用结合提供良好的基础。

AI技术持续向传统行业渗透，安防和金融领域与AI结合程度较深。安防领域主要利用AI视觉技术升级安全监控、安全防护等产品。金融行业拥有良好的数据积累基础，数字化改造进程稳定，银行、证券等金融机构较多愿意接受业务智能化尝试，因此AI技术在众多金融业务场景得到中得到推广。交通、零售、医疗、制造等行业在数据积累、AI应用基础方面有一定积累，均为进一步与AI应用结合提供良好的基础。

### 2.2.3政治背景

国家政策，十四五规划国家强调大力发展人工智能技术，争取早日实现科技强国，成为具备强大创新创造力的富强民主文明和谐的社会主义现代化强国。

## 2.3不同视觉的可行性分析

### 2.3.1技术可行性

AI技术方面计算机视觉识别技术已经发展成熟，基于此技术上实现分类可选机器学习的分类方法或由已标注训练集通过神经网络自动训练模型，前端技术使用vue，后端使用springboot，以上技术均已成熟，具有技术可行性。

### 2.3.2操作可行性

该项目写有前端交互页面，用户操作容易，具有操作可行性。

### 2.3.3市场可行性

人脸情绪分析目前市场还算较少人接触，应用前景及潜力十足，可在医疗、心理、治安等业务场景发挥重大作用，固市场需求存在，现阶段市场现有的技术尚未大力推广，此项目具备市场可行性。

### 2.3.4经济可行性

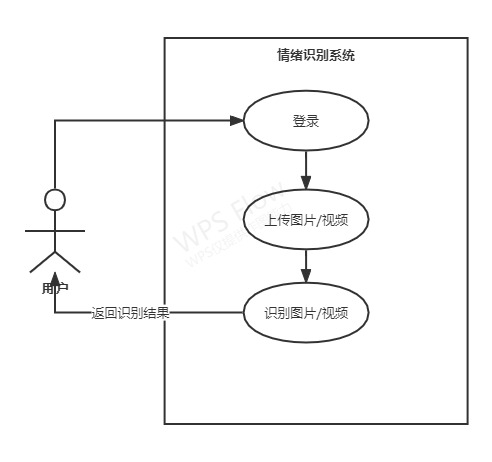
该项目应用前景广阔，可在众多业务场景下发挥重大作用，项目实施落地后与其他领域相结合，将产生巨大的经济效益，具有经济可行性。

### 2.3.5法律可行性

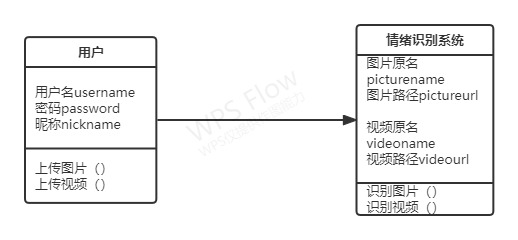
结合AI技术实现的人脸情绪分析，符合国家政策大力发展的人工智能，没有触犯国家法律，在政策上得到国家的支持，具有法律可行性。

# 3、UML统一建模语言设计

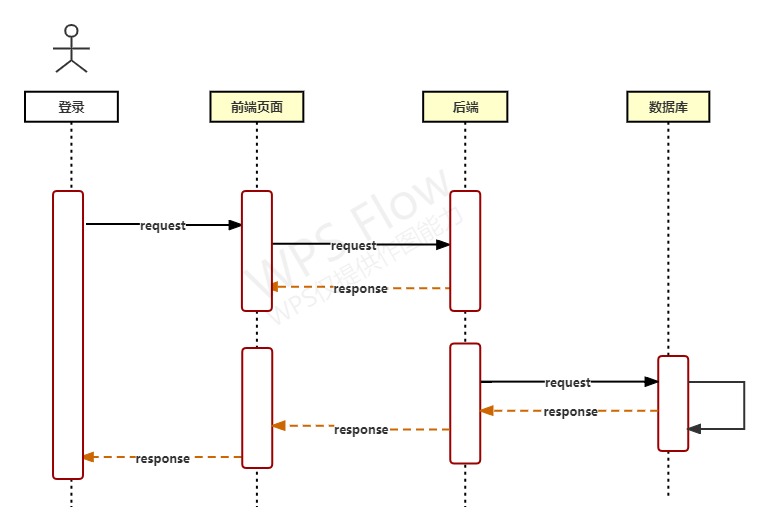
## 3.1用例图



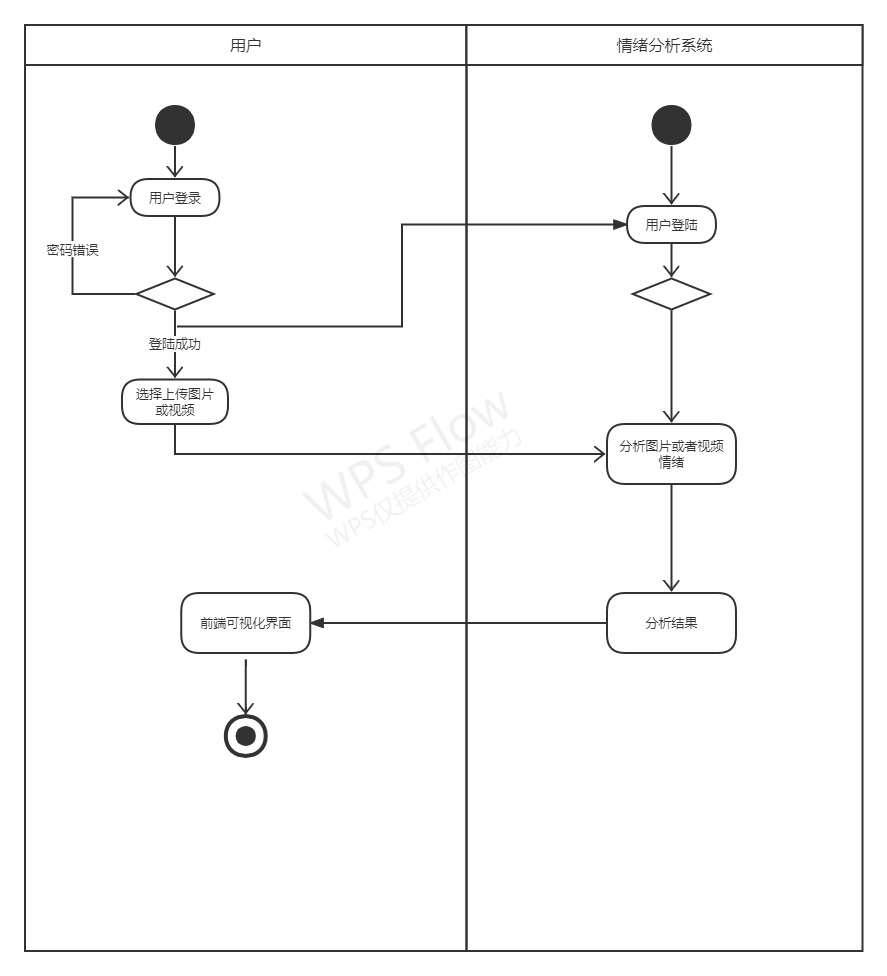
## 3.2类图



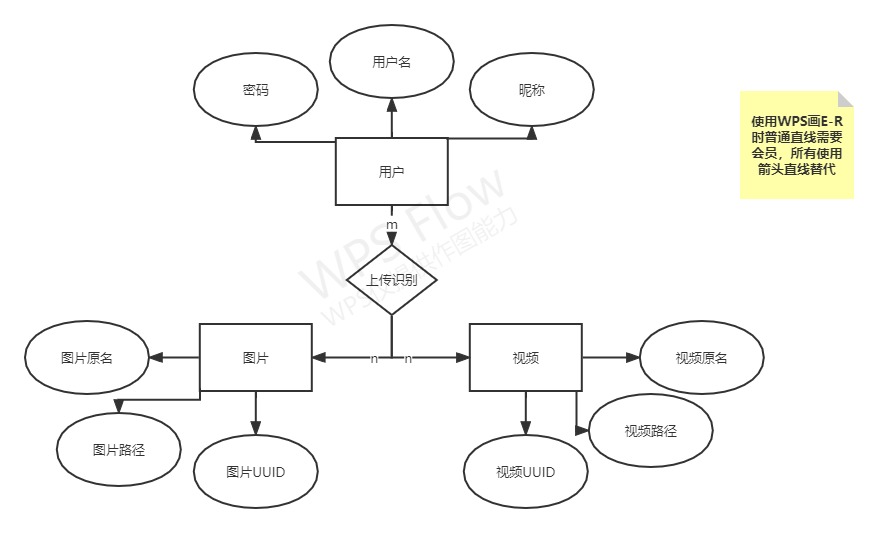
## 3.3时序图



## 3.4活动图



## 3.5 E-R图



# 4、数据库和接口（后端任务文档）

## 数据库

### 图片信息表user

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 允许为null | 默认值 | 备注 |
| id | id | Int(11) | 不允许 |  | 自增 |
| 用户名 | username | varchar(256) | 允许 |  |  |
| 密码 | password | varchar(256) | 允许 |  |  |
| 昵称 | nickname | Varchar(256) | 允许 |  |  |

### 视频信息表Uploadvideo

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 允许为null | 默认值 | 备注 |
| ID | id | Int(16) | 不允许 |  | 自增 |
| 视频原名 | videoname | varchar(256) | 允许 |  |  |
| 视频路径 | videourl | varchar(256) | 允许 |  |  |
| 视频UUID | videurl | Varchar(256) | 允许 |  |  |

### 图片信息表Uploadpicture

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 允许为null | 默认值 | 备注 |
| ID | id | Int(16) | 不允许 |  | 自增 |
| 图片原名 | picturename | varchar(256) | 允许 |  |  |
| 图片路径 | pictureurl | varchar(256) | 允许 |  |  |
| 图片UUID | pictureurl | Varchar(256) | 允许 |  |  |

## 接口

### 1.上传图片接口uploadPicture

* 接口描述：接收前端发送过来的图片，保存到指定文件夹下，然后进行识别，同时将识别结果保存下来
* 接口地址：http://localhost:8083/api/uploadPicture
* 请求类型：post
* 输入参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 地址路径 | SavePath | str | 前台传来的地址路径，也是用来后端保存在服务器哪个文件夹的地址 |
| 图片 | file | file |  |

* 输出参数：（输出内容为包含以下字段的json列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 状态 | resCode | int | 成功为200，失败为400 |
| 图片新名 | newPictureName | str | 图片的uuid |
| 图片路径 | pictureUrl | str |  |

### 2.上传视频接口uploadVideo

* 接口描述：接收前端发送过来的视频，保存到指定文件夹下，然后进行识别，同时将识别结果保存下来
* 接口地址：http://localhost:8083/api/uploadVideo
* 请求类型：post
* 输入参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 地址路径 | SavePath | str | 前台传来的地址路径，也是用来后端保存在服务器哪个文件夹的地址 |
| 视频 | file | file |  |

* 输出参数：（输出内容为包含以下字段的json列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 状态 | resCode | int | 成功为200，失败为400 |
| 视频新名 | newVideoName | str | 视频的uuid |
| 视频路径 | videoUrl | str |  |

### 3.删除图片接口deletePicture

* 接口描述：接收前端发送过来的图片id，删除数据库的图片信息并返回给前端
* 接口地址：http://localhost:8083/DeletePicture/{pictureId}
* 请求类型：get
* 输入参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 图片id | pictureId | int |  |

* 输出参数：（输出内容为包含以下字段的json列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 状态 | resCode | int | 成功为200，失败为400 |
| 图片信息 | pictureList | json | 用PictureUpload对象封装包含id,pitureName,pictureUrl,pictureUUID |

### 4.删除视频接口deleteVideo

* 接口描述：接收前端发送过来的视频id，删除数据库的图片信息并返回给前端
* 接口地址：http://localhost:8083/DeleteVideo/{videoId}
* 请求类型：get
* 输入参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 视频id | videoId | int |  |

* 输出参数：（输出内容为包含以下字段的json列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 状态 | resCode | int | 成功为200，失败为400 |

### 5.查询视频列表接口videoTable

* 接口描述：接收前端发送过来请求，返回视频列表信息
* 接口地址：http://localhost:8083/SelectVideo/table
* 请求类型：get
* 输入参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 无 | 无 | 无 |  |

* 输出参数：（输出内容为包含以下字段的json列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 状态 | resCode | int | 成功为200，失败为400 |
| 视频信息 | videoList | json | 用VideoUpload对象封装包含id,videoName,videoUrl,videoUUID |

### 6.查询图片列表接口pictureTable

* 接口描述：接收前端发送过来请求，返回图片列表信息
* 接口地址：http://localhost:8083/SelectPicture/table
* 请求类型：get
* 输入参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 无 | 无 | 无 |  |

* 输出参数：（输出内容为包含以下字段的json列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 状态 | resCode | int | 成功为200，失败为400 |
| 图片信息 | pictureList | json | 用PictureUpload对象封装包含id,pitureName,pictureUrl,pictureUUID |

### 7.输出图片接口picturePreview

* 接口描述：接收前端发送过来的图片id，返回指定文件夹内的指定图片
* 接口地址：[http://localhost:8083/SelectPicture/policemen/{pictureId}](http://localhost:8083/TestPicture/%7bvideoId%7d)
* 请求类型：get
* 输入参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 图片id | pictureId | int |  |

* 输出参数：（输出内容为包含以下字段的json列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 状态 | resCode | int | 成功为200，失败为400 |
| 图片 | Img | file |  |

### 8.输出视频接口videoPreview

* 接口描述：接收前端发送过来的视频id，返回指定文件夹内的指定视频
* 接口地址：[http://localhost:8083/SelectVideo/policemen/{videoId}](http://localhost:8083/TestPicture/%7bvideoId%7d)
* 请求类型：get
* 输入参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 图片id | videoId | int |  |
|  |  |  |  |

* 输出参数：（输出内容为包含以下字段的json列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 状态 | resCode | int | 成功为200，失败为400 |
| 视频 | video | file |  |

### 9.测试图片情绪接口testPicture

* 接口描述：接收前端发送过来的图片id,输出图片识别的结果
* 接口地址：[http://localhost:8083/SelectPicture/TestPicture/{pictureId}](http://localhost:8083/TestPicture/%7bvideoId%7d)
* 请求类型：get
* 输入参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 图片id | pictureId | int |  |
|  |  |  |  |

* 输出参数：（输出内容为包含以下字段的json列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| 状态 | resCode | int | 成功为200，失败为400 |
| 情绪信息 | pictureCharacterEmotion | json | 接口调用模型返回结果 |
|  |  |  |  |

### 10.登录接口

* 接口描述：接受前端传输的账号密码与数据库信息比对，返回result对象包含用户信息和token。
* 接口地址：[http://localhost:8083/user /login](http://localhost:8083/user%20/login)
* 请求类型：post
* 输入参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| username | 用户名 | String |  |
| Password | 密码 | String |  |

* 输出参数：（输出内容为包含以下字段的json列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| code | 状态码 | String | 200成功；400参数错误  401权限不足；500系统错误；600其他业务异常； |
| msg | 信息 | String |  |
| username | 用户名 | int |  |
| password | 密码 | String |  |
| nickname | 昵称 | String |  |
| token | 令牌 | String | 时效两个小时 |

### 11.注册接口

* 接口描述：接受前端传输的账号密码存储在数据库中，返回result结果包含注册状态。
* 接口地址：[http://localhost:8083/user /register](http://localhost:8083/user%20/register)
* 请求类型：post
* 输入参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| username | 用户名 | String |  |
| password | 密码 | String |  |

* 输出参数：（输出内容为包含以下字段的json列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| code | 状态码 | String | 200成功；400参数错误  401权限不足；500系统错误；600其他业务异常； |
| msg | 信息 | String |  |

### 12.修改用户信息接口

* 接口描述：接受前端传输的账号密码存储在数据库中，返回result结果包含注册状态。
* 接口地址：[http://localhost:8083/user /{user}](http://localhost:8083/user%20/%7buser%7d)
* 请求类型：post
* 输入参数：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| user | 用户信息 | Json |  |

* 输出参数：（输出内容为包含以下字段的json列表）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 变量名 | 类型 | 说明 |
| code | 状态码 | String | 200成功；400参数错误  401权限不足；500系统错误；600其他业务异常； |
| msg | 信息 | String |  |

# 5、AI\_API

**!cd yolov5 && python3 train.py --img 320 --batch 16 --epochs 100 --data dataset.yaml --weights yolov5m.pt**

在项目根目录进入到yolov5的文件夹，运行训练的python文件，图像大小为320，批量为16，迭代次数为100，数据集是dataset.yaml，使用yolov5m的模型进行训练。

--weights yolov5m.pt通过指定数据集、批量大小、图像大小以及预训练（推荐）或随机初始化--weights '' --cfg yolov5s.yaml（不推荐） ，在自定义数据集上训练 YOLOv5m 模型。

运行下面的命令训练dataset.ymal，可以有两种训练方式

--cfg yolov5s.yaml --weights ''：从头开始训练

--cfg yolov5s.yaml --weights yolov5s.pt：从预训练的模型加载开始训练

dataset.yaml结构如下

**# 训练/验证/测试集合表示 1) 根目录: path/to/imgs, 2) 文件: path/to/imgs.txt, or 3) 清单: [path/to/imgs1, path/to/imgs2, ..]**

**path: ../data/train # dataset root dir数据集的根目录**

**train: images # train images (relative to 'path') 128 images 训练集的相对路径**

**val: images # val images (relative to 'path') 128 images验证集的图片**

**test: # 可选的测试集合**

**# 类别**

**nc: 2 # number of classes类别总量**

**names: ['happy','sad'] # class names 类别昵称**

YOLOv5在coco128上训练5epochs的命令：

**python train.py --img 640 --batch 16 --epochs 5 --data ./data/coco128.yaml --cfg ./model yoyov5s.yaml --weights**

训练的更多可选参数：

--epochs：训练的epoch，默认值300

--batch-size：默认值16

--cfg：模型的配置文件，默认为yolov5s.yaml

--data：数据集的配置文件，默认为data/coco128.yaml

--img-size：训练和测试输入大小，默认为[640, 640]

--rect：rectangular training，布尔值

--resume：是否从最新的last.pt中恢复训练，布尔值

--nosave：仅仅保存最后的checkpoint，布尔值

--notest：仅仅在最后的epoch上测试，布尔值

--evolve：进化超参数（evolve hyperparameters），布尔值

--bucket：gsutil bucket，默认值''

--cache-images：缓存图片可以更快的开始训练，布尔值

--weights：初始化参数路径，默认值''

--name：如果提供，将results.txt重命名为results\_name.txt

--device：cuda设备，例如：0或0,1,2,3或cpu，默认''

**torch.hub.load(github, model, force\_reload=False, \*args, \*\*kwargs)**

参数:

* github – 必需，一个字符串对象，格式为“repo\_owner/repo\_name[:tag\_name]”，可选 tag/branch。如果未做指定，默认的 branch 是 master 。比方说: 'pytorch/vision[:hub]'
* model – 必须，一个字符串对象，名字在hubconf.py中定义。
* force\_reload – 可选， 是否丢弃现有缓存并强制重新下载。默认是：False。
* \*args – 可选， 可调用的model的相关args参数。
* \*\*kwargs – 可选， 可调用的model的相关kwargs参数。

|  |  |
| --- | --- |
| **返回:** | **一个有相关预训练权重的单一模型。** |

**torch.hub.set\_dir(d)**

也可以将hub\_dir设置为本地目录来保存中间模型和检查点文件。

如果未设置此参数,环境变量TORCH\_HUB\_DIR 会被首先搜寻，~/.torch/hub 将被创建并用作后备。

**os.path.join(path, \*paths)**

智能地拼接一个或多个路径部分。

返回值是 path 和 \*paths 的所有成员的拼接，其中每个非空部分后面都紧跟一个目录分隔符，最后一个部分除外，这意味着如果最后一个部分为空，则结果将以分隔符结尾。如果某个部分为绝对路径，则之前的所有部分会被丢弃并从绝对路径部分开始继续拼接。

在 Windows 上，遇到绝对路径部分（例如 r'\foo'）时，不会重置盘符。如果某部分路径包含盘符，则会丢弃所有先前的部分，并重置盘符。请注意，由于每个驱动器都有一个“当前目录”，所以 os.path.join("c:", "foo") 表示驱动器 C: 上当前目录的相对路径 (c:foo)，而不是 c:\foo。

**os.path.relpath(path[, start])**

返回从当前目录或 start 目录（可选）到达 path 之间要经过的相对路径。这仅仅是对路径的计算，不会访问文件系统来确认 path 或 start 的存在性或属性。

开始 默认为 [**os.curdir**](file:///D:\Google%20Chrome下载\WPS%20Office\11.1.0.10700\.541438054\cachedata\6AA17B773764480DADF4C3B50E623103\AI_API.docx#os.curdir)

可用性：Windows、Unix。

**os.getenv(key, default=None)**

如果存在则返回环境变量键的值，如果不存在则 返回默认值。键，默认值和结果是 str。请注意，由于uses ，映射也同样在导入时被捕获，并且该函数可能不会反映未来的环境变化。[getenv()](file:///D:\\Google%20Chrome下载\\WPS%20Office\\11.1.0.10700\\.541438054\\cachedata\\6AA17B773764480DADF4C3B50E623103\\AI_API.docx" \l "os.getenv" \o "os.getenv)[os.environ](file:///D:\Google%20Chrome下载\WPS%20Office\11.1.0.10700\.541438054\cachedata\6AA17B773764480DADF4C3B50E623103\AI_API.docx#os.environ)[getenv()](file:///D:\Google%20Chrome下载\WPS%20Office\11.1.0.10700\.541438054\cachedata\6AA17B773764480DADF4C3B50E623103\AI_API.docx#os.getenv)

在Unix系统上，键和值会使用 [sys.getfilesystemencoding()](file:///D:\Google%20Chrome下载\WPS%20Office\11.1.0.10700\.541438054\cachedata\6AA17B773764480DADF4C3B50E623103\AI_API.docx#sys.getfilesystemencoding) 和``'surrogateescape'`` 错误处理进行解码。如果你想使用其他的编码，使用 。[os.getenvb()](file:///D:\Google%20Chrome下载\WPS%20Office\11.1.0.10700\.541438054\cachedata\6AA17B773764480DADF4C3B50E623103\AI_API.docx#os.getenvb)

[可用性](file:///D:\Google%20Chrome下载\WPS%20Office\11.1.0.10700\.541438054\cachedata\6AA17B773764480DADF4C3B50E623103\AI_API.docx#availability): 大部分的Unix系统，Windows。

**numpy.squeeze(a, axis=None)**

从a中删除长度为 1 的轴。

参数:

* a – 必选，类似数组的，输入数据。
* axis – 可选，无或整数或整数元组。

选择形状中长度为 1 的条目的子集。如果选择了形状条目大于 1 的轴，则会引发错误。

|  |  |
| --- | --- |
| **返回** | **Squeezed：ndarray** |

输入数组，但删除了长度为 1 的所有维度或维度的子集。这始终是 a本身或进入a的视图。请注意，如果所有轴都被挤压，则结果是 0d 数组而不是标量。

|  |  |
| --- | --- |
| **注意** | **ValueError** |

如果轴不是无，并且被挤压的轴的长度不是 1

**cv2.VideoCapture()**

从视频文件或摄像机中捕获视频的类。

**cv2.VideoCapture() → <VideoCapture object>**

**cv2.VideoCapture(filename) → <VideoCapture object>**

**cv2.VideoCapture(device) → <VideoCapture object>**

参数:

* **filename**- 打开的视频文件的名称
* **device** – 打开的视频捕获设备的 id（即相机索引）。如果连接了单个摄像头，则只需传递 0。

**VideoCapture::open**[**¶**](file:///D:\Google%20Chrome下载\WPS%20Office\11.1.0.10700\.541438054\cachedata\6AA17B773764480DADF4C3B50E623103\AI_API.docx#videocapture-open)

参数：

* **filename**- 打开的视频文件的名称
* **device** – 打开的视频捕获设备的 id（即相机索引）。

这些方法首先调用**VideoCapture::release()**来关闭已经打开的文件或相机。

**VideoCapture::isOpened**

如果视频捕获已经初始化，则返回 true。

cv2.VideoCapture.isOpened ( )

如果先前对VideoCapture构造函数或VideoCapture::open的调用成功，则该方法返回 true。

**VideoCapture::release**

关闭视频文件或捕获设备。

cv2.VideoCapture.release()

后续VideoCapture::open()和VideoCapture析构函数会自动调用这些方法。

C 函数还释放内存并清除\*capture指针。

**VideoCapture::grab**

从视频文件或捕获设备中抓取下一帧。

cv2.VideoCapture.grab() → successFlag

cv.GrabFrame(capture) → int

方法/函数从视频文件或相机中抓取下一帧，并在成功的情况下返回真（非零）。

该功能的主要用途是在多摄像头环境中，尤其是在摄像头没有硬件同步的情况下。也就是说，您为每个摄像头调用VideoCapture::grab()，然后调用较慢的方法VideoCapture::retrieve()来解码并从每个摄像头获取帧。这样就消除了去马赛克或运动 jpeg 解压缩等的开销，并且从不同相机检索到的帧将在时间上更接近。

此外，当连接的摄像头是多头（例如，立体摄像头或 Kinect 设备）时，从中检索数据的正确方法是先调用VideoCapture::grab，然后再调用VideoCapture::retrieve()一个或更多次使用不同的通道参数值。见<https://code.ros.org/svn/opencv/trunk/opencv/samples/cpp/kinect_maps.cpp>

**VideoCapture::retrieve**

解码并返回抓取的视频帧。

cv2.VideoCapture.retrieve([image[, channel]]) → successFlag, image

cv.RetrieveFrame(capture) → iplimage

方法/函数解码并重新处理刚刚抓取的帧。如果没有抓取到任何帧（相机已断开连接，或者视频文件中没有更多帧），则方法返回 false 并且函数返回 NULL 指针。Note

**videoCapture::get**

返回指定的VideoCapture属性。

cv2.VideoCapture.get(propId) → retval

cv.GetCaptureProperty(capture, propId) → double

**参数：**

|  |  |
| --- | --- |
|  | propId –属性标识符。它可以是以下之一：   * CV\_CAP\_PROP\_POS\_MSEC视频文件的当前位置（以毫秒为单位）或视频捕获时间戳。 * CV\_CAP\_PROP\_POS\_FRAMES下一个要解码/捕获的帧的基于 0 的索引。 * CV\_CAP\_PROP\_POS\_AVI\_RATIO视频文件的相对位置：0 - 影片开始，1 - 影片结束。 * CV\_CAP\_PROP\_FRAME\_WIDTH视频流中帧的宽度。 * CV\_CAP\_PROP\_FRAME\_HEIGHT视频流中帧的高度。 * CV\_CAP\_PROP\_FPS帧速率。 * CV\_CAP\_PROP\_FOURCC编解码器的 4 字符代码。 * CV\_CAP\_PROP\_FRAME\_COUNT视频文件中的帧数。 * CV\_CAP\_PROP\_FORMAT由retrieve()返回的 Mat 对象的格式。 * CV\_CAP\_PROP\_MODE表示当前捕获模式的后端特定值。 * CV\_CAP\_PROP\_BRIGHTNESS图像的亮度（仅适用于相机）。 * CV\_CAP\_PROP\_CONTRAST图像对比度（仅适用于相机）。 * CV\_CAP\_PROP\_SATURATION图像的饱和度（仅适用于相机）。 * CV\_CAP\_PROP\_HUE图像的色调（仅适用于相机）。 * CV\_CAP\_PROP\_GAIN图像增益（仅适用于相机）。 * CV\_CAP\_PROP\_EXPOSURE曝光（仅适用于相机）。 * CV\_CAP\_PROP\_CONVERT\_RGB布尔标志，指示图像是否应转换为 RGB。 * CV\_CAP\_PROP\_WHITE\_BALANCE目前不支持 * CV\_CAP\_PROP\_RECTIFICATION立体相机的校正标志（注意：目前仅支持 DC1394 v 2.x 后端） |

注意：当查询VideoCapture类使用的后端不支持的属性时，返回值 0。

**VideoCapture::set**

在VideoCapture中设置一个属性。

cv2.VideoCapture.set(propId, value) → retval

cv.SetCaptureProperty(capture, propId, value) → None

参数：

|  |  |
| --- | --- |
|  | * propId –属性标识符。它可以是以下之一： * CV\_CAP\_PROP\_POS\_MSEC视频文件的当前位置，以毫秒为单位。 * CV\_CAP\_PROP\_POS\_FRAMES下一个要解码/捕获的帧的基于 0 的 索引。 * CV\_CAP\_PROP\_POS\_AVI\_RATIO视频文件的相对位置：0 - 影片开始，1 - 影片结束。 * CV\_CAP\_PROP\_FRAME\_WIDTH视频流中帧的宽度。 * CV\_CAP\_PROP\_FRAME\_HEIGHT视频流中帧的高度。 * CV\_CAP\_PROP\_FPS帧速率。 * CV\_CAP\_PROP\_FOURCC编解码器的 4 字符代码。 * CV\_CAP\_PROP\_FRAME\_COUNT视频文件中的帧数。 * CV\_CAP\_PROP\_FORMAT由retrieve()返回的 Mat 对象的格式。 * CV\_CAP\_PROP\_MODE表示当前捕获模式的后端特定值。 * CV\_CAP\_PROP\_BRIGHTNESS图像的亮度（仅适用于相机）。 * CV\_CAP\_PROP\_CONTRAST图像对比度（仅适用于相机）。 * CV\_CAP\_PROP\_SATURATION图像的饱和度（仅适用于相机）。 * CV\_CAP\_PROP\_HUE图像的色调（仅适用于相机）。 * CV\_CAP\_PROP\_GAIN图像增益（仅适用于相机）。 * CV\_CAP\_PROP\_EXPOSURE曝光（仅适用于相机）。 * CV\_CAP\_PROP\_CONVERT\_RGB布尔标志，指示图像是否应转换为 RGB。 * CV\_CAP\_PROP\_WHITE\_BALANCE目前不支持 * CV\_CAP\_PROP\_RECTIFICATION立体相机的校正标志（注意：目前仅支持 DC1394 v 2.x 后端） * value – 属性的值。 |
|  |  |

# 6、不足与展望

目前我们此项目的人脸情绪分析只能实现简单的开心happy和伤心sad情绪的分析，还有许许多多其他情绪待实现，比如愤怒、惊恐、尴尬等等，后续可以扩大训练数据集，以实现更多情绪分析识别。

此人脸情绪分析项目可适用性推广性很有前景，比如心理咨询时可先判断来访者的心理状态；可以使用该系统分析学校里学生的异常情绪，从而相应的可以采取预防措施。

|  |  |
| --- | --- |
|  |  |