

# 基于人脸识别的课堂签到系统

学生：徐翔 胡越 白杨 华天羽 李才瀚    指导教师：牛云云  
(信息工程学院)

**【摘要】** 本项目开发的基于人脸识别的课堂签到系统，定位为为教学工作者服务的课堂考勤解决方案。整个系统分为人脸识别模块，Web 后端模块和 Web 用户界面模块三大部分。人脸识别模块和 Web 后端模块之间以 MySQL 数据库连接。人脸识别和 Web 后端使用 python3.6 作为基础语言。Web 前端使用 Bootstrap4 作为框架。数据库系统为 MySQL5.7。人脸识别模块的核心功能分为视频图片截取，人脸检测，人脸特征识别，人脸检索四部分。视频图片截取的实现使用了 OpenCV 库；人脸检测，人脸特征识别，人脸检索三部分暂时使用了 Face++ 平台提供的 API。Web 后端模块的主要任务是将人脸识别模块的结果转成 json 格式的数据，通过服务器提供给 Web 前端页面。Web 前端页面的设计遵循了“简洁高效”的原则，便于用户查看具体信息。

**【关键词】** 人脸识别、课堂签到、Face++

**【项目编号】** A167

## 1.项目背景及研究现状

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。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，进而对检测到的人脸进行脸部的一系列相关技术，通常也叫做人像识别、面部识别。

近年来由于计算能力的增长和大量数据的产生，人脸识别技术目前已经有相当高的识别率，一些商业公司的模型的识别率已经很大程度地超过人工识别。人脸识别技术如今已经广泛地应用于电子商务、安全防务等领域。

传统的课堂签到是由教职人员点名。这种方式耗时长，人力成本大，遗漏率高。

我校一些使用了新技术的课堂让学生使用手机登录 web 客户端签到。这种方式与传统的方式相比，节省了老师的人力成本，但是依然存在遗漏的情况。

如果将人脸识别技术使用于课堂签到等等场景的身份识别，能够很大程度节省人力和识别需要花费的时间，并且提高准确度。

2017 年的秋季入学，北京高校中已经有如北京师范大学、北京科技大学等，使用人脸识别技术来完成新生注册，识别率在 90% 以上。在人脸识别应用于课堂签到方面，中国传媒大学的沈浩教授作过个人的在 ios 系统上搭建客户端的尝试。他的实现主要基于百度的开放 ai 平台，项目周期大概为两周。

总得来说，目前这项技术在高校的身份识别应用场景中只是实验性的，不普遍的。并且可以看出，基本上使用的是 (1: N) 或者小规模 (M: N) 的识别模式，大规模的 (M:N) 识别并没有现成的实现案例。

## 2.需求及功能设计

### 2.1 需求分析

我们需要的是一个可以完成签到功能的软件，而这个签到功能有以下几种实现方式：

(1) 使用教室摄像头，使用在监控中识别多个（规模达 100 人左右）人脸信息。(2) 使用单独摄像头，对学生小规模（3-5 人）或逐个识别。

在调查现有的技术资料之后，我们发现方法(1)无法找到较成熟的可以复现的方案，因此选择了方法二来完成此项目。

因此，需求定义为：在使用单独摄像头的情况下，实现小规模（5 人以下）场景的信息识别。

### 2.2 功能设计

我们对该软件的功能的设计如下图：

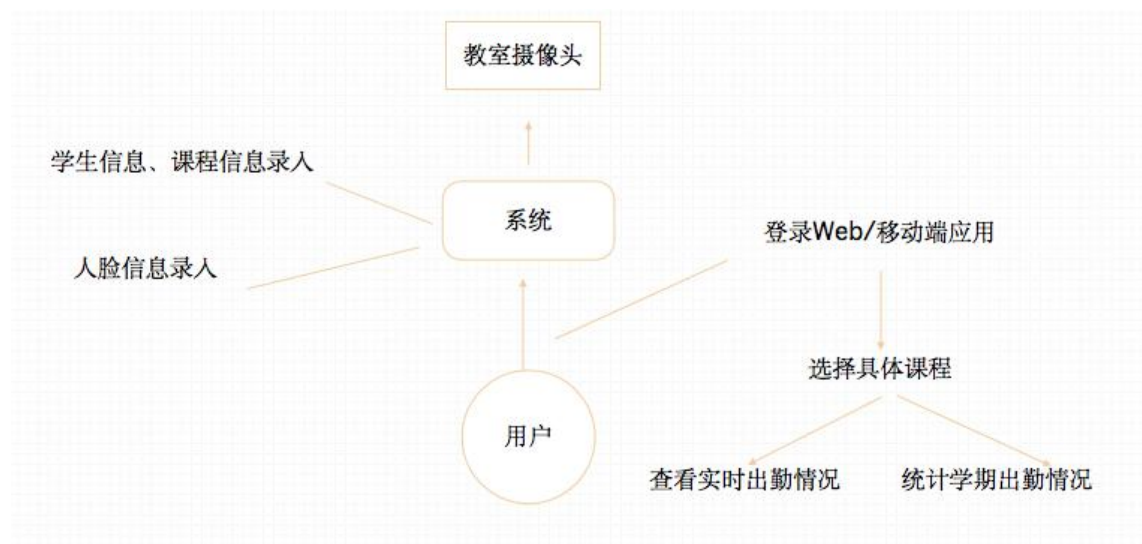


图 1

对于教师（即主要用户）来说，使用操作为：选择具体课程，让学生们进行签到。在期末或者其他需要统计分数的时候查看系统统计好的数据。

对于学校（系统维护者）来说，需要做的事情是录入课程和学生的信息以及学生的人脸数据。

### 3.设计架构

针对 2 中所述的功能设计，设计整个项目的架构如下：



图 2

项目的实现主要分为三块，分别是：(1) 数据库：使用 MySQL，负责管理数据 (2) 后端：主要分为三个模块：摄像模块、人脸识别模块、签到模块 (3) 前端：主要为签到情况展示页面。数据库和 python 的交互使用 pymysql 库，后端和前端通过存取 json 文件交互。

接下来按照重要顺序详细描述这三个主要部分。

### 4.后端模块

#### 4.1 摄像模块

Catch.py, 使用 opencv 库，定时照相并存储照片到本地。

```
1. import cv2
2. import time
3.
4. def opencamera(): # Tested
5.     cap = cv2.VideoCapture(0)
6.     while(cap.isOpened()):
7.         # time.sleep(3)
8.         ret,frame = cap.read()
9.         if ret:
10.            cv2.imshow('frame',frame)
```

```
11. cv2.imwrite('test.jpg',frame)
12. else:
13.     print("camera read error!")
14.     break
15. if cv2.waitKey(1) & 0xFF == ord('q'):
16.     print("manuallly quit")
17.     cap.release()
18.     break
19.
20. import os
21. cnt = -1
22. def findname(): # Tested
23.     global cnt
24.     if cnt!=-1:
25.         cnt+=1
26.         return cnt
27.     else:
28.         for root,dirs,files in os.walk('test_datas'): # Tested
29.             for img in files:
30.                 filename,extention = os.path.splitext(img)
31.                 if extention in ('.jpg','.jpeg','.png'):
32.                     try:
33.                         cnt = max(cnt,int(filename))
34.                     except ValueError:
35.                         continue
36.                 cnt+=1
37.             return cnt
38.
39. def takephoto():
40.     name = findname()
41.     cap = cv2.VideoCapture(0)
42.     if cap.isOpened():
43.         for i in range(4):
44.             print("\r{}".format(3-i),end=")
45.             time.sleep(1)
46.             print("\r",end=")
47.             ret,frame = cap.read()
48.             if ret:
49.                 cv2.imwrite('test_datas/{}.jpg'.format(name),frame)
50.                 print("takephoto success")
51.             else:
52.                 print("takephoto fail")
53.             cap.release()
```

## 4.2 人脸识别模块

Pre\_process.py: 预处理图片，标准化代码

```
1. # make the size of photo <= 2Mb(2000*2000)
2.
3. import os
4. data_path = '/Users/xander/Documents/code/classroom_recogniz
   ation_system/datas'
5. test_path = '/Users/xander/Documents/code/classroom_recogniz
   ation_system/test_datas'
6. from PIL import Image
7.
8. def small(img_url):
9.     fsize = os.path.getsize(img_url)
10.    if (fsize <= 1.8*1024*1024):
11.        return
12.    with Image.open(img_url) as im:
13.        width,height = im.size
14.        m = max(width,height)
15.        width = int(width/m*2000)
16.        height = int(height/m*2000)
17.        im = im.resize((width,height))
18.        im.save(img_url)
19.
20. def small_paths(paths):
21.    print('datas standarlization finished of {}'.format(paths))
22.    for root,dirs,files in os.walk(paths):
23.        for img in files:
24.            filename,extention = os.path.splitext(img)
25.            if extention in ('.jpg','.jpeg','.png'):
26.                img_url = os.path.join(root,img)
27.                fsize = os.path.getsize(img_url)
28.                if (fsize > 1.8*1024*1024):
29.                    small(img_url)
30.
31. def small_datas():
32.    small_paths(data_path)
33.    small_paths(test_path)
34.    print('datas standarlization finished')
```

Face\_API.py: 使用 face++提供的 API 完成人脸检测、人脸识别、人脸搜索功能。

```
1. #face_rec.py
2. import threading
3.
4. API_secret = 'QDpN35MX7OGGLK0TRuEPjwxYMO35KxwT'
5. API_key = '1RuUWbO531XgelX7QH1Ktr0frJA2XrC_'
6. outer_id = 'datas'
7.
8. API0_key = 'wZulEh6shrBv3kJyENxBx9ikU4M_qXra'
9. API0_secret = 'duek1s3zJ4p3YwlsHUY8guAi6yJ3_iv0'
10.
11. d = {'api_key':API_key, 'api_secret':API_secret}
12. d0 = {'api_key':API0_key, 'api_secret':API0_secret}
13.
14. addface_url = 'https://api-cn.faceplusplus.com/facepp/v3/faceset/addface'
15. detect_url = 'https://api-cn.faceplusplus.com/facepp/v3/detect'
16. getdetail_url = 'https://api-cn.faceplusplus.com/facepp/v3/faceset/getdetail'
17. facesetcreat_url = 'https://api-cn.faceplusplus.com/facepp/v3/faceset/create'
18. setuserid_url = 'https://api-cn.faceplusplus.com/facepp/v3/faceset/setuserid'
19. search_url = 'https://api-cn.faceplusplus.com/facepp/v3/search'
20. removefaceset_url = 'https://api-cn.faceplusplus.com/facepp/v3/faceset/removeface'
21.
22. import requests
23. import json
24.
25. def detect(img_url):# Tested
26.     # return a list of face_tokens
27.     try:
28.         fr = open(img_url,'rb')
29.         files = {'image_file':fr}
30.         data = {}
31.         data.update(d)
32.         r = requests.post(detect_url, data = data, files = files)
33.         r.raise_for_status()
34.     except Exception as e:
35.         print("detect error: {} !".format(e))
36.         return False
37.     else:
```

```
38. print("detect success!")
39. finally:
40.     fr.close()
41.
42. ls = []
43. dic = json.loads(r.text)
44. for ddic in dic["faces"]:
45.     ls.append(ddic["face_token"])
46.
47. return ls
48.
49. """
50. dic = json.loads(r.text)
51. with open('detect.json','w') as fp:
52.     json.dump(dic, fp, indent = 2)
53. print('success!')
54. """
55.
56. def setuserid(face_token, ID):# Tested
57.     try:
58.         data = {'face_token':face_token, 'user_id':ID}
59.         data.update(d)
60.         r = requests.post(setuserid_url, data = data)
61.         r.raise_for_status()
62.         # print(r.text)
63.     except Exception as e:
64.         print("set ID error: {} !".format(e))
65.     else:
66.         print("set ID success!")
67.
68.
69. def single_addface(img_url, ID):# Tested
70.     # input a single-face photo + an ID to the faceset
71.     ls = detect(img_url)
72.     if ls == False:
73.         print("{} detect failed!".format(img_url))
74.         return
75.     setuserid(ls[0],ID)
76.     try:
77.         data = {'outer_id':outer_id, 'face_tokens':ls[0]}
78.         data.update(d)
79.         r = requests.post(addface_url, data = data)
80.         r.raise_for_status()
81.     except Exception as e:
```

```
82.     print("addface error: {}".format(e))
83. else:
84.     print("addface success!")
85.
86. def single_search(face_token):# Tested
87.     # single search in faceset
88.     try:
89.         data = {'outer_id':outer_id, 'face_token':face_token}
90.         data.update(d)
91.         r = requests.post(search_url, data = data)
92.         r.raise_for_status()
93.     except Exception as e:
94.         print("search error: {}".format(e))
95.         return False
96.     else:
97.         print("search success!")
98.
99.     dic = json.loads(r.text)
100.     pre_conf = dic["thresholds"]["1e-5"]
101.     real_conf = dic["results"][0]["confidence"]
102.     if real_conf<pre_conf:
103.         return False
104.     else:
105.         ID = dic["results"][0]["user_id"]
106.         print('{} has been found, the confidence is {}'.format(ID,real
_conf))
107.         return ID
108.
109. def multi_search(img_url):#
110.     # multi persons photo to search, return a ID list
111.     ls = detect(img_url)
112.     id_list = []
113.     for i in ls:
114.         result = single_search(i)
115.         if result != False:
116.             id_list.append(result)
117.     return id_list
118.
119. def getdetail(f = 'detail.json'):# Tested
120.     try:
121.         data = {'outer_id':outer_id}
122.         data.update(d)
123.         r = requests.post(getdetail_url, data = data)
124.         r.raise_for_status()
```



```

125.     except Exception as e:
126.         print("getdetail error: {} !".format(e))
127.         return
128.     else:
129.         print("getdetail success!")
130.
131.     dic = json.loads(r.text)
132.     with open(f,'w') as fp:
133.         json.dump(dic, fp, indent = 2)
134.         print('success!')
135.
136.     def clear_faceset(): # Tested
137.         try:
138.             data = {'outer_id':outer_id, 'face_tokens':'RemoveAllFaceTo
kens'}
139.             data.update(d)
140.             r = requests.post(removefaceset_url, data = data)
141.             r.raise_for_status()
142.         except Exception as e:
143.             print("clear error: {} !".format(e))
144.         else:
145.             print("clear success!")

```

### 4.3 签到模块

Keyin.py: 处理识别出来的 ID，和数据库部分交互。

```

1. # input the photo in datas to faceset, set ID as its dir name
2. import os
3. from os import path
4. import face_API as face
5. import pre_process as pp
6.
7. data_path = '/Users/xander/Documents/code/classroom_recogniz
ation_system/datas'
8.
9. def updateall():
10.     pp.small_datas()
11.     face.clear_faceset()
12.     os.chdir(data_path)
13.     pa = os.getcwd()
14.     for ID in os.listdir(pa):
15.         try: # all ID is int number
16.             int(ID)

```

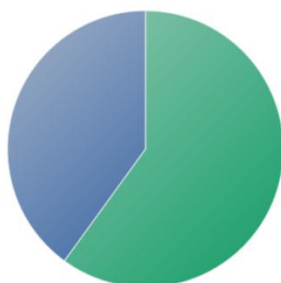
```

17. except ValueError:
18.     continue
19. else:
20.     imgpath = path.join(pa,ID)
21.     for img in os.listdir(imgpath):
22.         (filename,extension) = path.splitext(img)
23.         if extension in ('.jpg','.jpeg','.png'):
24.             face.single_addface(img_url = path.join(imgpath,img) , ID =
ID)
25.
26. def updateID(ID):
27.     try:
28.         int(ID)
29.     except (ValueError,FileNotFoundError):
30.         print("No such an ID!")
31.         return
32.     else:
33.         imgpath = path.join(data_path,ID)
34.         pp.small_paths(imgpath)
35.         for img in os.listdir(imgpath):
36.             (filename,extension) = path.splitext(img)
37.             if extension in ('.jpg','.jpeg','.png'):
38.                 face.single_addface(img_url = path.join(imgpath,img) , ID = I
D)

```

## 5.前端设计

使用 bootstrap4 的模态框和 jquery 实现饼图。



已到人数	3
未到人数	2
已到名单	未到名单

图 3

模态框实现的签到显示:



图 4

## 6.数据库说明

Attendance.py: 与数据库交互的部分。

```

1. import face_API as face
2. import pymysql as pl
3. import pre_process as pp
4. import json
5.
6. infodict = {}
7.
8. def entercourse(coursename):# Tested
9.     db = pl.connect(host="rm-m5ec899sxqwx2rc9tgo.mysql.rds.aliyunc
s.com",
10.                     user="root", password="Aa123456", db="classroom",
charset='utf8')
11.     cur = db.cursor()
12.
13.     findid = "select courseid from courses where coursename='{ }'".
format(coursename)
14.     cur.execute(findid)
15.     try:
16.         courseid = cur.fetchone()[0]
17.     except TypeError:
18.         print("No such a course!")
19.     return

```

```
20.
21. select = "select id from {}".format(courseid)
22. cur.execute(select)
23. idlist = []
24. for i in cur.fetchall():
25.     idlist.append(i[0])
26.
27. global infodict
28. infodict = {}
29. for ID in idlist:
30.     getinfo = "select * from students where ID=\"{}\"".format(ID)
31.     cur.execute(getinfo)
32.     lls = cur.fetchall()[0]
33.     infodict[lls[0]] = {'name':lls[1], 'class':lls[2]}
34. # print(infodict)
35. cur.close()
36. db.close()
37.
38. def rollcall(img_url):
39.     # return json
40.     global infodict
41.
42.     pp.small(img_url)
43.     ls = face.multi_search(img_url)
44.
45.     donels, undonels = [],[]
46.     for i in infodict:
47.         if i in ls:
48.             donels.append(i)
49.         else:
50.             undonels.append(i)
51.
52.     dic = {}
53.     dic["donenumber"] = len(donels)
54.     dic["undonenumber"] = len(undonels)
55.     dic["done"] = form(donels)
56.     dic["undone"] = form(undonels)
57.     with open("display/rollcall.json", "w") as fp:
58.         json.dump(dic, fp, ensure_ascii=False, indent = 2)
59.
60. def form(ls):# Tested
61.     global infodict
62.     dic = {}
63.     for ID in ls:
```

```

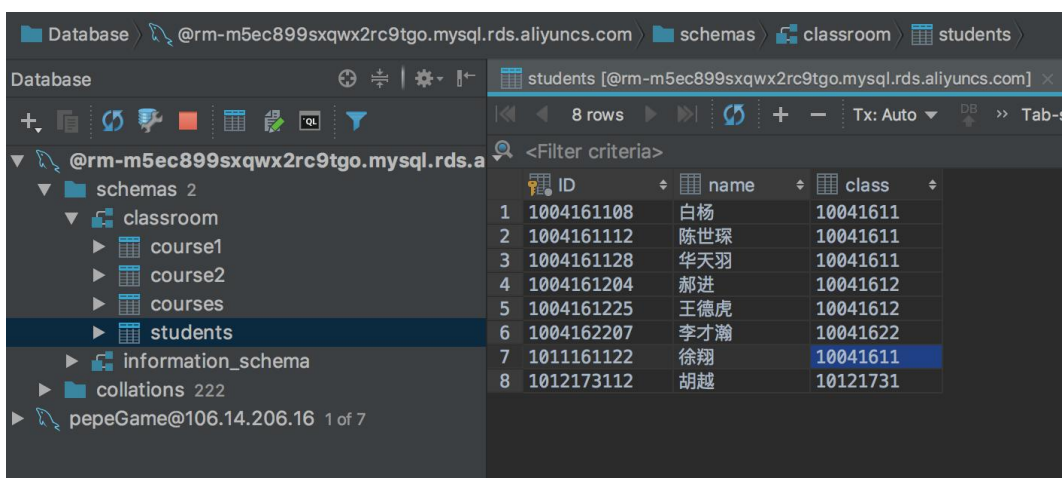
64. cl = infodict[ID]['class']
65. if cl not in dic:
66.     dic[cl] = {}
67. dic[cl][ID] = infodict[ID]['name']
68. return dic

```

数据库组成：

course 表，包含这门课的名单内的 ID。

students 表，包含 ID、name、class。



ID	name	class
1004161108	白杨	10041611
1004161112	陈世琛	10041611
1004161128	华天羽	10041611
1004161204	郝进	10041612
1004161225	王德虎	10041612
1004162207	李才瀚	10041622
1011161122	徐翔	10041611
1012173112	胡越	10121731

图 5

## 7.总结

本次项目主要的收获：

1. 如何将前端 Web、后端的处理模块以及底部的数据库连接起来（中间的连接方式）
2. 前端开发工具的使用
3. 后端处理模块中 openCV 等库的使用
4. 通过 Python 对 SQL 的操作

存在的问题主要如下：

1. 前端设计得很简陋
2. 后端没有使用较为先进的人脸识别方法
3. 整个项目的功能不够完整

致谢

感谢牛云云老师的指导。

感谢的信息工程学院大学生创新实验室的支持。

### **【参考文献】**

[1]何长婷. 课堂签到系统中的人脸识别方法研究与实现[D].中国科学技术大学,2018.

[2]李玮. 智慧课堂管理系统中人脸识别考勤技术的研究与实现[D].华中师范大学,2017.