

核酸检测结果识别系统

HELLO，大家好。现在是2022年8月9日20:14:44。

在老师和伙伴的帮助下，我完成了**核酸检测结果识别系统1.0**。相信这套简单方便的系统，能够为大家提供便利。

产品介绍

产品功能

由于高校频繁的核酸检测，导致每次结果统计非常麻烦。这套系统可以极大减轻负责人（例如辅导员）的工作压力。同学们把检测后的图片上传到网站入口，在下方输入框内填上姓名即可。

系统介绍：

系统作者：杨浩然，彭兴锴，曾品典，方友清，王晓晖

指导老师：吕皖丽老师，郭星老师，郭佳明（导生），张雨（学长）

特别鸣谢：丛言

提交要求：

1.上传常规的核酸检测结果截图，并填写下方学号。

2.字体要求正楷，不要使用艺术字体！

*报告图片：

点击右方按钮选择图片

 选择图片

*学号：

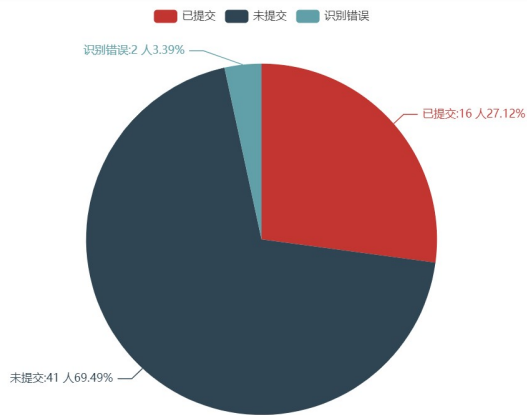
必填！

提交

重置

后台会自动识别图片的检测结果、时间并写入数据库汇总，实时反馈提交情况并制作图标，反馈信息。

提交结果



识别失败名单

学号	姓名	检测时间	检测结果
王念祖	E42014020	时间匹配失败	阴性
耿紫涵	E42014030	时间匹配失败	阴性

未提交&识别失败 名单

学号	姓名	检测时间	检测结果
季云龙	B02014068		
赵嘉怡	B02014118		
杨宇海	B02014152		

技术栈

开发工具: VSCode, PyCharm, sqlyog

前端技术: HTML+CSS+JS, jquery, echarts, bootstrap

后端技术: Python, OCR, opencv, SQL, numpy, pyecharts

使用说明

需要提供给网站管理员一份本班的学号+姓名表格作为数据库, 生成的数据表格 (result.xlsx)。

可视化数据界面实时更新提交和识别情况, 属于私有界面, 不会对外公开, 只对班级的管理员开放, 因此可以很大程度上保证使用者的信息安全。

技术算法讲解

组成部分

easyocr图像文字提取+opencv颜色识别+正则表达式结果匹配+Python写入Excel和数据库+pyecharts可视化界面 + schedule控制挂载运行;

HTML+CSS+JS+jquery+bootstrap (前端) + PHP (后端) + Nginx反向代理 + pm2 (项目部署)

easyocr

easyocr是python 2021年推出的第三方包, 可以非常精准的识别、提取图片文字 (亲测有效)。这项功能主要被用来提取学生的核酸检测时间。下面是easyocr的具体使用:

```
f = open('result.txt', 'w')
for filename in os.listdir(directory_name):
    reader = easyocr.Reader(['ch_sim', 'en'], gpu=False) # GPU or CPU
    result = reader.readtext(directory_name + r'/' + filename, detail=0)
    result = str.join(result)
    f.write(result)
    f.write("\n")
```

这里要注意两点，`['ch_sim', 'en']` 是识别语言的列表（中、英文），如果需要其他语言可以去搜它对应的代码。`gpu=false` 表示我的电脑配置没有gpu，设置成用cpu跑深度学习的代码。

另外，`reader.readtext()` 是不支持中文路径的，读取的路径一定要注意！

读取完图片文字后，我们把他写入一个txt中，下面要从杂乱的文字中提取出学生的核酸检测时间：

```
try:
    timeresult = re.search('采集时间[\d,\-,:.]*', result) # 获取采集时间
    timeres.append(timeresult.group()[4:]) # 去掉前四个字“采集时间”

except Exception as err:
    timeres.append("时间匹配失败")
```

这里我们一定要加入异常处理，防止因为匹配失败而直接报错中断运行。

opencv

本身是打算用easyocr直接提取“阴性”这两个字，但是发现这两个字实在不好提取。所以决定用opencv的颜色识别解决结果的判断问题。首先我们列出一个颜色清单，命名为colorList.py。为了识别更加准确，我们只留下红绿两种颜色。

```
import numpy as np
import collections

def getColorList():
    dict = collections.defaultdict(list)

    # 红色
    lower_red = np.array([156, 43, 46])
    upper_red = np.array([180, 255, 255])
    color_list = []
    color_list.append(lower_red)
    color_list.append(upper_red)
    dict[r'colorList/red'] = color_list

    # 绿色
    lower_green = np.array([35, 43, 46])
    upper_green = np.array([77, 255, 255])
    color_list = []
    color_list.append(lower_green)
    color_list.append(upper_green)
```

```
dict[r'coloList/green'] = color_list

return dict

if __name__ == '__main__':
    color_dict = getColorList()
```

之后用opencv函数进行颜色判断；但 `cv2.imwrite` 函数只支持jpg图片，所以我们在网站收集图片时，可以将目标图片进行.jpg文件转换。之后import刚刚的colorList，使用opencv识别图片的颜色。

```
import colorList

def get_color(frame):
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    maxsum = -100
    color = None
    color_dict = colorList.getColorList()
    for d in color_dict:
        mask = cv2.inRange(hsv, color_dict[d][0], color_dict[d][1])
        cv2.imwrite(d + '.jpg', mask) # 这里一定要保证，读取的图片是jpg
        binary = cv2.threshold(mask, 127, 255, cv2.THRESH_BINARY)[1]
        binary = cv2.dilate(binary, None, iterations=2)
        cnts, hiera = cv2.findContours(binary.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
        sum = 0
        for c in cnts:
            sum += cv2.contourArea(c)
        if sum > maxsum:
            maxsum = sum
            color = d

    return color
```

excel表和数据库

为了方便管理者导出，我们用python将结果写入了excel表格；为了做出可视化界面，我们也需要将结果写入database。数据库的建立比较简单，分为四个字段：

字段	数据类型
sno (学号)	varchar
sname (姓名)	varchar
time_result (时间检测结果)	varchar
test_result (检测结果)	varchar

python写入excel表格的操作比较简单，另外，如果直接在浏览器输入.xlsx的文件地址，可以直接下载获取，也是十分方便。

在写入之前，我们需要先清理干净excel表格，我选择直接删除Sheet1，然后新建一张。做数据库也是一样，每次清理一次数据库，防止旧数据干扰新一次的检测。刷新后，写入本次的结果。

```
db = pymysql.connect(host='localhost', user='root', passwd='root', port=3306,
db='hesuan_result_collection')

cursor = db.cursor()

bg = op.load_workbook(r"result.xlsx") # 应先将excel文件放入到工作目录下
bg.remove(bg["Sheet1"])

bg.create_sheet("Sheet1", index=0)
sheet = bg["Sheet1"]
sheet.cell(1, 1, "学号")
sheet.cell(1, 2, "姓名")
sheet.cell(1, 3, "检测时间")
sheet.cell(1, 4, "检测结果") # 刷新excel表格的数据

sql_delete = "Update xinan set time_result = '' , test_result = '' "
cursor.execute(sql_delete)
db.commit() # 刷新数据库的数据

if len(timeres) == len(testres):
    for i in range(1, len(timeres) + 1):
        sql_fetch_name = "select sname from xinan where sno = '%s' " % student_sno[i - 1] # 获取
        学号对应的姓名
        cursor.execute(sql_fetch_name)

        sheet.cell(i + 1, 1, student_sno[i - 1])
        try:
            sheet.cell(i + 1, 1, student_sno[i - 1])
        except Exception as err:
            sheet.cell(i + 1, 1, "找不到姓名")
        sheet.cell(i + 1, 3, timeres[i - 1])
        sheet.cell(i + 1, 4, testres[i - 1]) # 分别写入学号、姓名、检测时间、检测结果
        bg.save(r"result.xlsx") # 对文件进行保存

        sql_update = "UPDATE `xinan` " \
            "SET `time_result` = '%s' , `test_result` = '%s' " \
            "WHERE sno = '%s'; " % (timeres[i - 1], testres[i - 1], student_sno[i - 1])
        try:
            sql_update_result = cursor.execute(sql_update)
            db.commit() # 提交数据并保存
        except Exception as err:
            print("数据库写入失败:", err) # 这里同样也要加入异常处理

    else:
        print("长度不匹配")
print("finished")
```

Pyecharts可视化界面

做好了算法部分，我们也要能够展示实时提交的情况。这样可以更方便班委管理，及时催促指定同学提交核酸检测报告。首先我们连接数据库，写好相关的sql语句：

```
import pymysql
import main

db = pymysql.connect(host='localhost', user='root', passwd='root', port=3306,
db='hesuan_result_collection')
cursor = db.cursor() # 数据库连接

sql_find_failed = "SELECT DISTINCT sname,sno,time_result,test_result FROM xinan WHERE
time_result = '时间匹配失败' or test_result = '结果存疑'" # 查找检测失败的同学名单
sql_readyto_submit = "SELECT DISTINCT sname,sno,time_result,test_result FROM xinan " \
"WHERE time_result = '时间匹配失败' or test_result = '结果存疑'" \
"or time_result = '' or test_result = ''" # 查找需要重新提交的同学名单，包括
结果为空和结果错误的总和

main.student_failed = cursor.execute(sql_find_failed)
main.student_left = main.student_num - main.student_checked - main.student_failed
```

之后使用sql查找的结果，制作饼图和表格

```
def pie_base():
    label = ['识别成功', '未提交', '识别错误']
    values = [main.student_checked, main.student_left, main.student_failed] # 计算的三个变量制作饼
    图，识别成功+未提交+识别错误 == 所有学生
    c = (
        Pie()
        .add("", [list(z) for z in zip(label, values)])
        .set_global_opts(title_opts=opts.TitleOpts(title="20信安提交结果"))
        .set_series_opts(label_opts=opts.LabelOpts(formatter="{b}:{c}人 {d}%")) # 值得一提的是，{d}%为百分比
    )
    return c

def table_base() -> Table:
    cursor.execute(sql_find_failed)
    temp = cursor.fetchall() # 获取全部的结果
    failed_match = []
    for i in temp:
        x = list(i)
        failed_match.append(x) # 这里fetchall()函数返回的是元组，需要将元组转成列表进行表格制作
    table = Table()
    headers = ["学号", "姓名", "检测时间", "检测结果"]
    rows = failed_match
    table.add(headers, rows).set_global_opts(
        title_opts=opts.ComponentTitleOpts(title="识别失败名单")
    )
    return table
```

```
def page_simple_layout():
    page = Page(layout=Page.SimplePageLayout) # 简单布局
    page.add(
        pie_base(), # 展示饼图
        table_base(), # 展示识别失败名单
        table_base2(), # 展示
    )
    page.render("./result.html")

if __name__ == "__main__":
    page_simple_layout()
```

运行控制

使用python的schedule进行运行控制，我们可以有很多控制方法，将这个文件设置成run.py，之后将他用pm2或者nohup挂载运行即可。

```
import schedule
import os

def run():
    os.system("python view.py")

schedule.every().day.at("10:30").do(run) # 每天的10:30执行一次任务
# schedule.every().monday.do(run) # 每周一的这个时候执行一次任务
# schedule.every().hour.do(run) # 每隔一小时执行一次任务

while True:
    schedule.run_pending() # run_pending: 运行所有可以运行的任务
```

网站部分

前端

前端部分使用传统三件套+jquery+bootstrap，可以展示项目开发人员列表和提交要求，上传图片时实现图片预览，并且检测用户输入的是否为空，是否为学号。

首先我们引入jquery和bootstrap的核心css，js文件

```
<!-- jquery -->
<script src="https://cdn.bootcss.com/jquery/3.2.1/jquery.min.js"></script>

<!-- bootstrap核心 js 与 css 文件 -->
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@3.3.7/dist/css/bootstrap.min.css"
integrity="sha384-BVYiisSIFeK1dGmJRAkycuHAHRg320mUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
crossorigin="anonymous">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@3.3.7/dist/js/bootstrap.min.js"
```

```

integrity="sha384-Tc5IQib027qvyjSMfHj0MaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNlcpPD7Txa"
crossorigin="anonymous"></script>

<!-- bootstrap 图片上传相关文件 -->
    <link href="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-
fileinput/4.4.9/css/fileinput.min.css" media="all" rel="stylesheet" type="text/css" />
    <script src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-
fileinput/4.4.9/js/plugins/piexif.min.js" type="text/javascript"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-
fileinput/4.4.9/js/plugins/sortable.min.js" type="text/javascript"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-
fileinput/4.4.9/js/plugins/purify.min.js" type="text/javascript"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.11.0/umd/popper.min.js">
</script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-
fileinput/4.4.9/js/fileinput.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-
fileinput/4.4.9/themes/fa/theme.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-
fileinput/4.4.9/js/locales/(lang).js"></script>

```

首先，设置一个简单的空白抬头，使页面可以在中间显示：

```

style>
    #main-form
    {
        margin-top: 30px;
    }
</style>

```

之后在body下面套一层div和form表单：

```

<form id="form" action="hesuan.php" class="form-horizontal" method="post"
enctype="multipart/form-data" onSubmit="return finaljudgeSno();">
...
...
</form>

```

我们用textarea展示文本内容，例如开发者、鸣谢者名单和提交要求之类的信息。`class="form-control"` 是bootstrap专有样式，使用 `rows = 7` 可以对宽度进行调整。这里还要说明一下，textarea真的是textarea，回车空行都是会显示的，比较方面。


```

<div class="form-group">
    <label class="col-md-2 control-label small"><span class="text-danger">
</span>系统介绍: </label>
    <div class="col-md-10 has-success">
        <textarea name="introduction" readonly type="text" class="form-control"
placeholder="" rows = "7">系统作者: 杨浩然, 彭兴锴, 曾品典, 方友清, 王晓晖

指导老师: 吕皖丽老师, 郭星老师, 郭佳明 (导生), 张雨 (学长)

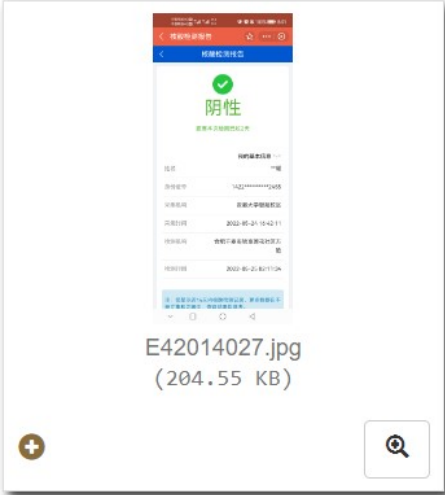
特别鸣谢: 丛言

        </textarea>
    </div>
</div>

```

选择图片后的界面是这样的:

*报告图
片:



E42014027.jpg
(204.55 KB)

E42014027.jpg

选择图片

*学号:

E42014039

格式正确

提交

重置

使用这段网上写好的js (现在真的找不到原作者了, 大海捞针) 来实现图片的预览, 我本人也不是什么JavaScript高手, 有些代码还是喜欢搬运一下哈哈

这里很多设置也都加入了注释, 允许同时上传文件个数我们设置成1。

```

<script>
    // 图上传与预览相关控制
    $('#pic').fileinput(
    {
        showUpload : false, //是否显示上传按钮, 跟随文本框的那个
        showRemove : false, //显示移除按钮, 跟随文本框的那个
    }

```

```

        showCaption : true, //是否显示标题,就是那个文本框
        showPreview : true, //是否显示预览,不写默认为true
        dropZoneEnabled : false, //是否显示拖拽区域, 默认不写为true, 但是会占用很大区域
        //minImageWidth: 50, //图片的最小宽度
        //minImageHeight: 50, //图片的最小高度
        //maxImageWidth: 1000, //图片的最大宽度
        //maxImageHeight: 1000, //图片的最大高度
        //maxFileSize: 0, //文件最大大小, 单位为kb, 如果为0表示不限制文件大小
        //minFileCount: 0, //文件最小大学, 单位为kb, 如果为0表示不限制文件大小

        maxFileCount : 1, //表示允许同时上传的最大文件个数
        enctype : 'multipart/form-data',
        validateInitialCount : true,
        previewFileIcon : "<i class='glyphicon glyphicon-king'></i>",
        allowedFileTypes : [ 'image' ], //配置允许文件上传的类型
        allowedPreviewTypes : [ 'image' ], //配置所有的被预览文件类型
        allowedPreviewMimeTypes : [ 'jpg', 'png', 'gif' ], //控制被预览的所有mime类型
        language : 'zh'
    }
)

// 控制生成的代码的样式
$('input.file-caption-name').attr('placeholder', '点击右方按钮选择图片');
$('span.hidden-xs').text("选择图片");
</script>

```

之后再用这段代码, 判断一下用户输入的是不是学号。

```

<script>
    var tag = 0;
    judgeSno = function() {
        var sno = document.getElementById("sno");
        var str = "";
        str = sno.value
        var Regex = /\w/; //英文 + 字母
        var str1 = str[0];
        var str1Code = str1.charCodeAt();
        var hightag = 0;
        if(str1Code >= 65 && str1Code <= 90){
            hightag = 1; // 判断首字母是否为大写
        }
        if (str.length == 9 && (sno.value.match(Regex) && hightag == 1 )) {
            document.getElementById("checkSno").innerText = "格式正确";
            document.getElementById("checkSno").style.color = "green";
            tag = 1;
        } else {
            document.getElementById("checkSno").innerText = "格式不对";
            document.getElementById("checkSno").style.color = "red";
            tag = 0;
        }
        return tag;
    }
</script>

```

```

<script>
    finaljudgeSno = function(){
        var finaltag = tag;
        if(finaltag == 0){
            alert("请输入正确的学号! ")
            return false;
            location='https://www.xinanzhijia.xyz/hesuan.html';
        }
        else{
            return true;
        }
    }
}
</script>

```

后端

后端使用简单的php，实现将上传图片移动到服务器某目录的功能。多余的一项功能是将上传的图片命名成学号，下面是PHP的后端代码

```

<?php
header("Content-type: text/html; charset=utf-8");
$upload_file = $_FILES["file"];
$upload_name = $_POST["name"];
$store_dir = 'hesuan/Xinan/'; // 改!!!! 1
if($upload_file["error"]>0){
    // echo "错误: ".$file["error"];
    if($upload_file["error"]==4){
        echo "<script>alert('请选择图片提交');
        location='hesuan.html'
        </script>";
    }
} // 链接改!!!!
if($upload_name==null)
{
    echo "<script>alert('请输入学号');
    location='hesuan.html'
    </script>";
}
else{
    $arr = ".jpg";
    $new_name = "{$upload_name}{$arr}";
    $upload_file["name"] = $new_name;
    $name = iconv('utf-8','gbk',"hesuan/Xinan/".$upload_file["name"]); // 改!!!!
    if(move_uploaded_file($upload_file['tmp_name'],$name)){
        move_uploaded_file($upload_file['tmp_name'],$store_dir.$new_name);
        echo "<script>alert('提交成功');
        location='hesuan.html';
        </script>";
    }
    else{
        echo "<script>alert('提交失败');

        location='hesuan.html';
    }
}

```

```
        </script>";  
    }  
}  
?>
```

总结

到这里，我们终于完成了逻辑闭环。

Step1：同学们将核酸检测的截图提交到网页（并输入合规的学号）

Step2：服务器将文件统一收到某个文件夹下，进行智能识别匹配，将结果写入对应班级的数据库。

Step3：从数据库调出目前的提交情况，反馈到可视化界面和Excel表格（管理员可查看\下载）

Step4：Python实现定时工作并定期删除全部的报告，数据实现阶段更新的大循环。