东软教育科技集团

项目实训总结报告

项 目 名 称:“识食物者”--美食识别系统

学 校 名 称: 内蒙古大学

项 目 讲 师: 王志勇

姓 名: 崔爽

学 号: 0191120227

实 训 日 期: 2022年7月11日-8月6日

**1 引言** （一级标题，小三黑体，居中）

1.1项目概述（二级标题，四号黑体，左对齐）  
说明项目目的、背景、意义及参考资料（包括企业提供的素材、DEMO）等；

* + 1. 项目背景  
        现今世界各地美食多种多样，我们时常会遇到不了解的美食，同时顺应夏季学期企业实训要求，一款可以识别六种美食的实时识别系统应运而生。
    2. 项目目的  
        为提供一款识别准确度较高的美食识别系统，帮助用户辨别食物并了解食物热量及价钱。对于本次项目主要是为了能够学习并了解机器学习、深度学习、python爬虫、pythonweb-flask框架相关知识，并锻炼实际动手能力和小组之间的合作能力，提前了解企业的工作方式，并养成撰写文档的良好习惯，学习到对自己有用的知识。
    3. 项目意义  
        根据拍摄照片，识别图片中美食名称，获取美食参考卡路里含量和百科信息，可结合识别结果进一步提供饮食推荐、健康管理方案等相关功能，增强用户体验。系统的学习到了人工智能相关的知识，锻炼了自己的实践能力和团队合作能力。
    4. 参考资料  
       [1] 中国大学慕课：Python网络爬虫与信息提取--北京理工大学

[2] 中国大学慕课：Python数据分析与展示--北京理工大学  
[3] Flask Web开发 : 基于Python的Web应用开发实战 / (美) 格林布戈 (Grinberg,M.) 著 ; 安道译. -- 北京人民邮电出版社, 2015.1 （图灵程序设计丛书）

[4] Python深度学习/(美)弗朗索瓦·肖莱 著 张亮(hysic) 译 人民邮电出版社出版时间：2018-08（图灵程序设计丛书）

1.2 需求分析 *(课程目标支撑4 ; 毕业要求支撑10.2)*为什么做这个题目，有什么实际价值

美食识别对于我们日常生活有着很重要的作用，同时它也可以作为一个“插件”植入到其他的应用场景中。我们可以根据拍摄照片，识别图片中美食名称，获取美食参考卡路里含量和百科信息，可结合识别结果进一步提供饮食推荐、健康管理方案等相关功能，同时我们也可以在餐厅中应用这种方式进行智能结算等。

同时根据这个项目我们会接触到许多全新的领域，锻炼我们的编程实践、项目实践能力。我们在做完此项目之后可以进行扩展。将之与人们的饮食健康连接起来，产生更大的社会效益。

1.3 运行环境  
软、硬件环境

（正文，小四号宋体，1.25倍行距）

1.3.1硬件环境：

1、CPU：Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz 2.60 GHz

2、RAM：16G

3、OS：win10x64

4、Storage：1T

1.3.2 软件环境：

1、编译器：PyCharm Community Edition 2022.1.3、Jupyter notebook

2、Python解释器：Python 3.7

3、GPU驱动：cuda7.6、cudnn10.1

4、python拓展包：urllib、beautifulsoup、time

**2 项目设计**（小三黑体，居中）*(课程目标支撑3 ; 毕业要求支撑9.2)*

（以个人主要负责内容为主）

2.1 设计思路  
 我负责的部分主要是数据爬取、数据清洗（包括图片去重、重命名、格式化等）、数据整合等操作。项目的开展和模型的训练都需要提供原始的数据集。我们初步拟定六类美食每一类最终爬取1000张图片作为原始数据集，随后就是爬取网站的选取，最初我试想在百度图片中进行数据的爬取，可是当我进行关键字搜索测试的时候发现百度最大的问题就是无关信息太多，这样对于数据清洗的代价太高。

通过我对不同网站的对比最后决定将bing网站作为数据爬取的主要目标，因为bing网站的图片无关信息较少，最重要的是bing的图片网址翻页很有规律可循，是下拉式加载。

在数据爬取阶段定义三个函数并抽象出三个方法，GetImage、findImgUrlFromHtml、getStartHtml。在数据清洗阶段我们要对爬取的数据做一个整合，并对所有的图片进行重命名操作，统一为相同格式，比如汉堡可以统一为：“hamburger.0.jpg”、“hamburger.1.jpg”........“hamburger.999.jpg”。还有格式化操作，将图片统一为相同尺寸大小并进行数据去重。

数据爬取过来之后由于格式不统一肯定是要进行数据清洗，数据清洗操作就要涉及到对图片进行批量的格式化、重命名、去重等操作。

2.2 模块功能介绍

数据爬取阶段，GetImage负责从原图的url中将原图保存到本地；findImgUrlFromHtml负责从缩略图列表页中找到原图的url，并返回这一页的图片数量；getStartHtml负责获取缩略图列表页。

数据清洗阶段，ImgResize负责把所有的图片统一修改成相同像素和格式，将png格式都转换成jpg格式；ImgRename负责将所有的图片名字改成相同的格式，以便模型训练；ImgRemoval负责去除重复图片，保证数据的健壮性；DataCut负责将整合好的数据切分成训练集、验证集、测试集，以便模型训练使用。

2.3 模块结构图

2.3.1：数据爬取模块

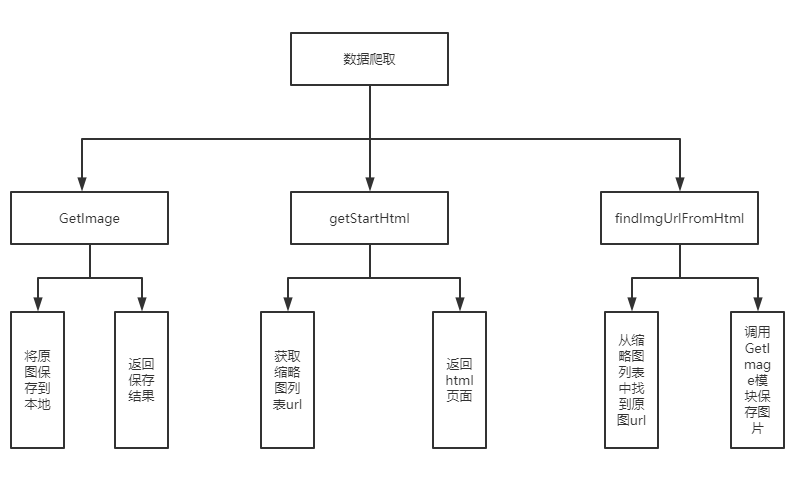


图2.3-1：数据爬取模块结构图

2.3.2：数据清洗模块

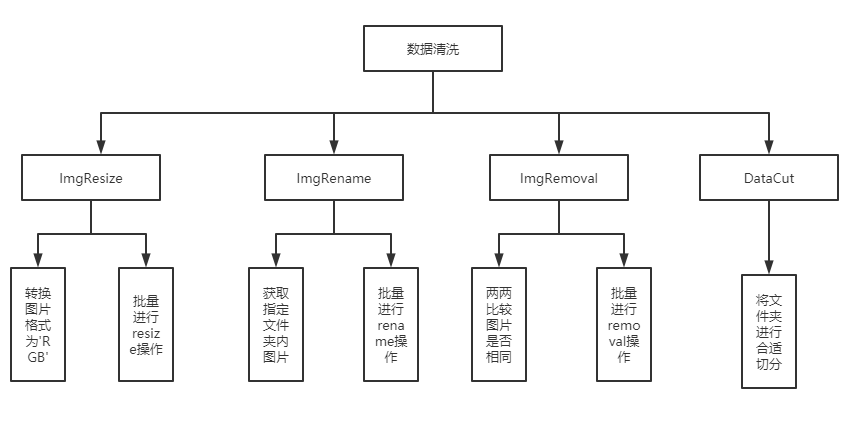


图2.3-2：数据清洗模块结构图

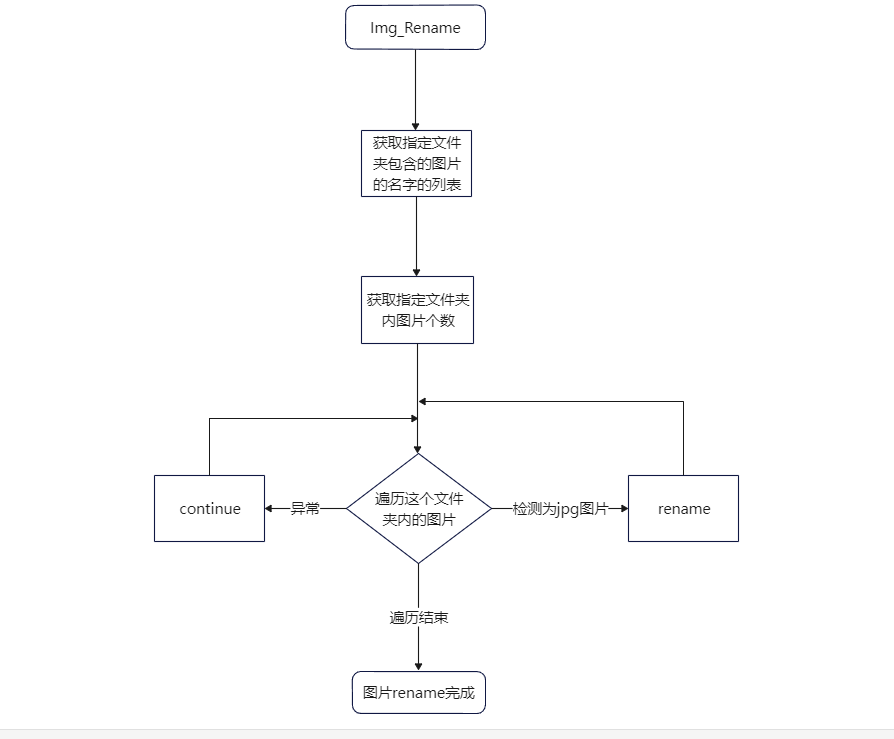
2.4 程序流程图

图2.4-1：图片重命名流程概要

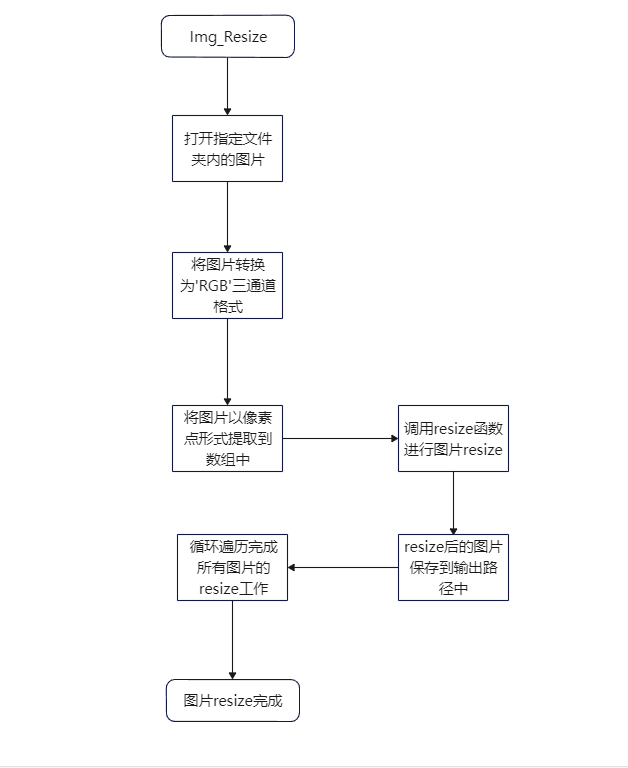


图2.4-2：图片格式化流程概要

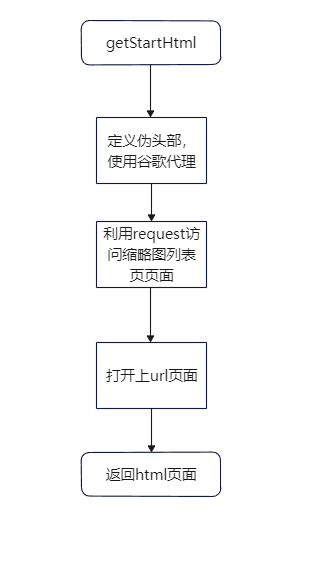


图2.4-3：getStartHtml流程概要

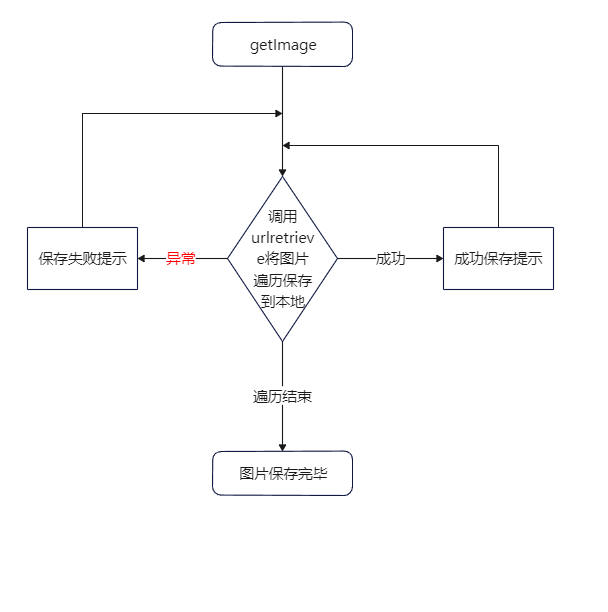


图2.4-4：getImage流程概要

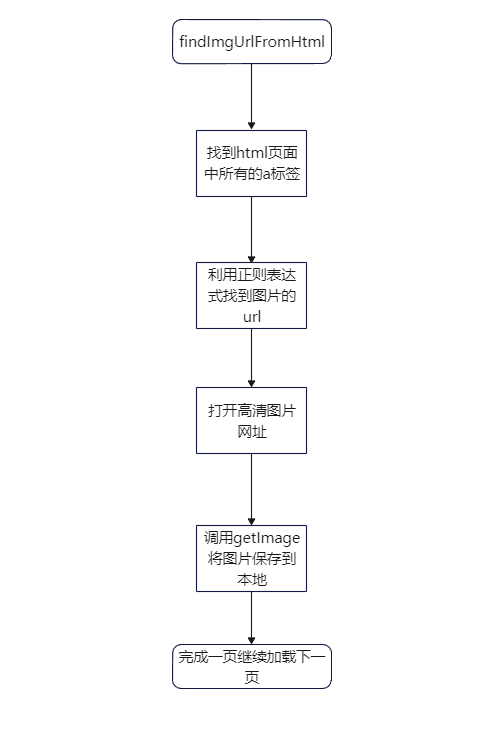


图2.4-5：findUrlFromHtml流程概要

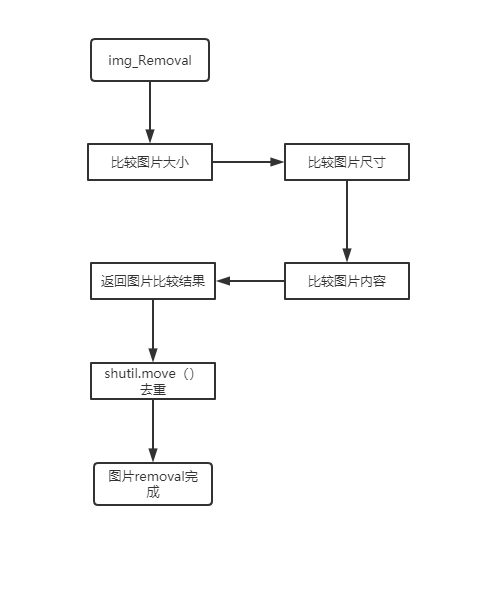


图2.4-6：图片去重流程概要

2.5 功能设计分工（团队分工）

需详细标注每部分的内容，具体任务及任务安排等。

|  |  |  |
| --- | --- | --- |
| **序号** | **任务名称** | **工时** |
| 1 | 抽象爬虫函数 | 0.25天 |
| 2 | 选取目标网站，进行网页规律找寻 | 0.25天 |
| 3 | 编写网络爬虫代码，进行数据爬取；编写数据清洗相关函数代码 | 1.5天 |
| 4 | 数据清洗、数据整合 | 1.0天 |

**3 详细设计**（小三黑体，居中）*(课程目标支撑3 ; 毕业要求支撑9.2)*

详细写明项目的设计（系统设计、数据库设计、页面等）等，并写明对所涉技术、核心功能等描述；

爬虫部分所用到的核心技术就是urllib和beautifulsoup库的使用。urllib的request模块可以非常方便地抓取URL内容，即将GET请求发送到指定页面，然后返回HTTP响应，如果想模拟浏览器来完成特定的功能，需要将请求伪装成浏览器。伪装方法是先监控浏览器发出的请求，然后根据浏览器的请求头进行伪装，User-Agent头是用来识别浏览器的，如果我们使用google浏览器可以轻松的查到User-Agent。而beautifulsoup用起来比较简便，同时又能很好的节省编程时间，最重要的是它可以很好的处理不规范标记并生成剖析树。

爬虫部分首先抽象出三个函数，GetImage负责从原图的url中将原图保存到本地；findImgUrlFromHtml负责从缩略图列表页中找到原图的url，并返回这一页的图片数量；getStartHtml负责获取缩略图列表页。

数据清洗部分主要用到了PIL库，其可用于图片剪切、粘贴、缩放、镜像、水印、颜色块、滤镜、图像格式转换、色场空间转换、验证码、旋转图像、图像增强、直方图处理、插值和滤波等功能。

数据清洗部分也抽象出了几个函数；ImgResize负责将图片格式化为统一样式；ImgRename负责将图片命名为统一样式；ImgRemoval负责进行图片去重操作；DataCut负责将整合好的数据切分成训练集、验证集、测试集。

**4 实现方案**（小三黑体，居中）*(课程目标支撑3 ; 毕业要求支撑9.2)*

详细写明项目功能实现描述、核心代码及实现结果等；

并对调试好的程序，从执行程序弹出界面开始，每一步操作截一个图，并附加说明（图片格式为JPG格式）。

4.1：实现描述

在getImage中我们直接做一个异常处理，如果正常的情况下就调用相关方法将图片从原图中保存到本地，当然这里我们需要做一个反爬机制，防止浏览器识别到我们爬取的频率太快从而将我们的IP进行封锁，所以这里我们调用sleep函数每特定时间进行爬取。

核心代码：urllib.request.urlretrieve(url,dstpath）

在findImgUrlFromHtml中我们首先利用beautifulsoup将其对象化，根据我对网站的分析我发现所有的高清图片url都存在了a标签下的murl里，所以我们这里利用beautifulsoup找到所有的a标签，然后再利用正则表达式找到高清大图的url。找到之后我们即可调用GetImage将图片保存到本地。

核心代码：soup = BeautifulSoup(html, "lxml")

soup.findall(‘a’)

rule=re.compile(r"\"murl\"\:\"http\S[^\"]+\.(?:jpg|JPG|png)")

在getStartHtml中我们直接找到缩略图列表的url，打开之后返回此url的html页面即可。

核心代码：html = urllib.request.urlopen(page)

在ImgResize中我们首先使用PIL将图片打开并提取到数组中，因为在爬虫阶段我们将图片强制类型转换为了.jpg格式，但并未真正改变，所以这里我们还要将其彻底转换成’RGB’格式，之后我们就可以调用PIL库中的方法进行resize操作。

核心代码：pic.convert('RGB')

img.resize((x,y))

在ImgRename中我们只需要调用os库中的rename方法，循环遍历改名即可。

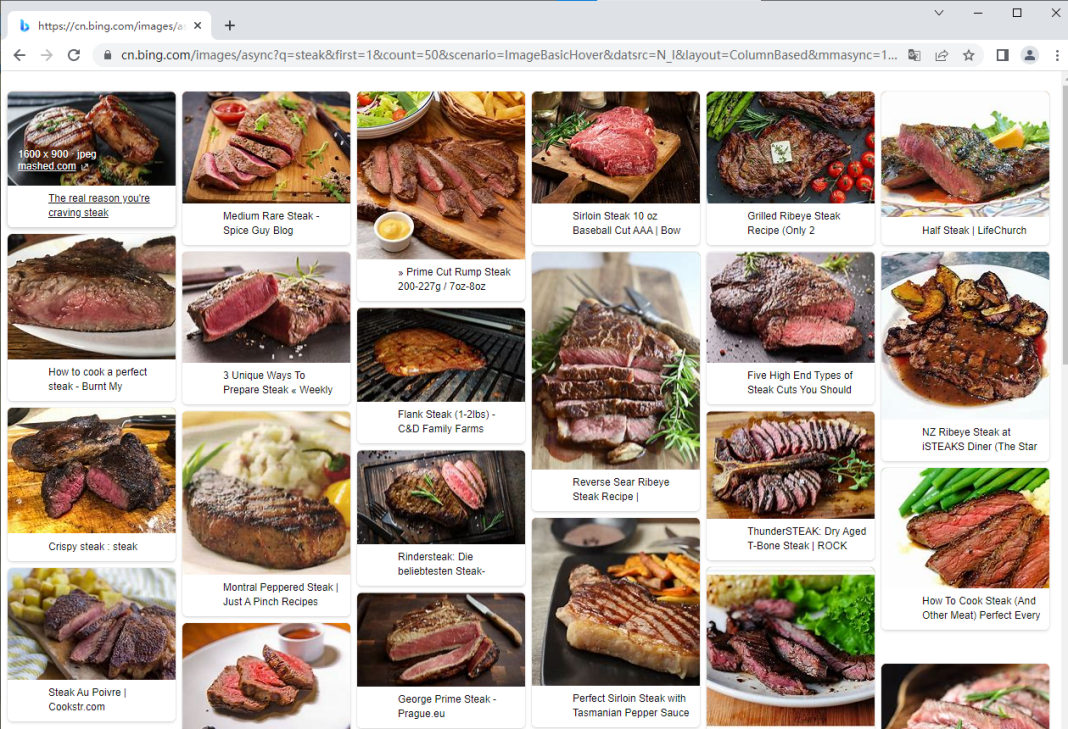
核心代码：os.rename(src, dst)

在ImgRemoval中我们要分别两两比较图片的大小、尺寸、内容，然后返回结果，这里我们可以使用shutil库进行文件移动操作。

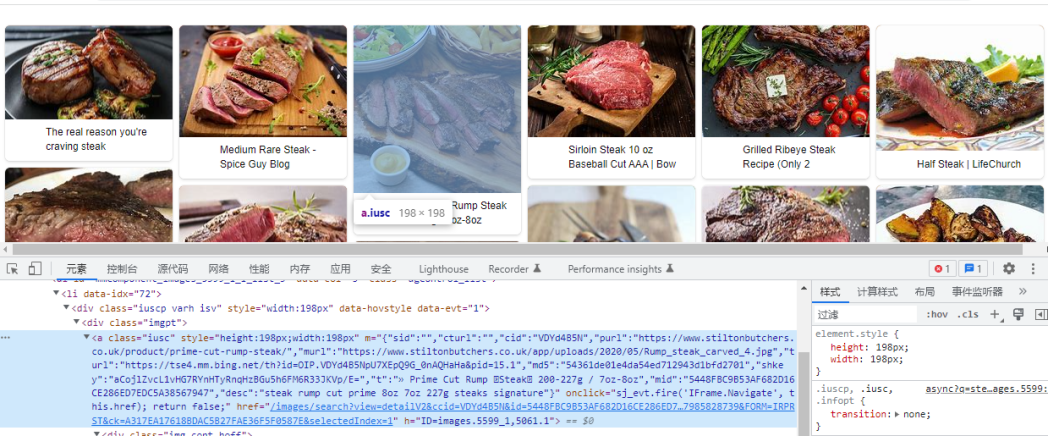
核心代码：shutil.move()

4.2、数据爬取部分程序调试：

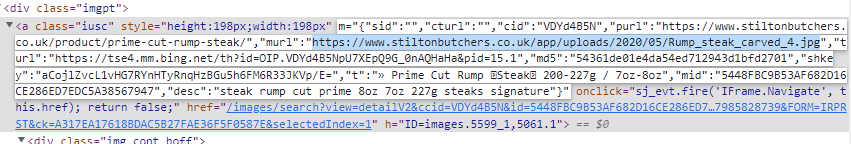
1. 首先找到目标网站缩略图列表然后进行分析。

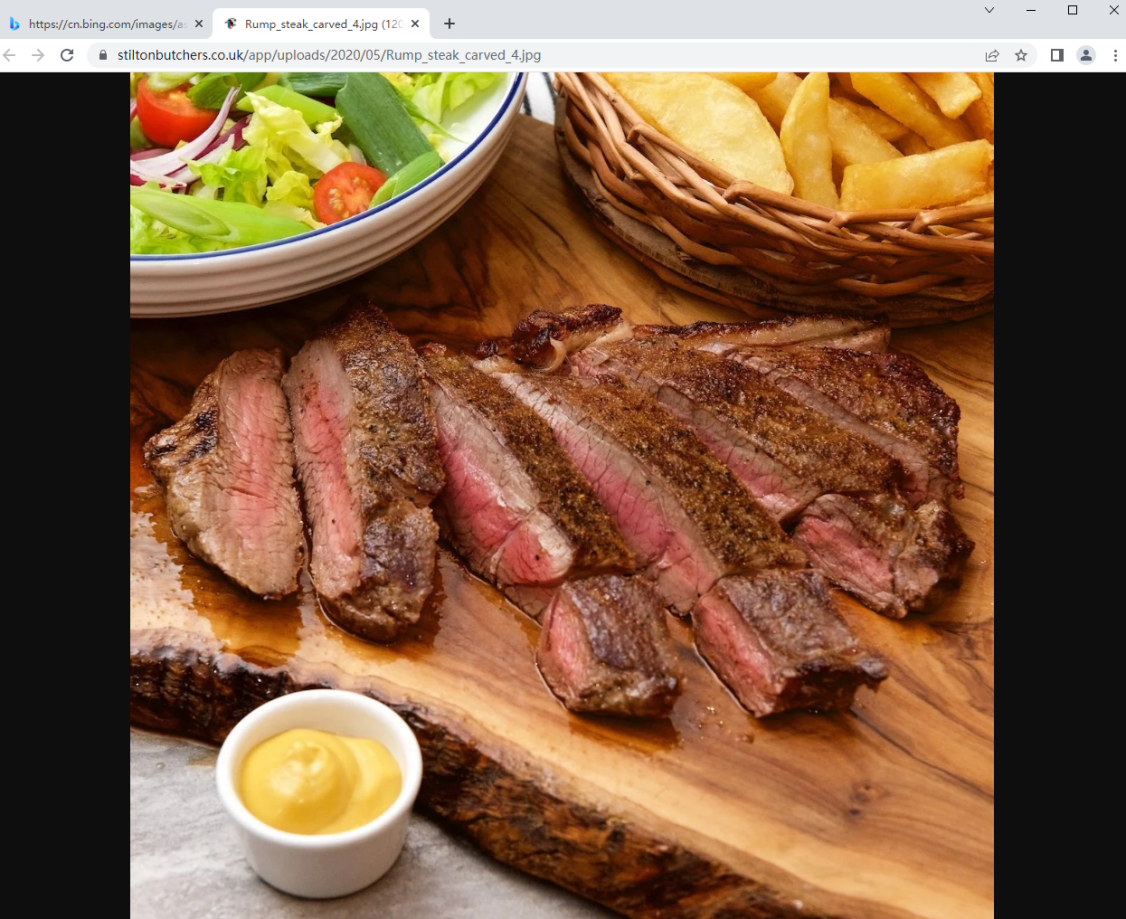


1. 按F12抓取图片所在标签，我们可以发现图片的url在class=’iusc’的a标签中。



1. 经对比几个图片的url我发现在murl中的图片是高清大图，所以我们可以获取murl作为原图的url。



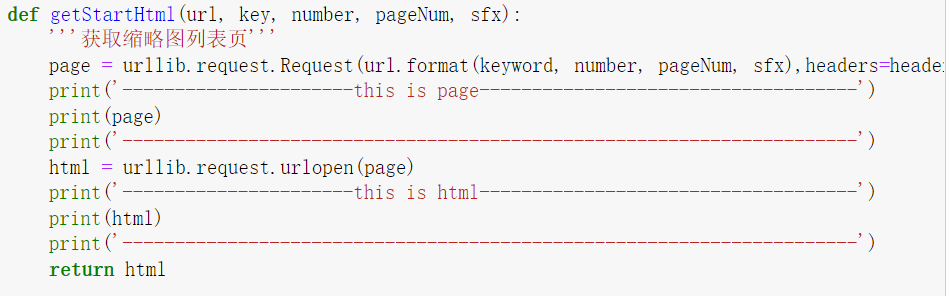


1. 这样我们就可以根据此情况写出匹配这样murl的正则表达式，我们这里只获取后缀为.jpg和.png的图片。

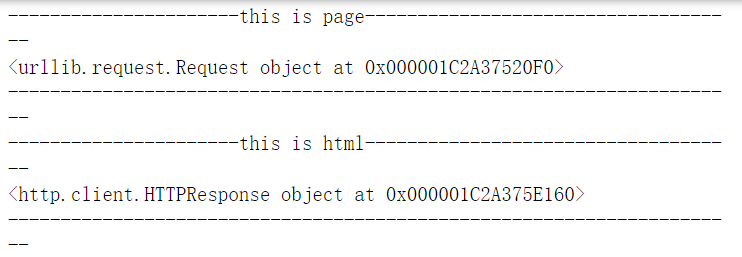
正则表达式：r"\"murl\"\:\"http\S[^\"]+\.(?:jpg|JPG|png)"

1. 接着我们进入程序运行阶段，我们使用jupyter notebook进行编译调试。首先调用getStartHtml（），获取目标网站页面信息，返回html。

关键代码：

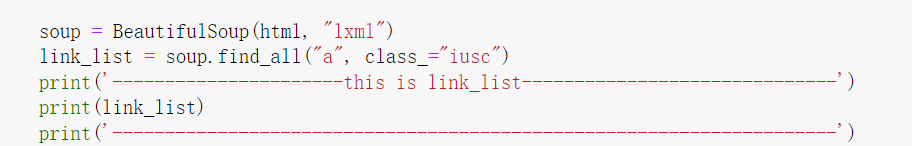


结果:



1. 接着调用findImgUrlFromHtml（），利用正则匹配搜索到所有class=’iusc’的a标签，并将这些a标签保存到一个列表中。

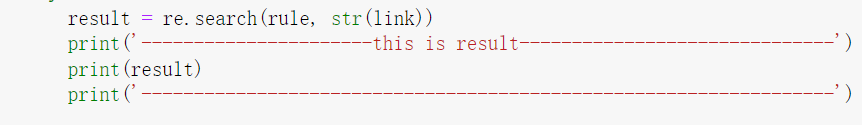
关键代码：



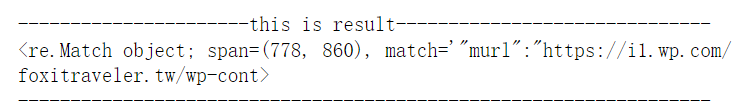
结果：



1. 利用正则匹配找到原图的url，并将结果返回到result.group（）中。

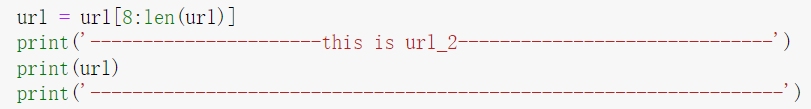
关键代码：

结果:

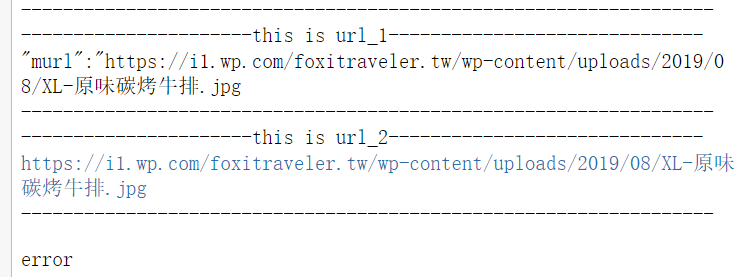


1. 提取图片的有效url,将前八位删掉，若图片链接打不开则做异常处理，报error。

关键代码：

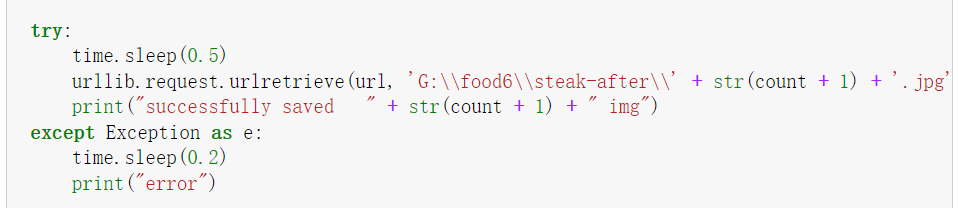


结果：



1. 否则会将图片保存到本地（调用getImage（））。

关键代码：



结果：



10、最终实现爬取目标网站高清大图。

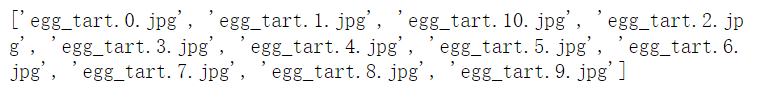
4.3、数据清洗部分程序调试：

1. 图片批量rename操作，首先获取指定文件夹内图片的名字，并存入列表中。

关键代码：

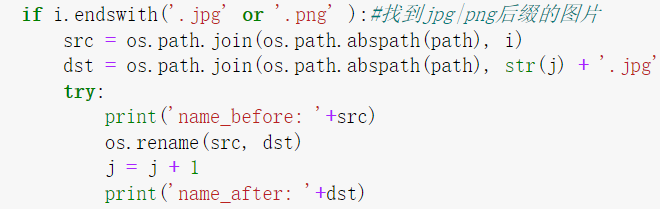


结果：

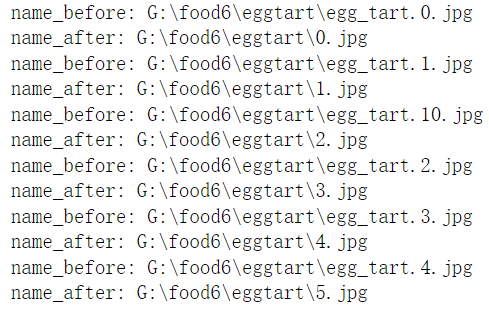


1. 找到.jpg/png后缀的图片，并利用os.rename进行改名。

关键代码：

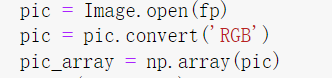


结果：

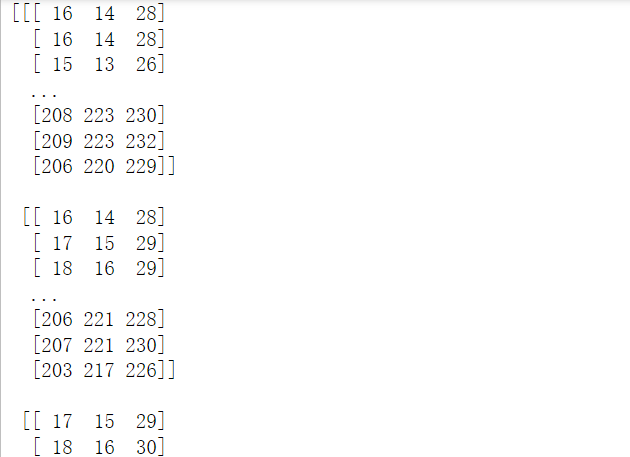


1. 图片批量格式化操作，进行resize和图片格式转换，首先打开目标路径的图片，将其全部转换成’RGB’格式，并将图片按像素点提取到数组中。

关键代码：



结果：



1. 之后通过resize函数批量进行resize操作。

关键代码：



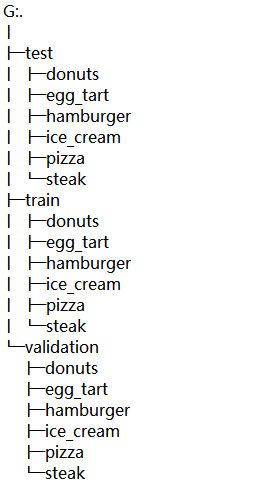
结果：



1. 数据切分。创建测试集、训练集、验证集的文件夹。直接调用os库的创建文件路径功能即可。



1. 同时我们可以生成windows文件目录树来检测一下：

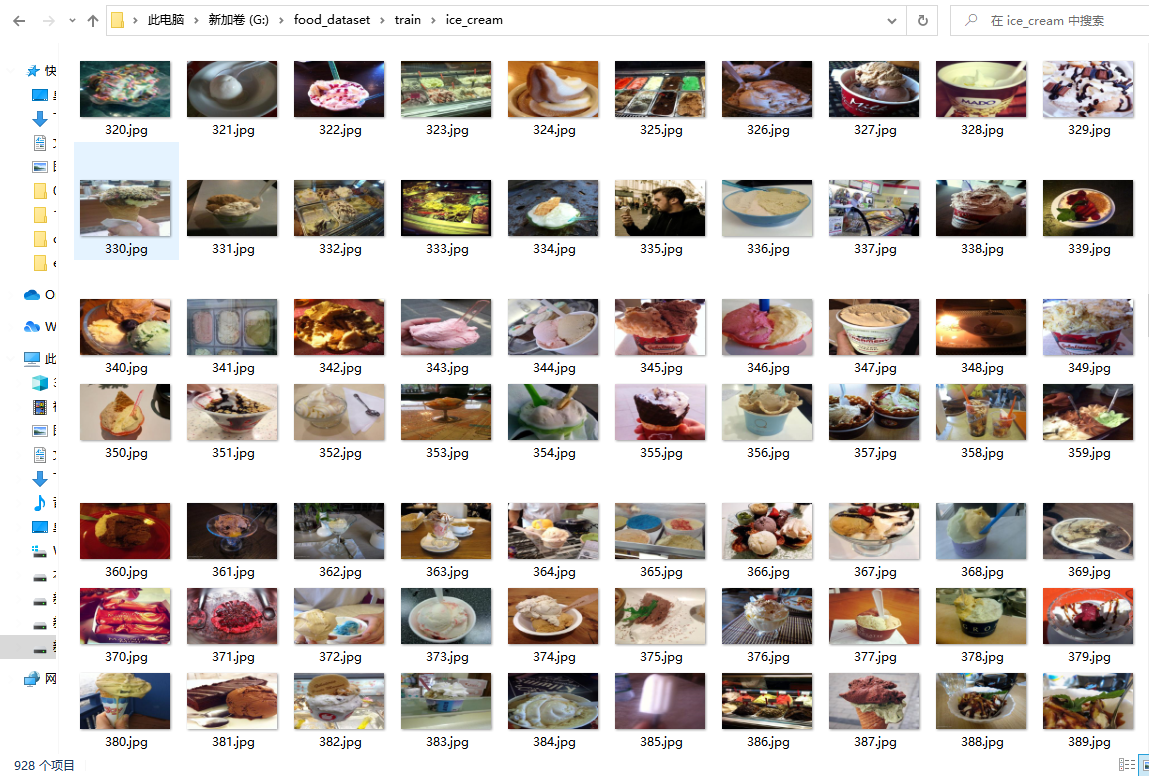


4.4、最终实现效果：

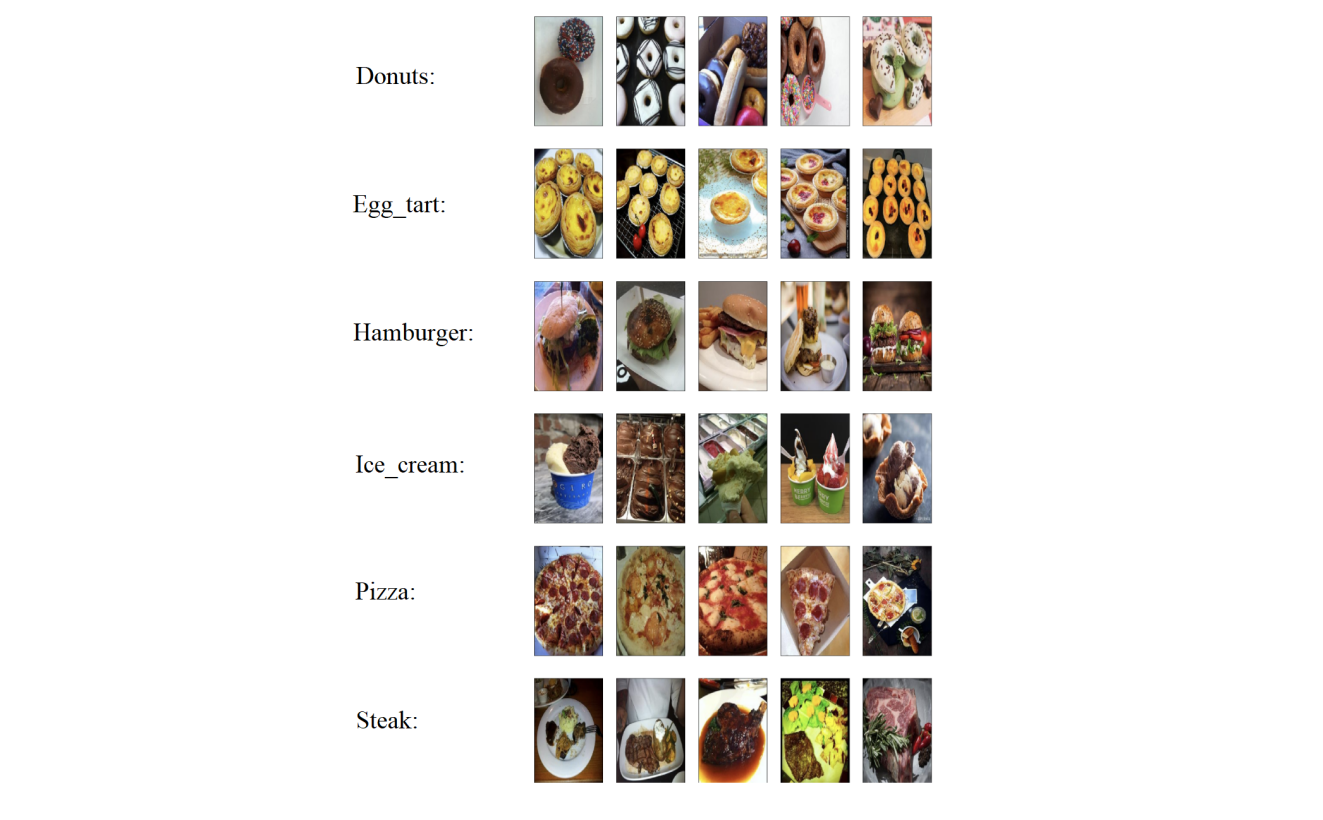
最后经过严格数据清洗每类大概爬取并整理出1100余张高质量图片，并切分成训练集、验证集、测试集交付给模型训练的小伙伴使用。







按类展示数据集样式，实现数据可视化：



**5 系统测试**（小三黑体，居中）*(课程目标支撑3 ; 毕业要求支撑9.2)*

详细写明测试方法、测试描述（bug与修正方法等）、预期结果等；

主要为灰盒测试。首先测试伪头部代理在模拟浏览器的时候是否有效，目标网站是否存在反爬技术。一开始进行爬取的时候总是报异常，后来发现是自己的代理加错了，后来修改了与自己浏览器相匹配的代理解决了问题。在爬虫撰写完毕后，我又进行了图片存取的测试，每存取一张图片到本地就打印一个结果进行反馈，如果异常就是另外一种结果。爬取图片过程中同时也添加了超时处理，因为我发现有些图片的url是无法正常打开的，如果不加入超时处理会白白浪费很多时间。

预期结果为最后每类食物都爬到1300张有效且高质量的数据，并将其进行数据清洗之后交给其他小伙伴进行其余工作。

**6 前沿分析**（小三黑体，居中）*(课程目标支撑6 ; 毕业要求支撑12.2)*

详细写实训过程的心得

目前互联网仅仅是一些文本等多媒体数据的聚合，内容虽然非常有价值，但是程序是无法使用那些没有结构化的数据。在2006年左右，有专家提出的web3.0，语义互联网，知识共享。虽然现在开放API，SOA概念越来越普及，真正语义上的互联网的时代似乎还非常遥远。因此爬虫依然是最重要的手段，一端不断解析，聚合互联网上的数据，另外一端向各种各样的的应用输送数据。数据时代的到来对数据的需求更加强烈。本次项目我只选择了众多爬虫技术中的一种：beautifulsoup。其实还有其他很多种好用的方法，例如scrapy、selenium等。Selenium可以直接模拟浏览器进行类似“人为”的操作，但是对于目前的我来说操作难度有些大，所以只是不断摸索学习。

本次实训虽然只有短短的一个月时间，但是我学到了许多非常有用的新知识。在王老师的带领下我们深入的理解了机器学习和深度学习的概念，尤其是CNN卷积神经网络，在通过和不同的模型对比下我们发现CNN卷积神经网络确实是最好用的方法，诸如朴素贝叶斯、决策树等虽然容易理解但是训练出的模型准确度无法与CNN相比。后来我们又简单学习了python-flask框架，它非常适合类似我们项目这种的轻量级开发，它的使用方法非常灵活而且好用，对于初学者来说只要稍加努力也可以成为“全栈工程师”，其boostrap模板更是让我们非常方便地实现响应式设计。

在这二十几天中我也逐渐了解了企业的大致工作模式，并逐渐的适应起来，坚持每天写日志，并着重看重文档的撰写。文档的撰写至少占到一个项目总比重的百分之七十，大多数情况下都是不能代码先行的，要养成良好的习惯。本次实训既锻炼了我的实践能力又培养了我的团队合作能力，一个人单打独斗再厉害也比不过一个团队的通力合作，希望以后能有更多机会进行类似的学习。