密　级	
编　号	

浙江大學

Zhejiang University

网络基础

网络爬虫及 Python 程序设计

Web Crawler and Python Programming

学　院　信息与电子工程学院
专　业　信息工程
姓　名
学　号　3200105426

2023 年 6 月 14 日

网络爬虫及 Python 程序设计

摘　要

Python 有着实用的模块和库，虽然牺牲了底层性，但却更加方便用于开发小型项目。基于 python 的网络爬虫技术，相比于通用的搜索引擎更具有目的性和灵活性，它能根据选定的抓取目标，有选择的访问万维网上的网页与相关的链接，获取所需要的信息。本文以人民日报新闻爬取和爬取后的保存及查询为研究，实现了一个基于 python 的人民日报新闻文章爬取程序。本论文还阐述了一些网络爬虫实现的常见问题，包括常用的 python 的网络请求、如何解决网页的反爬问题、数据保存写入问题等。

本程序最终可以实现对人民日报（http://paper.people.com.cn/）新闻文章的下载。可以输入要爬取的日期以及结束日期，将这些日期内的文章全部爬取下来，以日期为名自动生成一个主存储目录，爬取到的文章保存写入 txt 文件中, 每个文本的存储名字以日期加序号存储。

关键词：网络爬虫，Python，网络请求

目 录

摘 要 ····· **I**

第 1 章 绪论 ····· **1**

第 2 章 相关技术介绍 ····· **2**

2.1 爬虫的概念 ····· 2

2.2 爬虫的基本流程 ····· 2

2.3 Request 介绍 ····· 2

2.4 Response 介绍 ····· 2

2.5 解决网页反爬问题 ····· 3

第 3 章 设计要求与过程 ····· **4**

3.1 程序目录结构设计 ····· 4

3.2 程序总体结构设计 ····· 4

3.3 分析目标网站 ····· 4

3.3.1 URL 组成结构 ····· 4

3.3.2 分析网页 HTML 结构 ····· 5

第 4 章 代码实现 ····· **6**

第 5 章 运行结果分析 ····· **10**

第 1 章 绪论

随着科学技术的不断发展，信息流通日益方便，信息数据不断膨胀，充斥在各行各业。由于数据非常庞大，所以即使在搜索引擎存在的情况下，搜索结果的准确率也不高，这使得在网上查找关键有效信息也变为一项极具挑战性的复杂任务。我们可以通过基于 python 的网络爬虫技术很好的解决检索问题，首先我们通常使用的通用搜索引擎的目标是尽可能将网络覆盖率增大，其次在数据形式复杂的情况下，对于具有一定结构且信息含量密集的数据，往往不能被很好的搜索出来。而网络爬虫则更具有目的性，能根据选定的抓取目标，有选择的访问万维网上的网页与相关的链接，获取所需要的信息。

将爬虫技术应用于文章的爬取并筛选有效信息，可以节省科研人员时间，提高资源利用率，将有限的时间发挥更大的价值。

第 2 章 相关技术介绍

2.1 爬虫的概念

爬虫是一个自动化数据采集工具，可以自动请求网页、并数据抓取下来，然后使用一定的规则提取有价值的数据。。其背后的基本原理就是爬虫程序向目标服务器发起 HTTP 请求，然后目标服务器返回响应结果，爬虫客户端收到响应并从中提取数据，再进行数据清洗、数据存储工作。

2.2 爬虫的基本流程

1. 发起请求

通过 HTTP 库向目标站点发起请求，即发送一个 Request，请求可以包含额外的 headers 等信息，然后等待服务器响应。这个请求的过程其实可以看做是用程序对我们使用浏览器访问一个 URL 的行为模仿，最终的效果是相同的。

2. 获取响应内容

若服务器能够正常响应，我们会得到一个 Response，Response 的内容便是所要获取的内容，类型可能有 HTML、Json 字符串，二进制数据 (图片，视频等）等类型。这个过程就是服务器接收客户端的请求，进过解析发送给浏览器的网页 HTML 文件。

3. 解析内容

得到的内容可能是 HTML，可以使用正则表达式，网页解析库进行解析。也可能是 Json，可以直接转为 Json 对象解析。可能是二进制数据，可以做保存或者进一步处理。这一步相当于浏览器把服务器端的文件获取到本地，再进行解释并且展现出来。

4. 保存数据

保存的方式可以是把数据存为文本，也可以把数据保存到数据库，或者保存为特定的 jpg，mp4 等格式的文件。这就相当于我们在浏览网页时，下载了网页上的图片或者视频。

2.3 Request 介绍

浏览器发送信息给该网址所在的服务器，这个过程就叫做 HTTP Request。

请求方式的主要类型是 GET，POST 两种，另外还有 HEAD、PUT、DELETE 等。GET 请求的请求参数会显示在 URL 链接的后面，比如我们打开百度，搜索“图片”，我们会看到请求的 URL 链接为 https://www.baidu.com/s?wd= 图片。而 POST 请求的请求参数会存放在 Request 内，并不会出现在 URL 链接的后面，比如我们登录知乎，输入用户名和密码，我们会看到浏览器开发者工具的 Network 页，Request 请求有 Form Data 的键值对信息，那里就存放了我们的登录信息，有利于保护我们的账户信息安全；

2.4 Response 介绍

服务器收到浏览器发送的信息后，能够根据浏览器发送信息的内容，做出相应的处理，然后把消息回传给浏览器，这个过程就叫做 HTTP Response。

Response 包含响应状态、响应头、响应体。响应状态有多种，比如 200 代表成功，301 跳转页面，404 表示找不到页面，502 表示服务器错误；响应头 (Response Headers) 包括内容类型，内容长度，服务器信息，设置 Cookie 等；响应体是最主要的部分，包含了请求资源的内容，比如网页 HTML 代码，图片二进制数据等。

2.5 解决网页反爬问题

通常我们在请求一个网页时，无论是通过哪种请求，发现如果不带 headers 参数一般会出现 403 错误，这种错误的原因是通常网页为了防止恶意的采集数据信息会设置一些反爬措施，从而拒绝爬虫程序的访问。因而通过携带 headers 参数，可以达到模仿浏览器的头部信息进行访问。具体反爬策略：第一遍，先爬取版面目录，将每一个版面的链接保存下来；第二遍，依次访问每一个版面的链接，将该版面的文章链接保存下来；第三遍，依次访问每一个文章链接，将文章的标题和正文保存到本地。

第 3 章 设计要求与过程

实现对人民日报（http://paper.people.com.cn/）新闻文章的下载。可以输入要爬取的日期以及结束日期，将这些日期内的文章全部爬取下来，以日期为名自动生成一个主存储目录，爬取到的文章保存写入 txt 文件中, 每个文本的存储名字以日期加序号存储。本程序需要在 python 下，并且需要下载程序依赖的包才能运行。本程序需要用到的包主要有：requests、bs4、os、datetime。

3.1 程序目录结构设计

程序项目结构非常简单，一个主程序（paweb.py）, 还有是根据日期分类的资源总目录，总目录下自动根据日期生成存储文章的目录，再下面是是具体文章的 txt 文本，每个 txt 存储一篇文章。

(a) 程序结构 (b) 资源存放

图 3-1 程序目录

3.2 程序总体结构设计

该爬虫程序没有用户界面，基于 python 环境，运行在 Windows PowerShell 窗口中，使用流程为：输入需要爬取的开始日期，结束日期、回车后等待爬取即可，爬取完成后会有提示。工作流程为：根据输入的日期拼接 URL，获取当天报纸的各版面的链接列表，再获取报纸版面的文章链接列表，然后解析 HTML 网页，获取新闻的文章内容，获取到文章标题和正文信息后写入到对用的文件中，最后程序结束运行并提示已经爬取完成。

3.3 分析目标网站

3.3.1 URL 组成结构

人民日报网站的 URL 的结构比较直观，基本上什么重要的参数，比如日期，版面号，文章编号什么的，都在 URL 中有所体现，构成的规则也很简单，像这样：

版面目录：http://paper.people.com.cn/rmrb/html/2023-06/12/nbs.D110000renmrb_01.htm

文章内容：http://paper.people.com.cn/rmrb/html/2023-06/12/nw.D110000renmrb_20230612_2-01.htm

在版面目录的链接中，“/2023-06/12/”表示日期，后面的“_01”表示这是第一版面的链接。在文章内容的链接中，“/2023-06/12/”表示日期，后面的“_20230612_2_01”

表示这是 2023 年 6 月 12 日报纸的第 1 版第 2 篇文章，需要注意的是，在日期的“月”和“日”以及“版面号”的数字，若小于 10，需在前面补“0”，而文章的篇号则不必。

了解到这个之后，我们可以按照这个规则，构造出任意一天报纸中人一个版面的链接，以及任意一篇文章的链接。

3.3.2 分析网页 HTML 结构

在 URL 分析中，通过实验发现了网站的页面跳转是通过 URL 的改变完成的，不涉及到 Ajax 这样的动态加载方法。它的所有数据是一开始就加载好的，我们只需要去 html 中提取相应得数据即可。

(a) 标题位置 (b) 正文位置

图 3-2 网页解析

由上图分析后发现，进入文章内容页面之后，文章标题存放在 h1，h2，h3 标签中（有的文章标题只用到了 h1 标签，而有的文章有副标题可能会用到 h2 或 h3 标签），正文部分存放在 id = “ozoom”的 div 标签下的 p 标签里。至此，目标网站的 HTML 页面分析完成。具体爬虫程序分析流程图如下：

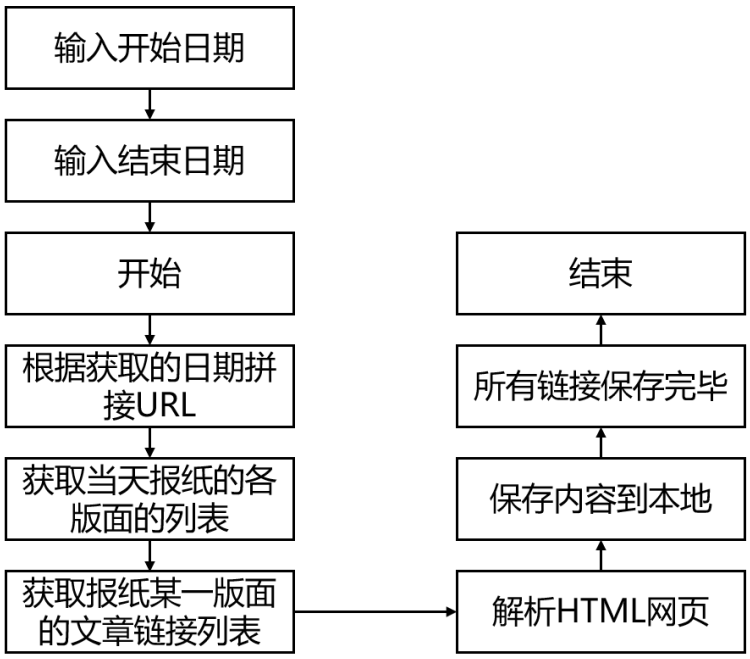

图 3-3 爬虫程序分析流程图

第 4 章 代码实现

代码 4-1 paweb

```
import requests
import bs4
import os
import datetime
import time

def fetchUrl(url):
    # 功能：访问 url 的网页，获取网页内容并返回
    # 参数：目标网页的 url
    # 返回：目标网页的 html 内容
    headers = {
        'accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8',
        'user-agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/68.0.3440.106 Safari/537.36',
    }
    r = requests.get(url, headers=headers)
    r.raise_for_status()
    r.encoding = r.apparent_encoding
    # 返回：目标网页的 html 内容
    return r.text


def getPageList(year, month, day):
    # 功能：获取当天报纸的各版面的链接列表
    # 参数：年，月，日，改变年月日拼接成需要爬取的url
    # 返回当天报纸的各版面的链接列表
    url = 'http://paper.people.com.cn/rmrb/html/' + year + '-' + month + '/' + day + '/nbs.D110000renmrb_01.htm'
    html = fetchUrl(url)
    bsobj = bs4.BeautifulSoup(html, 'html.parser')
    temp = bsobj.find('div', attrs={'id': 'pageList'})
    if temp:
        pageList = temp.ul.find_all('div', attrs={'class': 'right_title-name'})
    else:
        pageList = bsobj.find('div', attrs={'class': 'swiper-container'}).find_all('div', attrs={'class': 'swiper-slide'})
    linkList = []

    for page in pageList:
        link = page.a["href"]
```

```
38          url = 'http://paper.people.com.cn/rmrb/html/' + year
    + '-' + month + '/' + day + '/' + link
39          linkList.append(url)
40      # 返回当天报纸的各版面的链接列表
41      return linkList
42
43
44  def getTitleList(year, month, day, pageUrl):
45      # 功能：获取报纸某一版面的文章链接列表
46      # 参数：年，月，日，该版面的链接
47      html = fetchUrl(pageUrl)
48      bsobj = bs4.BeautifulSoup(html, 'html.parser')
49      temp = bsobj.find('div', attrs={'id': 'titleList'})
50      if temp:
51          titleList = temp.ul.find_all('li')
52      else:
53          titleList = bsobj.find('ul', attrs={'class': 'news-
    list'}).find_all('li')
54      linkList = []
55
56      for title in titleList:
57          tempList = title.find_all('a')
58          for temp in tempList:
59              link = temp["href"]
60              if 'nw.D110000renmrb' in link:
61                  url = 'http://paper.people.com.cn/rmrb/html/'
    + year + '-' + month + '/' + day + '/' + link
62                  linkList.append(url)
63      return linkList
64
65
66  def getContent(html):
67      # 功能：解析 HTML 网页，获取新闻的文章内容
68      # 参数：html 网页内容
69      # 返回结果 标题+内容
70
71      bsobj = bs4.BeautifulSoup(html, 'html.parser')
72
73      # 获取文章 标题
74      title = bsobj.h3.text + '\n' + bsobj.h1.text + '\n' +
    bsobj.h2.text + '\n'
75      # print(title)
76
77      # 获取文章 内容
78      pList = bsobj.find('div', attrs={'id': 'ozoom'}).find_all
    ('p')
79      content = ''
80      for p in pList:
81          content += p.text + '\n'
82      # print(content)
```

```
83
84      # 返回结果 标题+内容
85      resp = title + content
86      return resp
87
88
89  def saveFile(content, path, filename):
90
91      # 功能：将文章内容 content 保存到本地文件中
92      # 参数：要保存的内容，路径，文件名
93      # 如果没有该文件夹，则自动生成
94      if not os.path.exists(path):
95          os.makedirs(path)
96          print('爬取到文章，正在下载中...')
97
98      # 保存文件
99      with open(path + filename, 'w', encoding='utf-8') as f:
100         f.write(content)
101     print('爬取到文章，正在下载中...')
102     print('文章已写入：' + path)
103
104
105 def download_rmrb(year, month, day, destdir):
106
107     # 功能：爬取《人民日报》网站某年,某月,某日的新闻内容，并保存在指定目
        录下
108     # 参数：年，月，日，文件保存的根目录
109     # 保存文件
110
111     pageList = getPageList(year, month, day)
112     for page in pageList:
113         titleList = getTitleList(year, month, day, page)
114         for url in titleList:
115
116             # 获取新闻文章内容
117             html = fetchUrl(url)
118             content = getContent(html)
119
120             # 生成保存的文件路径及文件名
121             temp = url.split('_')[2].split('.')[0].split('-')
122             pageNo = temp[1]
123             titleNo = temp[0] if int(temp[0]) ⩾ 10 else '0'
        + temp[0]
124             path = destdir + '/' + year + month + day + '/'
125             fileName = year + month + day + '-' + pageNo + '-
    ' + titleNo + '.txt'
126
127             # 保存文件
128             saveFile(content, path, fileName)
129
```

```
130
131 def gen_dates(b_date, days):
132     day = datetime.timedelta(days = 1)
133     for i in range(days):
134         yield b_date + day * i
135
136
137 def get_date_list(beginDate, endDate):
138     # 获取日期列表
139     # :param start: 开始日期
140     # :param end: 结束日期
141     start = datetime.datetime.strptime(beginDate, "%Y%m%d")
142     end = datetime.datetime.strptime(endDate, "%Y%m%d")
143
144     data = []
145     for d in gen_dates(start, (end-start).days):
146         data.append(d)
147     # return: 返回开始日期和结束日期之间的日期列表
148     return data
149
150
151 # 主函数: 程序入口
152 if __name__ == '__main__':
153
154     # 输入起止日期, 爬取之间的新闻
155     print('---文章爬取系统---')
156     beginDate = input('请输入开始日期(格式如20220706):')
157     endDate = input('请输入结束日期(格式如20220706):')
158     data = get_date_list(beginDate, endDate)
159
160     for d in data:
161         year = str(d.year)
162         month = str(d.month) if d.month ⩾ 10 else '0' + str(
    d.month)
163         day = str(d.day) if d.day ⩾ 10 else '0' + str(d.day)
164         # 爬取后文章t统一存到这个文件夹,没有会自动创建
165         destdir = "./人民日报"
166
167         download_rmrb(year, month, day, destdir)
168         print('---文章爬取系统---')
169         print("爬取文章完成! ")
170         print('爬取文章时间为: ' + year + '/' + month + '/' +
    day + '的文章已成功写入文件夹中! ')
171         print('---文章爬取系统---')
```

第 5 章 运行结果分析

1. 开始运行程序，输入爬取文章的开始日期和结束日期

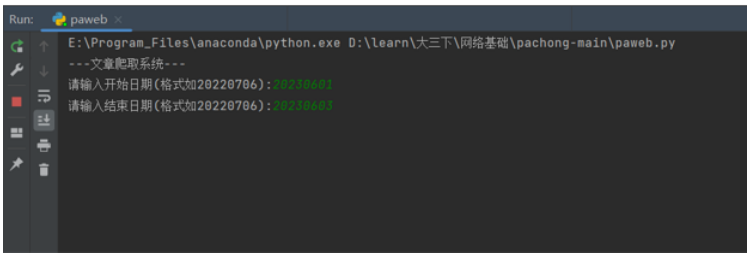

图 5-1 运行图

2. 程序运行，爬取完成

图 5-2 运行图

3. 写入本地效果

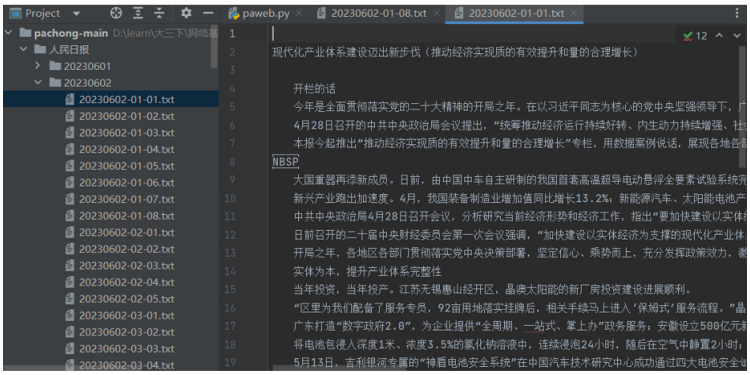

图 5-3 运行图