爬虫原理+爬取步骤

什么是爬虫

请求网站并且提取数据的自动化程序

爬虫的兴起

随着互联网的发展 数据资源变得非常丰富且容易搜索 人们发现从网页上找到他们想要的信息是一件非常简单的事情，他们通常分布在大量的网站上。但另一个问题出现了，当他们想要数据的时候，并非每个网站都提供下载按钮，如果进行手动复制显然是非常低效且乏味的。

网络爬虫就诞生了 。网络爬虫实际上是由网页机器人/爬虫驱动的，其功能与搜索引擎相同。简单来说就是，抓取和复制。唯一的不同可能是规模。网络数据抓取是从特定的网站提取特定的数据，而搜索引擎通常是在万维网上搜索出大部分的网站。

特别是现在的 数据分析 人工智能 大数据的兴起 对于数据的需求越来越高 爬虫蓬勃发展

数据从哪里来

1. 企业产生的用户数据
2. 数据平台买数据
3. 公开数据
4. 数据管理咨询公司
5. 爬虫爬取网络数据

爬虫的分类

通用爬虫

- 将网页下载到本地

聚焦爬虫

- 从网页中提取我们想要的数据

爬虫的本质

模拟客户端发起请求 接收响应

原则上 只要是浏览器(app)能看到的 都可以爬取 万物皆可爬

爬虫不能做的事情

1. 爬虫的频次要控制

- 不要把人家服务器搞崩

2. 隐私数据不要爬

- 比如：身份证 手机号 住址 等等

3. 不正当竞争/搬运盈利

- 比如：爬取人家的数据 去卖钱 (做大了人家会找你麻烦)

4. robots协议

- 规定了哪些能爬，哪些不能爬，谁可以爬
- 只是一个不成文的约定，一没有法律效应，二也没有强制作用

爬虫≠黑客 爬虫需谨慎

爬虫该怎么写

1. 明确目标

- 要爬取的网站的url
- url
    - 统一资源定位符： 是用于完整地描述Internet上网页和其他资源的地址的一种标识方法。
    - https://www.runoob.com/python/python-tutorial.html
    - https（协议）
        - http
            - 超文本传输协议
            - 是一种发布和接收HTML页面的方法
            - 80端口
        - https

        - ssl 安全套接层
            - 主要用于Web的安全传输协议，在传输层对网络连接进行加密，保障在Internet上数据传输的安全。
        - 443端口
    - www.runoob.com(域名)
    - python/python-tutorial.html path 路由

浏览网页的基本过程

- 比如我们在浏览器地址栏输入：http://www.baidu.com，回车后会浏览器会显示百度的首页。
- 这是一个发起请求获取响应的过程
- 这段过程发生了以下四个步骤
    - 浏览器通过 DNS服务器 查找域名对应的 IP地址;
    - 向 IP地址 对应的 Web服务器 发送请求
    - Web服务器 响应请求，发回 HTML页面
    - 浏览器解析 HTML内容，并显示出来
- DNS服务器
    - DNS是计算机域名系统 (Domain Name System 或Domain Name Service) 的缩写，由解析器和域名服务器组成的。
    - 域名服务器是指保存有该网络中所有主机的域名和对应IP地址，并具有将域名转换为IP地址功能的服务器。
    - 一个域名必须对应一个IP地址，而一个IP地址不一定会有域名。

2. 爬取

- 发起请求 (当用户在浏览器的地址栏中输入一个URL地址并按回车键之后，浏览器会向HTTP服务器发送HTTP请求)
    - request请求
        - 请求方式
            - http请求主要有两种
                - get

                - 从服务器获取在指定页面信息
                - GET请求参数都显示在URL上，服务器根据该请求所包含URL中的参数来产生响应内容。 "Get" 请求的参数 是URL的一部分。
            - post
                - 向服务提交数据并获取页面信息
                - "POST"请求的参数 不在URL中，而在请求体中。
        - 请求的url
        - 请求头
            - 包含请求时的头部信息
            - 比如UA host cookies
            - 请求头数据是能相信的
        - 请求体
            - 请求额外携带的数据
            - 比如：页数
    - User-Agent
        - 身份标识
- 获取响应(response)
    - 响应状态
        - 响应码
            -
                - 100~199`：表示服务器成功接收部分请求，要求客户端继续提交其余请求才能完成整个处理过程。
                - `200~299`：表示服务器成功接收请求并已完成整个处理过程。常用200（OK 请求成功）。
                - `300~399`：为完成请求，客户需进一步细化请求。例如：请求的资源已经移动一个新地址、常用302（所请求的页面已经临时转移至新的

url）、307和304（使用缓存资源）。
    - `400~499`：客户端的请求有错误，常用404（服务器无法找到被请求的页面）、403（服务器拒绝访问，权限不够）。
    - `500~599`：服务器端出现错误，常用500（请求未完成。服务器遇到不可预知的情况）。
  - 响应头
    - 服务器告诉我们的信息
    - 比如内容类型 内容长度 服务器信息
    - 不能全部相信
  - 响应体
    - 请求资源的内容 比如网页的HTML

3. 取

4 存

开发工具

Elements（元素面板）

- 操纵DOM和CSS来重演您网站的布局和设计

Console（控制台面板）

Sources（源代码面板）

Network（网络面板）**重点**

- 从发起网页页面请求Request后得到的各个请求资源信息（包括状态、资源类型、大小、所用时间等）
- preserce log
  - 保存日志
  - 勾选上每次刷新不会清空前面的请求 不勾选每次请求都会刷新清空上次的请求
- disable cache

    - 勾选每次就会重新请求服务器得到资源 不勾选是拿内存中已经缓存的资源 不会再去请求服务器有一些不会进行网络请求

常用的请求报头

- Host (主机和端口号)
  对应网址URL中的Web名称和端口号，用于指定被请求资源的Internet主机和端口号，通常属于URL的Host部分
- User-Agent (浏览器名称)：标识客户端身份的名称，通常页面会根据不同的User-Agent信息自动做出适配，甚至返回不同的响应内容。
- Accept (传输文件类型)
  Accept：指浏览器或其他客户端可以接受的MIME（Multipurpose Internet Mail Extensions（多用途互联网邮件扩展））文件类型，服务器可以根据它判断并返回适当的文件格式。
- Referer (页面跳转来源)
  表明产生请求的网页来自于哪个URL，用户是从该 Referer页面访问到当前请求的页面。这个属性可以用来跟踪Web请求来自哪个页面，是从什么网站来的等。
- Accept-Encoding（文件编解码格式）
  指出浏览器可以接受的编码方式。编码方式不同于文件格式，它是为了压缩文件并加速文件传递速度。浏览器在接收到Web响应之后先解码，然后再检查文件格式，许多情形下这可以减少大量的下载时间。
- Accept-Language（语言种类）
  指出浏览器可以接受的语言种类，如en或en-us指英语，zh或者zh-cn指中文，当服务器能够提供一种以上的语言版本时要用到。
- Accept-Charset（字符编码）
  指出浏览器可以接受的字符编码。
- Cookie：浏览器用这个属性向服务器发送Cookie。Cookie是在浏览器中寄存的小型数据体，它可以记载和服务器相关的用户信息，也可以用来实现模拟登陆，之后会详细讲

常用的响应报头

- Cache-Control
  这个值告诉客户端，服务端不希望客户端缓存资源，在下次请求资源时，必须要从新请求服务器，不能从缓存副本中获取资源。
- Content-Encoding:gzip
  告诉客户端，服务端发送的资源是采用gzip编码的，客户端看到这个信息后，应该采用gzip对资源进行解码。

- Content-Type：text/html;charset=UTF-8:资源文件的类型，还有字符编码，客户端通过utf-8对资源进行解码，然后对资源进行html解析。通常我们会看到有些网站是乱码的，往往就是服务器端没有返回正确的编码。
- Date：Sun, 21 Sep 2016 06:18:21 GMT
  这个是服务端发送资源时的服务器时间，GMT是格林尼治所在地的标准时间。http协议中发送的时间都是GMT的，这主要是解决在互联网上，不同时区在相互请求资源的时候，时间混乱问题
- Server：Tengine/1.4.6
  这个是服务器和相对应的版本，只是告诉客户端服务器的信息。

模块下载

pip install requests

- 爬虫的请求库 目前用的最多最主流的

作用

- 发起网络请求
- 返回响应数据

requests 的底层实现其实就是 urllib3

get请求

requests.get(url='http://httpbin.org/get',headers=headers) # get请求

get请求传参

- 1. params={'age': '1888'}
- 2. ?id=1&age=18
  - 问号前面的是路由 后面的是参数
  - 多个参数&连接

伪造请求头

```
headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/94.0.4606.61 Safari/537.36",
    }
```

- 伪装身份信息

post请求

requests.post(url='http://httpbin.org/post?id=1&age=18', headers=headers)

post请求传参

- data={'username':'nanfeng'}
- 不推荐在url地址栏写参数 这种是get写法
- 如果参数是密码 地址栏写参数会暴露密码数据
- post请求一般参数会在body里
- 不需要去纠结到底怎么传递参数 用哪种方法去传递 我们只要看人家网站是怎么传的就行 爬虫是模拟用户发起请求！！！

response的常用属性

查看响应内容，response.text 字符串数据

查看响应内容，response.content返回的字节流数据 二进制数据

查看完整url地址 response.url

查看响应头部字符编码 response.encoding

查看响应码 response.status_code

查看响应头 response.headers

查看请求头 response.request.headers

百度首页

```
'''
        需求
            获取百度首页的响应数据
        实现
            1. 明确目标url
                https://www.baidu.com/
            2. 发起请求 获取响应
                我们没有伪装身份 被百度抓到了
                伪装一下身份 骗过百度
```

```
'''
import requests

# 伪装成真人
headers = {
    # "Accept-Language": "zh-CN,zh;q=0.9,ee;q=0.8",
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.102
Safari/537.36'
}

response = requests.get(url='https://www.baidu.com/',
headers=headers)  # 发起请求
print(response.encoding)  # 查看响应头部字符编码
response.encoding
response.encoding = 'utf-8'
print(response.encoding)
print(response.status_code)
# print(response.text)  # 查看响应内容，response.text 字符串数据
print(response.headers)  # 查看响应头 response.headers
print(response.request.headers)  # 查看请求头
response.request.headers
print(response.url)  # 查看完整url地址 response.url
print(response.content.decode())  # 查看响应内容，
response.content返回的字节流数据 二进制数据
```

百度词语联想

```
'''
        需求
            爬取百度词语联想的内容
        1. 明确目标url
            找到词语联想存在哪个请求的响应里面
            他是异步加载的 我们抓包选择XHR
            https://www.baidu.com/sugrec?pre=1&p=3&ie=utf-
8&json=1&prod=pc&from=pc_web&sugsid=36550,38106,38093,38051,3
7907,38146,38265,38179,38173,38231,38035,37937,26350,38100,38
008,37881&wd=%E5%8D%97%E9%A3%8E&req=2&csor=2&cb=jQuery1102031
60503682954243_1677161732483&_=1677161732484
```

```
            https://www.baidu.com/sugrec?pre=1&p=3&ie=utf-
8&json=1&prod=pc&from=pc_web&sugsid=36550,38106,38093,38051,3
7907,38146,38265,38179,38173,38231,38035,37937,26350,38100,38
008,37881&wd=%E5%8C%97%E9%A3%8E&req=2&csor=2&pwd=%E5%8D%97%E9
%A3%8E&cb=jQuery110203160503682954243_1677161732483&_=1677161
732485
        2. 发起请求 获取响应
'''
import requests
# 伪装成真人
headers = {
    # "Accept-Language": "zh-CN,zh;q=0.9,ee;q=0.8",
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.102
Safari/537.36'
}
url ='https://www.baidu.com/sugrec?pre=1&p=3&ie=utf-
8&json=1&prod=pc&wd=%E5%8C%97%E9%A3%8E'
response = requests.get(url=url, headers=headers)  # 发起请求
print(response.text)
```

网页加载的两种方法

同步加载

- 改变网址上的某些请求参数会导致网页发生改变，例如：翻页 网页会发生改变 加载页面

异步加载

- 改变网址上的请求参数不会使网页发生改变，例如：翻页后网址不会发生变化 局部的刷新数据

反爬虫的原因

不遵守规范的爬虫会影响网站的正常使用

网站上的数据是公司的重要资产

爬虫对网站的爬取会造成网站统计数据的污染

反爬分类

基于身份识别的反爬

基于爬虫行为的反爬

基于数据加密的反爬

爬虫概念

爬虫

- 使用技术手段 获取网站数据

反爬虫

- 阻止爬虫获取自己网站数据

误伤

- 反爬虫的过程中 错误的将普通用户识别成爬虫

拦截

- 成功阻止爬虫访问