爬虫

2017 年 8 月 23 日

知乎—爬虫

0.1 模拟登录

1. 知乎爬取时必须带上header。

2. _xref 可以从登录网页的源码中提取，但是提取和登陆的网页必须是同一个。urllib2.urlopen 是打开可能不同的，可以使用requests库建立session。

0.2 爬取follow，问题等内容

【部分知乎问题已有答案了】

知乎网页是动态加载问题，怎么解决？以爬取问题或关注人为例，通过开发者工具中的Network – XHR / Js 选项，向下拉取页面，可以看出网页在加载时是通过offset 和start_offset 来控制起始的爬取网页，来获取新的内容（在XHR中网页不是在类似batch的请求中，**注意：** 如果要爬取全部内容start_offset = 0 而不是3，offset 与显示的不同），然后复制出URL备用。

编写header, 除了基本header外，必须还要有authorization ，authorization貌似在不同的爬取需求中都是相同的，但是header必须包含它。打开网页后的内容好像是json 格式，使用json.loads(url.content) 将内容转化为字典形式。

通过正则表达式或是BeautifulSoup 可以提取需要的内容，在提取速度和存储方式上，文件操作等方面有待加强。

0.3 多线程

0.3.1 反爬措施防止被屏蔽

使用多线程多次爬取后知乎服务器貌似会屏蔽访问，不知道是IP 的问题还是什么，如果还能在浏览器正常打开知乎的话可以采用更换headers中authorization 的方法进行访问。

或是使用IP代理？

0.3.2 多线程的顺序问题

多线程或多进程由于是并发，在访问网页时不能按Page 顺序访问，在下载排序命名时也会顺序错乱。直接采用**全局变量**并加**多线程，多线进锁**，这样虽然会影响Class 的独立性但是可以使得顺序不错乱且按照顺序保存。

【标号顺序不乱但是内容顺序仍有可能混乱，不一定的按照网页内容了。】

0.4 多进程

0.4.1 Windows 下多进程无法运行

SAD −_−

多进程程序在Windows下仍无法运行，在Linux下运行正常。

0.4.2 多进程全局变量共享

global 变量和Queue模块中的Queue() 在全局变量中貌似不起作用，使用multiprocessing 中的各个模块来在进程中同步信号。

```
from multiprocessing import Process, JoinableQueue,Lock ,Queue ,
                                    Pool , Value
counter = Value('i', 0)
counter.value += 1
```

0.4.3 使用进程池可以退出

当爬取网页太多时，进程会也不停止也不报错。尝试使用进程池可以正常退出。

```
pool  = Pool( multiprocessing.cpu_count() )
for i in xrange( 2 * multiprocessing.cpu_count() ) :
        pool.apply_async( func , args = ( ) )
```

```
pool.close()
pool.join()
```

Scrapy—爬虫

0.5 Windows 下安装问题

windows 下anaconda 安装会出现

```
from cryptography.hazmat.bindings._openssl import ffi, lib
ImportError: DLL load failed
```

: 操作系统无法运行%1。

删除了window/system32/ 的libeay32.dll和ssleay32.dll，不过我的建议是重命名，直接后面加_bak 就好，免得出什么问题。

0.6 Scrapy

1. Scrapy 是一种爬虫框架/ 爬虫引擎，你需要在里面填上爬虫函数，并对数据下载，提取，处理，存储。

2. Scrapy 中所以爬虫都继承自scrapy.spiders.Spider。另外有别的作用更加细分的爬虫如Crawlspider 通过设置自身的rules 从爬去的网页中提取URL 继续爬取。

3. Scrapy 调试在Shell 中，开发的话应该掌握。Xpath() 语法用于对数据进行提取，类似正则。

4. items.py 中定义各个数据的存储容器。pipeline.py 进行对item 数据的查重，丢弃，验证，保存工作。

0.7 Tips of Scrapy

0.7.1 不使用命令行启动

在根目录下创建main.py

```
from scrapy import cmdline
cmdline.execute('scrapy crawl Spider_name'.split())
```

0.7.2 爬虫开始，结束时传递参数

Scrapy 默认在开始结束时运行：

```
def open_spider(self, spider):
def close_spider(self, spider):
```

与Spider.py 中的Super 结合使用

```
super(Spider_name,  self).__init__()
```