Super Share

(基于 python 的资源搜索系统)

姓名	学号	Github 账号	分工	成绩
杨飞龙	201592009	2296685742@qq.com	组长	
封振华	201592114	694350829@qq.com	成员	
张劭华	201592356	nosiysilence@icloud.com	成员	
金秋羽	201592462	1776934883@qq.com	成员	
王悦	201592044	332401422@qq.com	成员	

项目网址: https://github.com/JasenYang/super_share

项目部署网址: http://yangfeilong.top:666

2018 年 1 月

大连理工大学

目　录

一．引言

1.1 背景

虽然当前人们生活在互联网的环境中， 我们有很多的资源可以获取、利用。但是， 在实际中， 我们也渐渐发现， 互联网上也充斥着很多的无用资源，我们大部分情况下需要花费大量的精力去排除这些干扰因素。

举一个很常见的例子， 目前百度云盘的分享技术确实很好， 方便人们共享资源。 然而， 应用中， 很多分享链接却是已经失效的， 但是任然存在于互联网的大环境中。 这给我们搜索资源带来了极大的干扰。 因此， 我们希望， 这个项目可以帮助大家搜索到真正有效、有用的资源， 通过 python 的爬虫机制， 代替人们自己去排除垃圾信息的干扰。

1.2 项目的特点

Super share 是一个给人们提供方便获取共享资源的途径的项目。旨在丰富人们的生活， 让人们最大化利用互联网得到自己需要的资源。

我们针对实际情况将 super share 项目分为两大模块:

i. 云盘资源分享链接自动获取和识别:

通过输入关键字， 可以自动搜索到所有网上已经存在的、相关的百度云分享链接，并且可以确保这些链接是有效的。 如果某些链接是需要密码的， 则同时获取密码所在的网页链接。

ii. 共享账号的自动获取和识别:

网上存在的有诸如 迅雷会员账号分享、优酷会员账号分享等等，有些是有用的，然而有些却是无用的。 针对这个， super_share 项

目完成自动获取网上存在的会员账号，并且识别哪些是无用的、哪些是有用的， 然后回馈给用户。

1.3 项目的意义

我们认为， super share 项目最大的意义就在于使用机器代替人们做那些具有重复性的工作，这正是信息化时代特征的体现。在信息量如此庞大的当下， 我们需要利用一些机器化工具来处理本身就不需要人来完成的任务。Python 作为一门脚本语言， 完美的完成了这个任务。 在这个项目中， 我们主要使用到了 python 的爬虫以及模拟访问、登陆技术。 以及大量使用 python 的正则匹配功能来协助我们得到需要的信息。

二．用途

2.1 适用对象

我们项目本身就属于一种服务型的应用， 所以， 我们打算先小范围推广使用（在本校师生范围内），如果大家都认为我们的项目很有意义、 非常认可我们的成果， 就继续完善，通过 git 平台集大家之力共同打造一款应对大数据时代的信息筛选工具。

2.2 功能

我们依据模块划分功能， 在项目的初期阶段， 打算深入挖掘个别模块。 因此， 对于云盘方向， 我们重点选取百度云盘为目标， 从网上众多网盘分享链接中得到我们真正需要的、 真正有效的链接。 对于 VIP 账号分享方向，我们拟选取迅雷和优酷两个平台作为代表， 尝试使用验证登陆功能区分账号的有效和无效性。

三．运行环境

3.1 硬件设备

Super share 项目最终的成果是展现于网页端的。 由于网页的设计并没有考虑响应式布局设计，因此在手机端访问我们网站的体验是很差的，推荐在电脑的浏览器端访问我们项目。

同时， 我们的项目已经部署到远程服务器上， 访问网址：

http://yangfeilong.top:666 即可。

3.2 支持软件

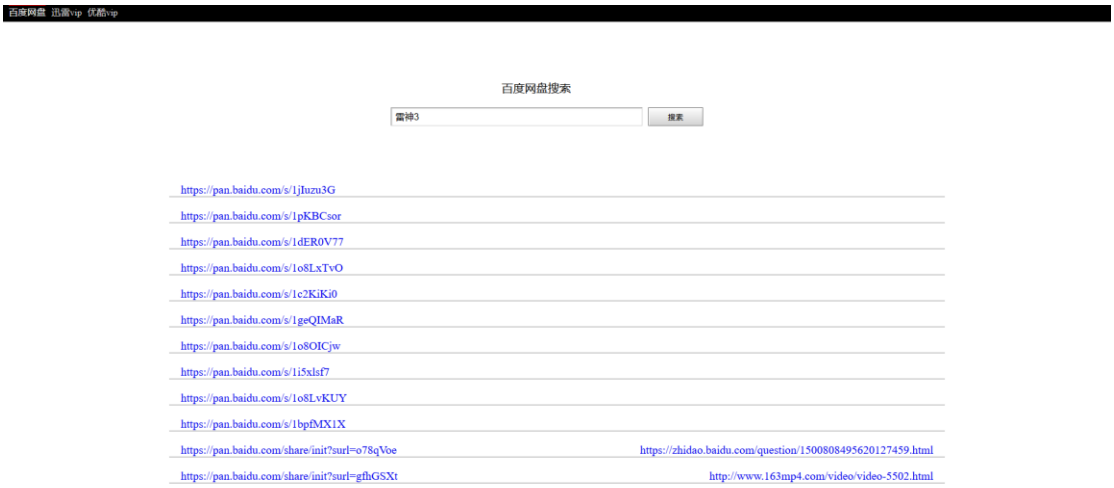
考虑到浏览器兼容性问题， 推荐使用火狐浏览器。 在测试中， 谷歌浏览的表现效果偶尔出现了问题。 其余的浏览器测试情况未知。

四．使用过程

4.1 操作界面中主要对象使用说明

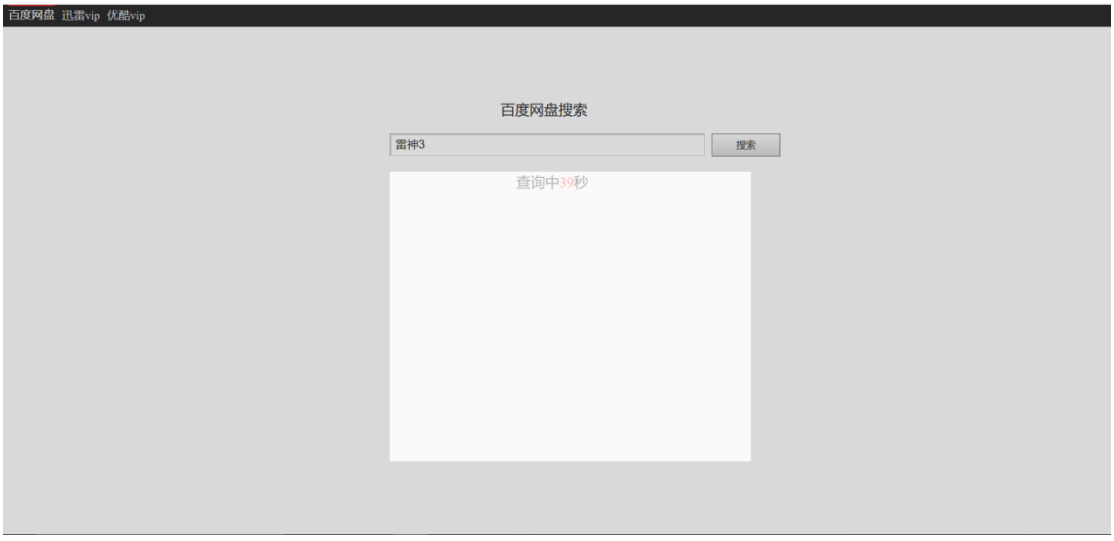
a. 使用云盘搜索功能

在使用云盘搜索中，我们可以得到的数据有两种类型。 第一种是无需密码的分享链接； 第二种是需要密码的分享链接。 本来的计划是， 如果某些链接是需要密码的， 则相应的将密码匹配链接提取出来， 但是后来在实现过程中发现， 这种作为是无法靠正则匹配或其他相关技术完成的。 所以， 我们最终决定， 对于需要密码的链接， 我们将该链接所在的网页对应的 url 地址提取出来， 提供给用户， 由用户自己取找。 最终的结果如下图所示：



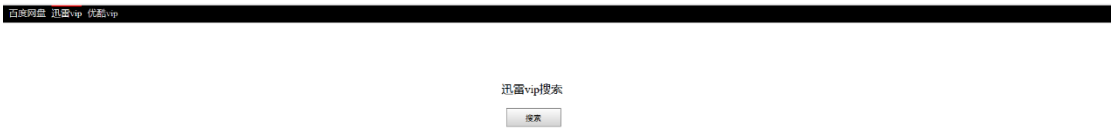
其中， 最后两栏是需要密码的链接。通过访问右侧链接， 即可找到我们需要的密码。 最关键的是， 我们保证了展现出来的云盘链接是经过我们层层筛选出来的、都是有效的。 不存在当用户点击某一个分享链接后，会显示链接已经失效或者链接不存在的问题。

也正是由于我们对于所有链接都进行验证性判断，所以整个的查询过程比较缓慢（即使实现时采用多进程并发技术） ， 因此我们特定在用户查询中显示一个倒计时。 我们测试过， 平均一次查询大约在 30 秒左右， 考虑到网速， 查询时间最长也不会超过 40 秒（代码中有对于 requests 请求有 timeout 判定） 。因此， 查询等待的效果图如下:

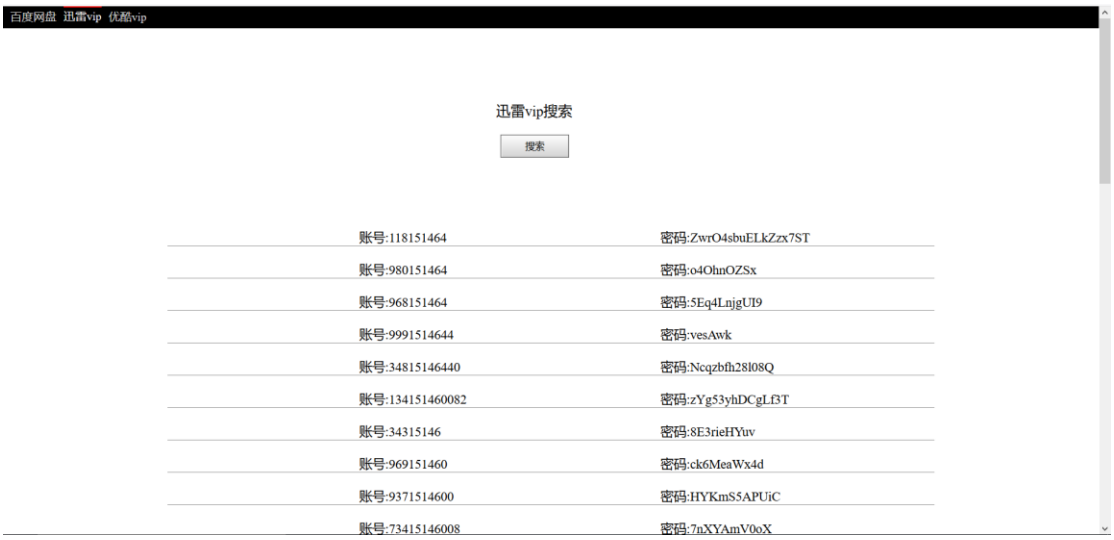


b. 使用 VIP 账号搜索功能

查询迅雷 vip 账号的操作和优酷 vip 账号的操作都是一样的。 在对应的主页面中，点击查询按钮，即可实时查询网上已经有人分享的 vip 账号。



查询结果:



但是由于迅雷和优酷平台的安全级别比较高， 我们尝试自定 request 请求登陆均失败。 最终虽然使用 selenium 自动化打开火狐浏览器成功模拟登陆， 但是这样的效率实在是太慢。 我们测试过， 如果要将我们爬取到的账号全部验证一遍，几乎需要十几分钟(还是在完全不考虑性能，经可能开最多的进程的情况下） 。 所以， 我们对于 vip 账号的爬取最

终只停留在得到账号的程度， 并不能保证账号都是真实有效的。 这个问题， 在小组内商议， 决定等到以后积累足够的知识后在继续解决。

4.2 项目预期的效果

理想情况下， 预期的效果应该是云盘链接的密码应该也是由机器自动获得的， 并且 vip 账号应该是有效的， 网站也是可以自适应手机端的。 但是由于技术匮乏、 时间有限各个方面原因导致项目有点小瑕疵， 这些在日后都会继续完善的。

五． 项目中遇到的问题以及解决方案

5.1 爬虫过程中某些链接存在重定向问题

理想状态下，我们得到的 url 地址应该可以在请求之后得到 response 直接就是我们想要的页面。 但是， 大部分情况下， 这都存在着一个重定向情况。 最常见的有: http 重定向成 https, 百度搜索得到的链接需要重定向成真正的地址等等。 这就需要我们对于返回的 response 的 statues 进行判断。 如果 status 的值为 307 或者 302， 那么就表示这个 response 中包含 Location 属性，而 Location 属性则是跳转的地址。

5.2 请求响应时间过长

由于整个查询的结构是要等到所有的子查询(进程)都结束之后才会返回，所以最慢的请求决定了整个查询所花费的时间。 为了避免由于某一个请求等待过长的时间而导致整个查询变得异常缓慢， 特添加 timeout 超时机制， 避免过度等待。

5.3 关于 flask 框架的一些问题

最终将 flask 框架制作的 web 应用部署的服务器上后才发现， flask 框架不允许外网访问， 最终只得采用 nginx 转发技术。 外网通过访问 nginx 服务器， 然后 nginx 服务器将请求转发到 flask 应用上。

六． 项目心得体会

总结一下， 整体的项目架构以模块化设计思想为主， 单元测试为辅构成的。 但是， 由于时间仓促， 项目开始之前只是储备不够， 导致很多东西都是现学现用， 所以整体代码结构前期比较清晰，然后到后来则有点杂乱。 幸好即使调整， 不至于到代码可复用性差、不可维护的地步。