**Python爬虫数据获取分析**

**目录**

[（一）目标 2](#_Toc12660)

[（二）数据对象及其选择原因 2](#_Toc18904)

[（三）数据收集思路 2](#_Toc24297)

[（四）微博帖子结构及爬取分析 3](#_Toc3887)

[1.网页端微博操作流程分析 3](#_Toc20830)

[①打开微博首页并登陆——搜索框输入搜索关键词，进行搜索——得到相关发帖信息 3](#_Toc28024)

[②分析帖子具体网址拼接规律 4](#_Toc7721)

[③打开具体帖子网页，进行网页结构以及信息获取机制分析 6](#_Toc13903)

[2.微博网页爬取的注意点 7](#_Toc924)

[①微博的反爬机制： 7](#_Toc23843)

[②微博帖子内容的处理 7](#_Toc17462)

[（五） 爬取结果总结说明 8](#_Toc19510)

[（六） 数据的不足之处 8](#_Toc29696)

**图目录**

[图 1：必须登录获取完整内容 3](#_Toc8115)

[图 2：新建搜索结果浏览器窗口 3](#_Toc32257)

[图 3：无法直接呈现留言具体信息 3](#_Toc22034)

[图 4：必须点击跳转具体帖子网页 4](#_Toc15776)

[图 5：前两条帖子的具体网址 5](#_Toc7589)

[图 6：所有帖子的大div标签 5](#_Toc18363)

[图 7：第一条帖子的具体标签内容 5](#_Toc119)

[图 8：微博帖子内容的html标签 7](#_Toc12542)

[图 9：爬取结果示例 8](#_Toc20518)

# 数据准备

## （一）目标

获取有关“盲人运动员点燃冬残奥主火炬”事件的相关微博帖子及留言相关信息。

## （二）数据对象及其选择原因

数据对象为“盲人点火”、“盲人火炬手”、“盲人点燃冬残奥主火炬”、“盲人运动员点燃主火炬”这四个关键词的微博搜索结果，包括发帖用户名、发帖内容、留言用户名、留言时间、留言内容五个维度的数据。

因为本次数据分析以“盲人运动员点燃冬残奥主火炬”事件在微博平台上的公开讨论为研究内容，进行主题模型分析、情感分析等自然语言处理，归纳本次舆论观点和情感特征。同时，分析也需要相当庞大且可靠的数据，所以本次数据收集选择讨论热度较高的上述四个主题词进行搜索，并选择与时间、内容、用户关系相关的特征数据进行爬取。

## （三）数据收集思路

1.以“盲人点火”、“盲人火炬手”、“盲人点燃冬残奥主火炬”、“盲人运动员点燃主火炬”为搜索关键词，进行相关微博帖子搜寻；

2.分析微博网页结构以及数据传输机制，根据分析结果制定爬虫计划，编写python程序进行数据爬取

3.收集时，剔除发帖时间不符合要求的帖子信息，并对不同关键词下的搜索结果重复数据进行去重，而且对不能解析的表情进行转换，如表情不能转换则予以删除。

## （四）微博帖子结构及爬取分析

### 1.网页端微博操作流程分析

#### ①打开微博首页并登陆——搜索框输入搜索关键词，进行搜索——得到相关发帖信息

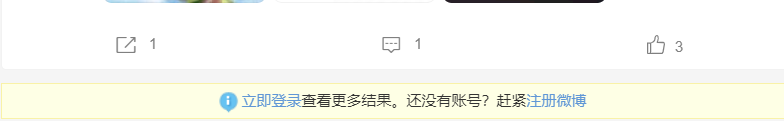


图 1：必须登录获取完整内容



图 2：新建搜索结果浏览器窗口



图 3：无法直接呈现留言具体信息

在第一流程中，我们发现三个关键点。第一，如果想要获取微博的全部发帖内容而非节选，必须得在搜索之前登录个人账号；第二，在输入关键词搜索后，会打开一个新的浏览器界面呈现搜索内容；第三，直接呈现的搜索内容可以在html代码中可以直接获取到的信息有发帖用户名、发帖用户id、发帖时间、发帖内容、转发留言点赞量，不能直接获取帖子具体的网页url以及留言用户的具体信息。

所以针对以上发现，在设计爬虫时首先考虑爬取相关主题帖子的代码里发帖用户名、发帖内容。这一部分可以通过python爬虫的xpath方法实现，这个方法避免了搜索之前需要登陆个人账号的问题；同时爬虫代码里的网页url必须是搜索结果界面的网址url，因为在使用微博首页url的情况下，xpath方法无法转换到新的浏览器界面爬取信息。

最后，为了获取留言相关具体内容，有两种可行的方法：第一种是把xpath方法换成selenium方法，在打开的网页上点击留言图标，然后在打开的留言内容上进行爬取；第二种仍旧使用xpath方法，通过分析网页上不同帖子的html代码内容，发现帖子具体网址的拼写规律，将每个帖子的具体url拼写出来，然后再打开这些网址进行爬取。因为第一个方法中，改变爬虫爬取方式会增大代码两，而且就算通过网页点击留言图标打开留言区，也无法直接获得所有的留言信息，如果留言超过一定数量就会出现“后面还有xx条评论，点击查看”按钮，还是得再次跳转到新网页，所以接下来决定采用第二种方法进行帖子具体url的收集。

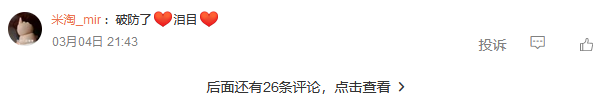


图 4：必须点击跳转具体帖子网页

#### ②分析帖子具体网址拼接规律

首先选择前两个不同的帖子，点开具体的帖子界面，将URL复制到word文档中，分析规律。发现每个帖子具体url中前面部分“https://weibo.com/1726918143/”是相同的，变化的只有后面的字母部分。所以接下来的任务是寻找前面的数字部分和后面字母部分的含义以及组织规律。



图 5：前两条帖子的具体网址

根据上述任务，对照帖子具体的网址URL，分析每条帖子的div标签里的内容。

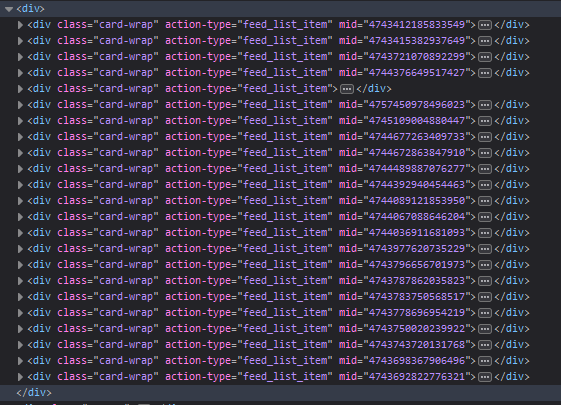


图 6：所有帖子的大div标签

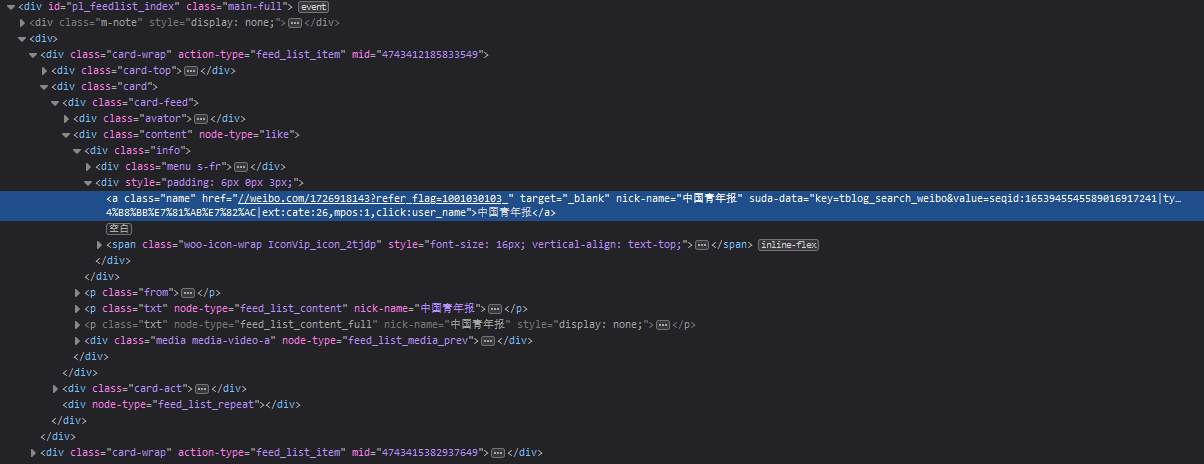


图 7：第一条帖子的具体标签内容

我们发现，在帖子用户名的标签里，有一个href属性包括帖子具体网址url前面部分，我们推测这一部分是帖子的用户页url，那么里面的数字就是用户id。接下来进行验证，将这一部分进行搜索，果真呈现微博用户浏览器页面。

最后探索网页具体网址中后面的字母部分的含义以及来源。在每条帖子的具体标签内容中，我们没有找到相应的字母信息，但是我们推测字母部分一定是可以唯一标识这条帖子的信息，所以就在具体标签内容中寻找可能唯一标识帖子的信息。我们在每条帖子的大div标签中找到mid属性，这串数字可能唯一标识帖子。接下来进行验证，将具体帖子网址url中后面的字母部分替换成相应的mid数字，进行搜索，最后结果果然是对应的具体帖子网页界面，说明推测成功。

这时，我们可以通过爬虫的xpath方法，直接在搜索结果浏览界面的html代码中爬取每条帖子用户名标签的href属性、大div标签mid属性，然后将结果按照网页规律拼凑成完整的url信息，并储存在csv文件中。拿到url后，我们就可以通过python设置循环代码，读取csv文件中的url，直接进入具体帖子界面进行评论爬取。

#### ③打开具体帖子网页，进行网页结构以及信息获取机制分析

为了考虑到尽可能多的情况，本次选择留言数较多（1314条）的帖子界面进行观察分析。结果发现，发帖用户名、发帖内容、留言用户名、留言时间、留言内容五个维度的数据均可以在帖子具体界面的html内容中一次性爬取到。

但值得注意的是，因为留言信息众多，网页并没有一次性获取全部留言信息的div标签，而是设置了网页右侧滚动条下滑到网页底部更新加载更多留言的信息获取方式。由于微博该机制的设置，这里可以采取两种方法获取全部的留言信息：第一个方法是点击浏览器该网页的检查窗口，进入网络界面进行网页抓包分析，这种滚动条下拉刷新的机制本质上是翻页，如果找到网页信息包里有关留言的规律，就可以直接通过xpath方法爬去相应的留言数据；第二种方法则要采用python的selenium包，直接操控网页右侧滚动栏下滑，下滑后爬取，再下滑再爬取，最终获得所有留言。

这里因为时间紧迫，抓包分析没能获得很好的结果，所以我进行爬虫的selenium方法尝试，最后成功获取到了留言数据。我这里采取的是selenium操控谷歌浏览器的方式进行爬取，但是因为微博登陆机制极为复杂，验证码图片识别难以实施，所以我直接设置了延时机制，再打开网页后直接人工扫码登陆，虽然耗费人力但确实是最快捷的selenium爬取方式。

值得注意的是，在滚动条下拉刷新的过程中留言标签的替换会有重复，所以在爬取后要对储存的数据进行去重操作。

### 2.微博网页爬取的注意点

#### ①微博的反爬机制：

除了上述提到的获取完整信息强制登陆的机制、滚动条刷新机制，在爬取数据的过程中我们还发现了微博的两个反爬虫机制。

第一个反爬虫机制就是微博会不定时选择性呈现搜索内容。在初次登录搜索时，搜索结果呈现完整的信息，页面最上方的帖子是热门帖；但是当我们多次试运行爬虫程序后，就触发了反爬虫机制，微博搜索结果完全改变，最上方帖子变为浏览量和留言数均极低的普通帖子，这时微博帖子内容在html代码结构中的位置也完全改变。为了应对这种机制，本次数据爬取书写了两个版本的爬虫代码，在搜索内容改变时选择相应的代码进行爬取。

第二个反爬虫机制出现在获取帖子具体网址时。当多次运行程序爬取每个帖子的mid值时，会出发微博这一反爬虫机制，所有帖子的mid值变为同一个数字串，跳转到微博搜索首页。在进行多次探索之后，我们发现只能通过刷新微博，等待它重新变回原状再进行爬取才能应对这种机制。

#### ②微博帖子内容的处理

微博帖子内容的处理包括两个方面，第一个方面是换行符和空格的删除，第二个方面是表情符号的转换。

首先在观察每个具体帖子内容时，我们发现受到微博话题的跳转链接的影响，微博帖子内容在html标签中被切成几个部分呈现（如下图），如果直接爬取大div标签的内容content，则会把切割的换行符和空格也包含在内，一起爬取下来，这时我们要采取replace方法将换行符和空格删除，由此得到较规整的内容。

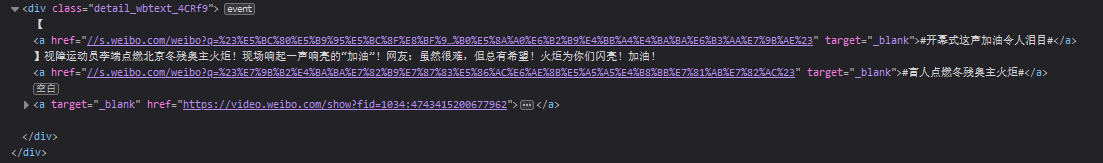


图 8：微博帖子内容的html标签

关于评论内容中表情符号的处理，我们首先观察了表情符号的大致组成，发现少部分表情符号，如冰墩墩、雪容融这种新型表情可以通过改变python代码中对内容的编码格式转换成中括号包含文字的形式呈现（如[冰墩墩]）。但是极大一部分原始的表情符号是通过图片标签<img>的形式呈现在网页里，如果不通过图片识别的方式很难转换。之前也提到过图片识别代码繁琐复杂，因技术限制以及时间紧迫，决定只转换可以转换的新型表情，放弃转换图片格式的旧表情。

最后有一个必须说明的地方，在爬去的评论中有一部分评论内容只有一个旧表情符号，所以爬取结果呈现[‘’]。考虑到这种无有效内容的评论会对分析结果产生影响，最后导出两版csv数据，一般是所有爬取结果去重以后的结果，另一版是去重以后再删除空白评论的保存结果，以供分析使用。

## 爬取结果总结说明



图 9：爬取结果示例

搜索主题词一共有四个：盲人点火、盲人火炬手、盲人点燃冬残奥主火炬、盲人运动员点燃主火炬；

首先排除主题词下发布时间在冬奥会之前的帖子；

爬取了发帖人、发帖内容、评论人、评论时间、评论内容四个维度的数据；

然后进行去重，得到“all\_re.csv”文件，共1603条数据：但是其中包含表情包单独作为内容的评论，爬取结果以“[‘’]”呈现，无意义（因为微博有大部分老一代表情包是通过图片的形式放在网页里的）；

去除单独表情评论，最终得到“all\_de\_emo.csv”文件，共1327条数据。

## 数据的不足之处

在爬取的过程中，我们发现有少部分帖子虽然带有本次搜索的主题标签，但是帖子内容与主题内容并不密切相关，由此得到的评论内容也有一部分与主题无关，这部分数据因为无法统一通过Python代码进行排除，也爬取时也一并被爬取下来，极难在爬取这一环节筛选出来去除掉。所以如果这部分数据在分析时被使用，会对分析结果造成一定程度的影响。