项目报告

- 摘要
- Abstract
- 1 项目概述
- 2 项目技术路线
  - 2.1 Python-Requests 库
  - 2.2 Python-bs4 库
  - 2.3 Python-jieba 库
  - 2.4 Python-TextRank4zh 库
    - 2.4.1 PageRank 算法
    - 2.4.2 TextRank 算法
    - 2.4.3 Python-TextRank4zh 库
- 3 项目实现
  - 3.1 爬虫策略
    - 3.1.1 分析 URL 结构
    - 3.1.2 分析 HTML 结构
    - 3.1.3 爬虫策略和程序流程图
  - 3.2 爬虫实现
    - 3.2.1 生成日期列表
    - 3.2.2 设计爬虫程序
    - 3.2.3 处理反爬虫
  - 3.3 新闻内容摘要
- 4 程序结果展示
  - 4.1 新闻爬虫结果
  - 新闻内容摘要结果

摘要

现代化社会中我们每天都要浏览大量的信息和新闻，但我们并不是的对所有的信息和新闻都感兴趣，看一篇不感兴趣新闻会浪费我们的时间和精力，因此对新闻内容进行关键词提取和摘要显得尤为重要。
自然语言处理目的在于让计算机“理解”人说的话或者文字，随着自然语言处理的快速发展使得这一目的变得可行，本项目中所采用的 TextRank 算法是一种高效率的无监督学习算法，主要功能为对文章进行关键词提取和摘要。
在本项目中，我首先设计了一个爬虫程序，爬取的目标是人民网的人民日报，可以输入开始的日期和结束的日期来爬取对应时间区间的人民日报的原稿。之后又设计了新闻摘要程序，对刚刚取的新闻原稿进行关键词提取和摘要。

Abstract

In the modern society, we browse a lot of information and news every day, but we are not interested in all the information and news. Watching a piece of uninteresting news will waste our time and energy,

so we use keywords for news content. Extraction and summarization are particularly important.
The purpose of natural language processing is to allow computers to "understand" human speech or words, and the rapid development of natural language processing has made this purpose feasible. The TextRank algorithm used in this project is an efficient unsupervised learning algorithm, and its main function is to extract keywords and summarize articles.
In this project, I first designed a crawler program. The crawling target is the People's Daily of People's Daily. You can enter the start date and end date to crawl the original manuscript of the People's Daily in the corresponding time interval. After that, a news summary program was designed to extract keywords and summarise the news manuscripts just taken.

1 项目概述

本项目由两个部分组成：
（1）爬取人民网一段时间内的所有新闻
针对人民网人民日报页面，根据用户输入的开始时间和结束时间进行爬取新闻内容。使用 Python 中的 Requests 库请求 HTTP 响应，并使用 Python 中的 bs4 库解析 HTML 页面，根据 HTML 的结构爬取新闻标题和内容，并按照新闻的日期分类保存至本地 txt 文件中。

（2）对爬取的新闻内容进行关键词提取、摘要
使用 TextRank 无监督算法快速地对爬取并保存的所有新闻进行关键词提取和摘要，并把提取的各篇新闻的标题、关键词、关键短语和摘要保存至本地 abstract.txt 文件中。

2 项目技术路线

2.1 Python-Requests 库

Requests 是用 Python 语言编写的，基于 urllib 的，采用 Apache2 Licensed 开源协议的 HTTP 库。它比 urllib 更加方便，可以节约我们大量的工作，完全满足 HTTP 测试需求。Requests 允许您非常轻松地发送 HTTP/1.1 请求。 无需手动将查询字符串添加到您的 URL，或对您的 POST 数据进行表单编码。 由于基于 urllib3，保持活动和 HTTP 连接池是 100% 自动的。

Requests 库的基本用法：
使用 Requests 以 get 方式请求 HTTP 响应：

```
r = requests.get('URL')
```

同理也可以以其他方式请求 HTTP 响应：

```
r = requests.post('URL')
r = requests.put('URL')
r = requests.head('URL')
r = requests.delete('URL')
r = requests.options('URL')
```

使用 Requests 传递参数:

```
payload = {'key1': 'value1', 'key2': 'value2'}
r = requests.get('URL', params=payload)
```

获取 HTTP 响应的内容:

```
# 自动解码来自 Web 服务器的内容
print(r.text)

# 获取二进制响应内容
print(r.content)

# 获取 JSON 响应内容
print(r.json())
```

获取 HTTP 响应的其他信息:

```
# 获取响应的状态码
print(r.status_code)

# 获取响应头
print(r.headers)

# 获取 Cookies
print(r.cookies)
```

2.2 Python-bs4 库

bs4(BeautifulSoup) 是一个 Python 实现的 HTML 或 XML 的解析器，最主要的内容就是从网页中抓取数据。BeautifulSoup 提供一些简单的 Python 式的函数用来处理导航、搜索、修改分析树等功能。它是一个工具箱，通过解析文档为用户提供需要抓取的数据，因为简单，所以不需要多少代码就可以写出一个完整的应用程序。
BeautifulSoup 自动将输入文档转换为 Unicode 编码，输出文档转换为 utf-8 编码。你不需要考虑编码方式，除非文档没有指定一个编码方式，这时，BeautifulSoup 就不能自动识别编码方式了，但是你仅仅需要说明一下原始编码方式就可以了。BeautifulSoup已成为和 lxml、html6lib 一样出色的 Python 解析器，为用户灵活地提供不同的解析策略或强劲的速度。

BeautifulSoup 的基本用法:

```
from bs4 import BeautifulSoup

# r = requests.get("URL")
soup = BeautifulSoup(r.text, "lxml")
```

```
# 通过标签选择
tag = soup.select("标签")

# 通过类名选择
tag = soup.select(".class")

# 通过 id 选择
tag = soup.select("#id")

# 例子：找到所有的 a 标签
link = soup.find_all('a')
# 获得 a 标签中的链接
href = link.get('href')
```

2.3 Python-jieba 库

> 自然语言处理目的在于让计算机“理解”人说的话或者文字，而在中文自然语言处理中第一步是获取语料，第二步就是对语料进行预处理，预处理的一个重要的环节就是对语料进行分词，其目的在于将一句话或者一个段落拆分成许多独立个体的词，这样能够方便后面将词转化成向量。

jieba 库是一个 Python 中文分词库，原理是利用一个中文词库，确定汉字之间的关联概率，汉字间概率大的组成词组，形成分词结果。除了分词，用户还可以添加自定义的词组。jieba 库提供了三种分词模式：
（1）精确模式：就是把一段文本精确地切分成若干个中文单词，若干个中文单词之间经过组合，就精确地还原为之前的文本。其中不存在冗余单词。
（2）全模式：将一段文本中所有可能的词语都扫描出来，可能有一段文本它可以切分成不同的模式，或者有不同的角度来切分变成不同的词语，在全模式下，Jieba库会将各种不同的组合都挖掘出来。分词后的信息再组合起来会有冗余，不再是原来的文本。
（3）搜索引擎模式：在精确模式基础上，对发现的那些长的词语，我们会对它再次切分，进而适合搜索引擎对短词语的索引和搜索。也有冗余。

jieba 库的基本用法：

```
# 精确模式
ls = jieba.lcut("要分词的句子")
print(ls)

# 全模式
ls = jieba.lcut("要分词的句子", cut_all = True)
print(ls)

# 搜索引擎模式
ls = jieba.lcut_for_search("要分词的句子")
print(ls)
```

```
# 向分词词典中增加新词
jieba.add_word("要增加的词")
```

2.4 Python-TextRank4zh 库

2.4.1 PageRank 算法

TextRank 算法是由**网页重要性排序算法 PageRank 算法**迁移得来，主要有关键词提取和文本摘要两个功能。
先介绍一下 PageRank 算法，PageRank 算法被应用在谷歌搜索引擎中对网页进行排名，是一种基于有向图的无监督学习的算法。该算法的排序指标：

- 链接的数量：一个网页被其他网页链接的数量，即有向图中该节点的入度
- 链接的质量：一个网页被很重要的网页链接，也表明这个网页也很重要

该算法的实现步骤：
（1）构建有向图
将 Web 上的一个网页作为一个结点，如果一个网页中链接了其他网页则构成一条指向其他结点的有向的边，根据这样的规则延伸出一个有向图。
（2）基于有向图的邻接矩阵，$S(V_i)$ 是网页 i 的 Rank 权重，d 是阻尼系数，一般为 0.85，$In(V_i)$ 是整个互联网中所存在的有指向网页 i 的链接的网页集合，$Out(V_i)$ 是网页 j 中存在的指向所有外部网页的链接集合，$|Out(V_i)|$ 是该集合中元素的个数。

$$S(V_i) = (1-\mathrm{d}) + \mathrm{d} * \sum_{j \in \mathrm{In}(V_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

（3）根据上述公式求权值，不断迭代到收敛，这是一个马尔科夫过程。

2.4.2 TextRank 算法

TextRank 算法由 PageRank 算法改进而来，将一篇文章看作一个图，将文章中的词看作一个节点，词与词之间的“共现”关系看作边，“共现”就是共同出现，即在一个给定大小的滑动窗口内的词，认为是共同出现的，这些词对应图中的结点之间有无向的边。

> 比如一句话：我周末去电影院看电影
> 分词后：我 周末 去 电影院 看 电影
> 假设滑动窗口大小为 3，则我-周末-去三个词共同出现，周末-去-电影院三个词共同出现，去-电影院-看三个词共同出现，电影院-看-电影三个词共同出现，这些共同出现的节点之间存在边。

TextRank 的排序指标：

- 如果一个词与很多词共现，那么说明这个单词比较重要
- 一个词与一个重要性很高的词共现，也能说明这个词比较重要

该算法的实现步骤：
（1）将给定的文本按照整句进行分割，即 $T=[S_1,S_2,...,S_m]$

（2）对于每个句子 $S_i \in T$，对其进行分词和词性标注，然后剔除停用词，只保留指定词性的词，如名词、动词、形容词等，即 $S_i=[t_{i,1},t_{i,2},...,t_{i,n}]$，其中为句子 i 中保留下的词

（3）构建词图 $G=(V,E)$，其中 V 为节点集合，由以上步骤生成的词组成，然后采用共现关系构造任意两个节点之间的边：两个节点之间存在边仅当它们对应的词在长度为K的窗口中共现，K 表示窗口大小，即最多共现 K 个单词，一般 K 取 2

（4）其中，$WS(V_i)$ 表示句子 i 的权重，右侧的求和表示每个相邻句子对本句子的贡献度，在单文档中，我们可以粗略的认为所有句子都是相相邻的，仅需单一文档窗口即可，W_{ji} 表示两个句子的相似度 $WS(V_j)$ 代表上次迭代出
的句子 j 的权重。d 是阻尼系数，一般为 0.85 。

$$WS(V_i)=(1-d)+d*\sum_{V_j\in \mathrm{In}(V_i)}\frac{w_{ji}}{\sum_{V_i\in O_u(V_i)}w_{jk}}WS(V_j)$$

（5）根据上面的公式，迭代计算各节点的权重，直至收敛；

在求得每个词的权重之后：

- 提取关键词：将权重降序排列选取前 20 个词语作为关键词
- 提取关键短语：在关键词中寻找相邻的词构成关键短语

而对句子的摘要则与提取关键词不同，在构建无向图时，不再以“共现关系”作为结点间是否存在边依据，而是根据句子之间的相似度构造边，如果两个句子的相似度较高，则对应图中的结点之间存在无向的边。衡量句子之间相似性的公式为：

$$\text{Similarity}\left(S_i,S_j\right)=\frac{\left|w_k\right|w_k \epsilon S_i \cap w_k \epsilon S_j\ \left|\right.}{\log\left(\left|S_i\right|\right)+\log\left(\left|S_j\right|\right)}$$

2.4.3 Python-TextRank4zh 库

TextRank4zh 库是针对中文文本的 TextRank 算法的 Python 实现。

TextRank4zh 的基本用法：

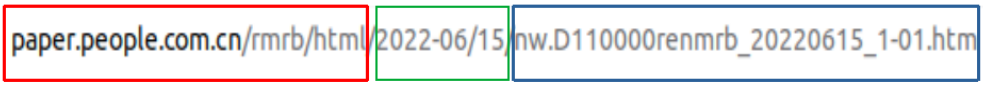
```
# 提取关键词
abstract  = TextRank4Keyword()
abstract.analyze(text=要提取的文本, lower=True or False, window=2)

# 摘要
abstract  = TextRank4Sentence()
abstract.analyze(text=要提取的文本, lower=True or False, source="all_filters")
```

3 项目实现

3.1 爬虫策略

3.1.1 分析 URL 结构

进入人民网人民日报页面，进入一篇新闻观察 URL：

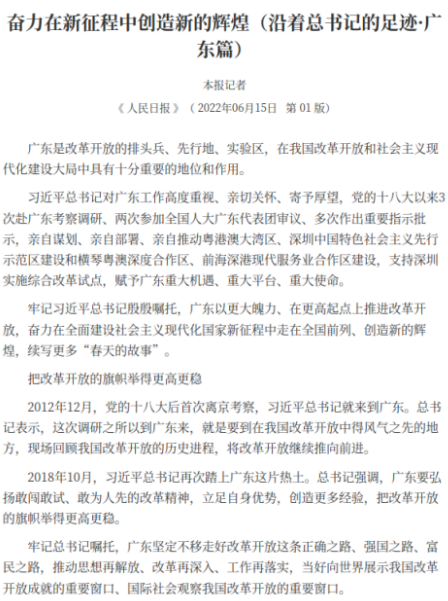


可观察出一个新闻页面的 URL 组成结构为 `人民网域名/rmrb/html/年-月/日/nw.D110000renmrb_年月日_报纸版面号-文章号.htm`

3.1.2 分析 HTML 结构

找到 pageLink 标签和 new-list 标签，分别对应报纸版面号和该报纸上的新闻列表的链接

进入一篇新闻，查看源代码可看出新闻标题所在的标签是 artichle，新闻内容所在的标签是 ozoom



奋力在新征程中创造新的辉煌（沿着总书记的足迹·广东篇）

本报记者

《人民日报》（2022年06月15日 第01版）

广东是改革开放的排头兵、先行地、实验区，在我国改革开放和社会主义现代化建设大局中具有十分重要的地位和作用。

习近平总书记对广东工作高度重视、亲切关怀、寄予厚望，党的十八大以来3次赴广东考察调研、两次参加全国人大广东代表团审议、多次作出重要指示批示，亲自谋划、亲自部署、亲自推动粤港澳大湾区、深圳中国特色社会主义先行示范区建设和横琴粤澳深度合作区、前海深港现代服务业合作区建设，支持深圳实施综合改革试点，赋予广东重大机遇、重大平台、重大使命。

牢记习近平总书记殷殷嘱托，广东以更大魄力、在更高起点上推进改革开放，奋力在全面建设社会主义现代化国家新征程中走在全国前列、创造新的辉煌，续写更多“春天的故事”。

把改革开放的旗帜举得更高更稳

2012年12月，党的十八大后首次离京考察，习近平总书记就来到广东。总书记表示，这次调研之所以到广东来，就是要到在我国改革开放中得风气之先的地方，现场回顾我国改革开放的历史进程，将改革开放继续推向前进。

2018年10月，习近平总书记再次踏上广东这片热土。总书记强调，广东要弘扬敢闯敢试、敢为人先的改革精神，立足自身优势，创造更多经验，把改革开放的旗帜举得更高更稳。

牢记总书记嘱托，广东坚定不移走好改革开放这条正确之路、强国之路、富民之路，推动思想再解放、改革再深入、工作再落实，当好向世界展示我国改革开放成就的重要窗口、国际社会观察我国改革开放的重要窗口。

3.1.3 爬虫策略和程序流程图

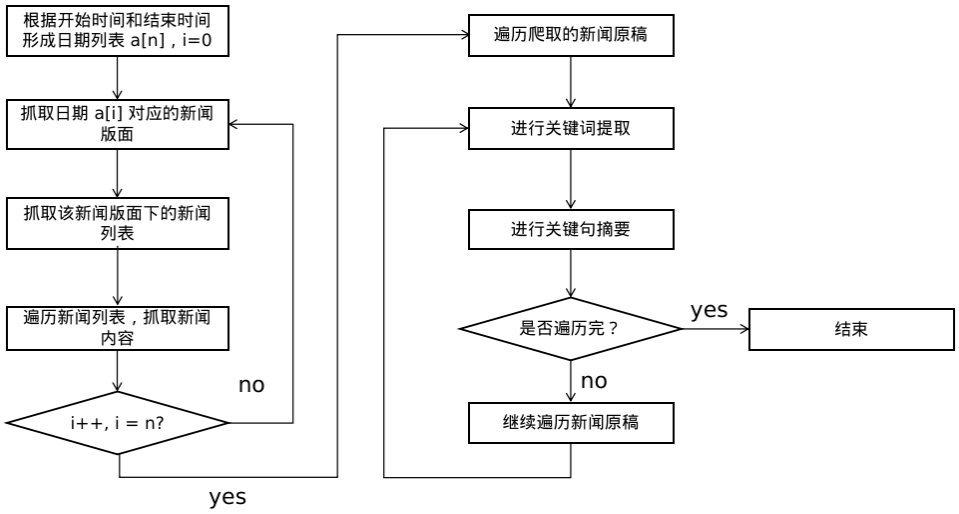
根据对 URL 和 HTML 结构的分析设计爬虫策略：

（1）先爬取版面目录，保存每一个版面的链接（pageLink）

（2）依次访问每一个版面的链接，保存该版面的文章链接（new-list）

（3）依次访问每一个文章链接，将文章的标题（article）和正文（ozoom）保存到本地 txt 文件中

程序的流程图:



3.2 爬虫实现

3.2.1 生成日期列表

定义 `gen_dates(b_date, date)` 函数和 `get_date_list(beginDate, endDate)` 函数根据开始日期和结束日期生成日期列表

```
def gen_dates(b_date, days):
    day = datetime.timedelta(days = 1)
    for i in range(days):
        yield b_date + day * i


def get_date_list(beginDate, endDate):
    """
    获取日期列表
    :param start: 开始日期
    :param end: 结束日期
    :return: 开始日期和结束日期之间的日期列表
    """

    start = datetime.datetime.strptime(beginDate, "%Y%m%d")
    end = datetime.datetime.strptime(endDate, "%Y%m%d")

    data = []
    for d in gen_dates(start, (end-start).days):
        data.append(d)

    return data
```

3.2.2 设计爬虫程序

定义 `fetchurl(url)` 函数抓取 URL，访问 URL 的网页并获取网页的内容

```
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
    return r.text
```

定义 `getPageList(year, month, day)` 函数获取人民日报某一天的各版面的链接列表

```
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
    '''
    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"]
        url = 'http://paper.people.com.cn/rmrb/html/'  + year + '-' + month +
'/' + day + '/' + link
```

```
        linkList.append(url)

    return linkList
```

定义 `getTitleList(year, month, day, pageUrl)` 获取报纸某一版面的文章链接列表

```
def getTitleList(year, month, day, pageUrl):
    '''
    功能：获取报纸某一版面的文章链接列表
    参数：年，月，日，该版面的链接
    '''
    html = fetchUrl(pageUrl)
    bsobj = bs4.BeautifulSoup(html,'html.parser')
    temp = bsobj.find('div', attrs = {'id': 'titleList'})
    if temp:
        titleList = temp.ul.find_all('li')
    else:
        titleList = bsobj.find('ul', attrs = {'class': 'news-
list'}).find_all('li')
    linkList = []

    for title in titleList:
        tempList = title.find_all('a')
        for temp in tempList:
            link = temp["href"]
            if 'nw.D110000renmrb' in link:
                url = 'http://paper.people.com.cn/rmrb/html/'  + year + '-' +
month + '/' + day + '/' + link
                linkList.append(url)

    return linkList
```

定义 `getContent(html)` 函数解析 HTML 网页，获取新闻的文章的标题和内容

```
def getContent(html):
    '''
    功能：解析 HTML 网页，获取新闻的文章内容
    参数：html 网页内容
    '''
    bsobj = bs4.BeautifulSoup(html,'html.parser')

    # 获取文章 标题
    title = bsobj.h3.text + '\n' + bsobj.h1.text + '\n' + bsobj.h2.text + '\n'
    #print(title)

    # 获取文章 内容
    pList = bsobj.find('div', attrs = {'id': 'ozoom'}).find_all('p')
```

```
    content = ''
    for p in pList:
        content += p.text + '\n'
    #print(content)

    # 返回结果 标题+内容
    resp = title + content
    return resp
```

定义 `saveFile(content, path, filename)` 函数将文章内容 content 保存到本地文件（path/filename）中

```
def saveFile(content, path, filename):
    '''
    功能：将文章内容 content 保存到本地文件中
    参数：要保存的内容，路径，文件名
    '''
    # 如果没有该文件夹，则自动生成
    if not os.path.exists(path):
        os.makedirs(path)

    # 保存文件
    with open(path + filename, 'w', encoding='utf-8') as f:
        f.write(content)
```

定义 `download_rmrb(year, month, day, destdir)` 函数以实现完整的爬取某一天人民网所有新闻并保存至本地的全过程

```
def download_rmrb(year, month, day, destdir):
    '''
    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
    参数：年，月，日，文件保存的根目录
    '''
    pageList = getPageList(year, month, day)
    for page in pageList:
        titleList = getTitleList(year, month, day, page)
        for url in titleList:

            # 获取新闻文章内容
            html = fetchUrl(url)
            content = getContent(html)

            # 生成保存的文件路径及文件名
            temp = url.split('_')[2].split('.')[0].split('-')
            pageNo = temp[1]
            titleNo = temp[0] if int(temp[0]) >= 10 else '0' + temp[0]
            path = destdir + '/' + year + month + day + '/'
            fileName = year + month + day + '-' + pageNo + '-' + titleNo +
```

```
'.txt'

            # 保存文件
            saveFile(content, path, fileName)
```

3.2.3 处理反爬虫

人民日报网设置了反爬措施，当频繁的爬取时会被拒绝新的连接。为了防止反爬虫，设定了 `time.sleep(3)` 语句，即每当爬取完一天的新闻后，系统等待 3 秒钟再继续进行爬取，成功避免了被反爬虫措施拒绝连接

```
 for d in data:
        year = str(d.year)
        month = str(d.month) if d.month >=10 else '0' + str(d.month)
        day = str(d.day) if d.day >=10 else '0' + str(d.day)
        download_rmrb(year, month, day, 'data')
        print("爬取完成: " + year + month + day)
        time.Sleep(3)
```

3.3 新闻内容摘要

定义 `analy(file)` 函数对 file 文件中的新闻内容进行关键词提取和文章摘要，并将提取的关键词、关键短语以及其对应的权重排名、摘要（关键句）保存至 abstract.txt 文件中

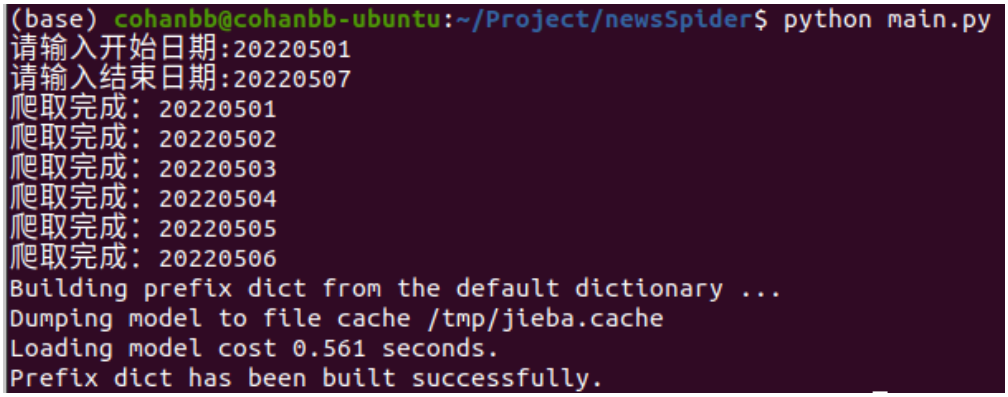
```
def analy(file):
    # 待读取的文本文件，一则新闻
    # 打开并读取文本文件
    text = codecs.open(file, 'r', 'utf-8').read()
    news_name = os.path.basename(file)

    # 创建分词类的实例
    tr4w = TextRank4Keyword()
    # 对文本进行分析，设定窗口大小为2，并将英文单词小写
    tr4w.analyze(text=text, lower=True, window=2)

    """输出"""
    file_handle = open('./abstract.txt', mode='a')
    file_handle.writelines(['新闻原稿: ', news_name, '\n'])
    file_handle.write('关键词为: \n')
    # 从关键词列表中获取前20个关键词
    for item in tr4w.get_keywords(num=20, word_min_len=1):
        # 打印每个关键词的内容及关键词的权重
        file_handle.writelines([item.word, ' ', str(item.weight), '\n'])
    file_handle.write('\n')

    file_handle.write('关键短语为: \n')
    # 从关键短语列表中获取关键短语
```

```
    for phrase in tr4w.get_keyphrases(keywords_num=20, min_occur_num=2):
        file_handle.write(phrase)
    file_handle.write('\n')

    # 创建分句类的实例
    tr4s = TextRank4Sentence()
    # 英文单词小写，进行词性过滤并剔除停用词
    tr4s.analyze(text=text, lower=True, source='all_filters')

    file_handle.write('\n摘要为：\n')
    # 抽取3条句子作为摘要
    for item in tr4s.get_key_sentences(num=3):
        # 打印句子的索引、权重和内容
        file_handle.writelines([str(item.index), ' ', str(item.weight), ' ',
item.sentence, '\n'])
    file_handle.write('\n')
```

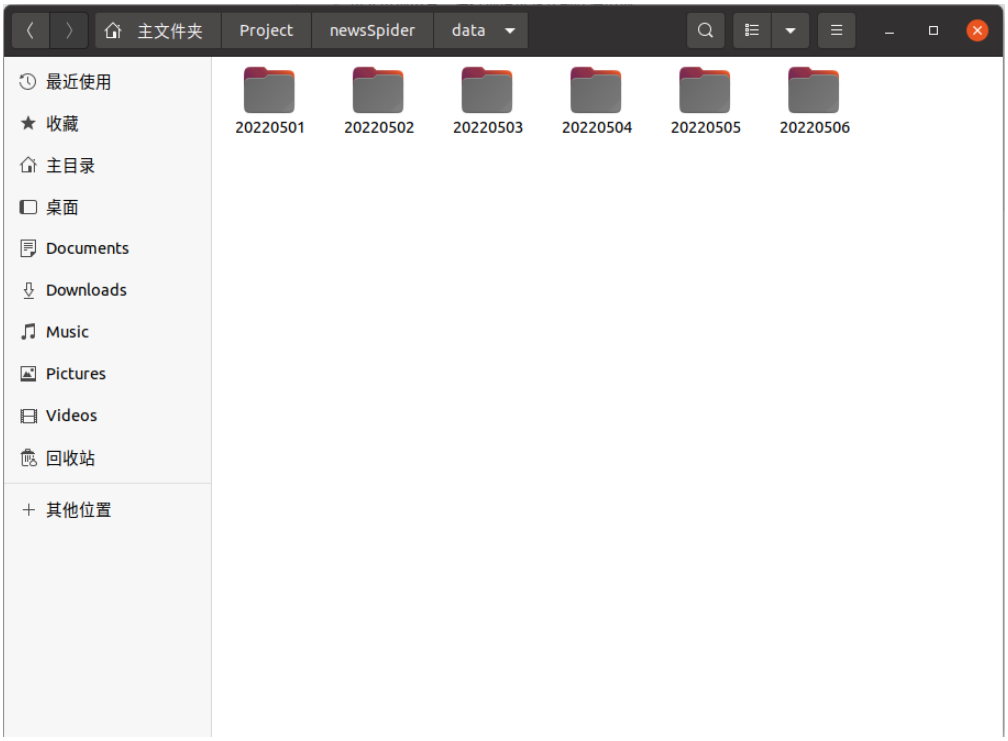
4 程序结果展示

4.1 新闻爬虫结果

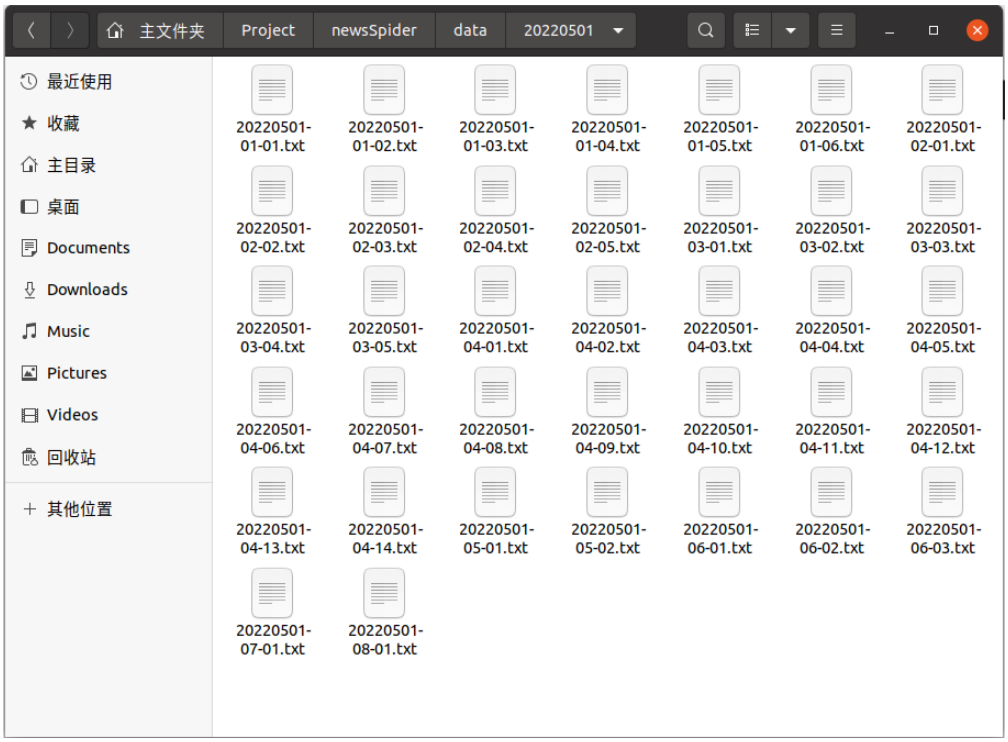
输入开始日期和结束日期后开始爬虫：
输入开始日期为 2022 年 5 月 1 日，输入结束日期为 2022 年 5 月 7 日。



爬虫结果：
爬取的新闻以日期为目录存放在本地：



考虑到篇幅，此处仅展示所爬取的 2022 年 5 月 1 日的新闻内容，其他日期的效果与之相同：



新闻内容摘要结果

摘要的内容被存放在 abstracts.txt 文件中：

此处给出一个具体的新闻例子和其摘要后的效果进行展示:

新闻原稿:

> 习近平对湖南长沙居民自建房倒塌事故作出重要指示
> 要求不惜代价搜救被困人员 全力救治受伤人员对全国自建房安全开展专项整治 坚决防范各类重大事故发生
> 李克强作出批示
>
> 新华社北京4月30日电 4月29日12时24分，湖南长沙市望城区金山桥街道金坪社区一居民自建房发生倒塌事故。截至目前，仍有数十人被困。
>
> 事故发生后，中共中央总书记、国家主席、中央军委主席习近平立即作出重要指示，要不惜代价搜救被困人员，全力救治受伤人员，妥善做好安抚安置等善后工作；同时注意科学施救，防止发生次生灾害。要彻查事故原因，依法严肃追究责任，从严处理相关责任人，及时发布权威信息。近年来多次发生自建房倒塌事故，造成重大人员伤亡，务必引起高度重视。要对全国自建房安全开展专项整治，彻查隐患，及时解决。坚决防范各类重大事故发生，切实保障人民群众生命财产安全和社会大局稳定。
>
> 中共中央政治局常委、国务院总理李克强作出批示，要求抢抓黄金救援时间全力、科学搜救被困人员，做好伤员救治，尽最大努力减少因伤致残和死亡。同时妥善做好家属安抚等善后工作，实事求是、公开透明发布信息。要认真查明事故原因，依法依规严肃问责。要督促各地深入排查整治建筑行业尤其是经营性自建房安全隐患，严防重特大事故发生。
>
> 根据习近平指示和李克强要求，国务委员王勇和应急管理部、住房和城乡建设部、教育部、国家卫生健康委等部门负责同志赶赴现场指导事故救援和应急处置工作。湖南省、

> 长沙市党政负责同志已在现场指挥处置。目前，现场救援、伤员救治和事故原因调查等工作正在紧张有序进行中。

关键词提取和摘要后的效果：