

Deep News Nankai

李兴贺 1611071

2018/12/14

目录

1	Deep News 搜索引擎简介	1
2	系统环境	2
3	Deep News 使用说明	3
4	数据抓取子系统	7
5	内容索引子系统	10
6	链接结构分析子系统	13
7	内容检索子系统	14
8	总结	16

1 Deep News 搜索引擎简介

这学期选了温老师的信息检索系统原理，第三次作业是要做一个 Deep Web Search Engine。说实话，直到完成这次的代码编写和测试工作，我仍然对这个 Deep 的真正含义不是很清楚。

虽然不是很清楚，但是我不能就这么坐以待毙呀，想起温老师曾经说过，这次作业和普通的搜索引擎相比，就是爬虫的部分写法不太一样，我深以为然，开始一步一步动手做出这个项目。

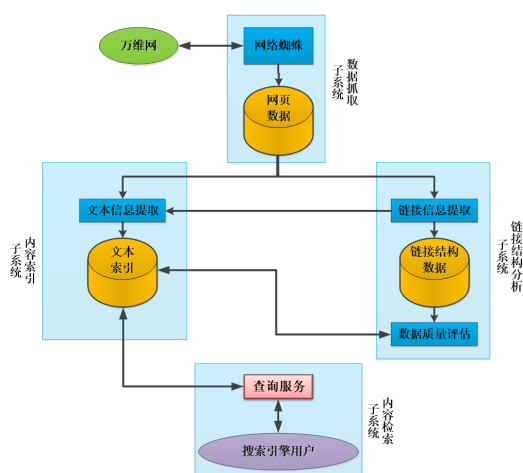


图 1: 搜索引擎体系结构

通常，搜索引擎体系结构由四个部分构成，分别是数据抓取子系统、内容索引子系统、链接结构分析子系统、内容检索子系统。Deep News 也不例外，下面我将演示搜索引擎的使用，并分别介绍这几个子系统的设计、实现和调用。

2 系统环境

系统:Windows10

编程语言: Python3.6

Python 模块: math operator datetime configparser time os xml

sqlite3(数据库)

jieba(中文分词)

flask(web 框架)

requests(HTTP 库)

newspaper(新闻处理)

pyinstaller(py 打包 exe)

本项目已将内容检索子系统打包成 exe 文件，可以在没有 Python 环境的系统下直接运行。其他子系统均需要以上 Python 模块，可以运行../deep-news/env.py 一键安装环境。

本项目中唯一的配置文件是../deep-news/config.ini，配置文件的解析由 configparser 模块完成。

3 Deep News 使用说明

运行../deep-news/main.exe(其实是快捷方式,本体是../deep-news/web/main.exe)

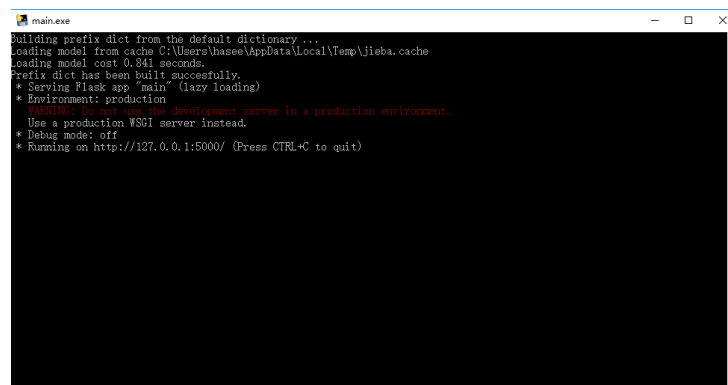


图 2: main.exe 后台运行截图

按照命令行提示, 接下来打开浏览器, 输入 127.0.0.1:5000 进入搜索引擎主界面。

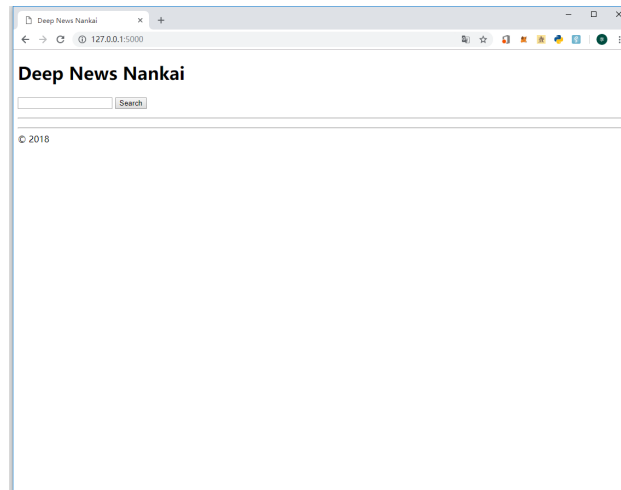


图 3: Deep News 初始界面

在搜索栏输入想要搜索的内容，点击”Search” 进行检索。Deep-News 提供三种检索方式，可以在搜索栏下方点击” 相关度”、” 时间” 或” 热度”，再点击”Ok”，修改检索方式。



图 4: 检索页面示例 1

页面底部可以自由跳页查看全部检索结果。



图 5: 检索页面示例 2



图 6: 检索结果示例

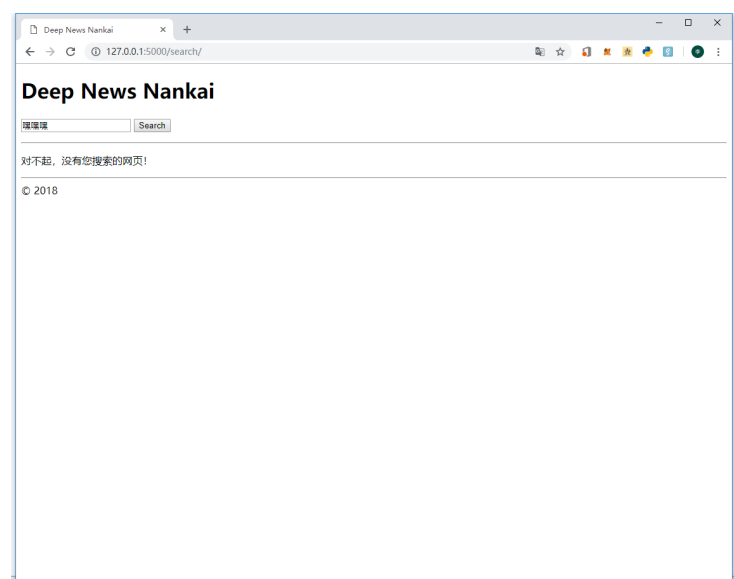


图 7: 检索失败示例

4 数据抓取子系统

源码../deep-news/code/spider.py

抓取目标：南开大学新闻网 (<http://news.nankai.edu.cn/>)

抓取页面数量:11143

这里选择南开大学新闻网的原因是，网页的 html 代码相对比较规整（其实是用现有的工具可以解析），并且爬虫运行过程中并不需要登录也不用面对验证码。数据抓取子系统的思路是遍历整个网站，将新闻内容的链接加入 newspool，然后访问 newspool 中的每个链接向其发送 get 请求并对返回的页面进行解析，将得到的有效信息保存在../data/news/xxx.xml 文件中。（事实上，得到了一万多个 xml 文件后，我们甚至可以不做一个搜索引擎，转而做一个“南开大学旧闻网”。显然，这并没有什么意义）。newspool 中所有的链接都爬取过后，数据抓取子系统的工作就结束了。

```
def get_news_pool():
    news_pool = []
    for i in range(200,243):
        temp=442-i
        target =
            'http://news.nankai.edu.cn/nkyw/system/more/32000000/0002/32000000_0000
        req = requests.get(url = target)
        html = req.text
        bf = BeautifulSoup(html)
        texts = bf.find_all('a', class_ = 'news')
        for i in range(len(texts)):
            url=str(texts[i])
            start=url.find('http')
            end=url.find('shtml')+5
            url=url[start:end]
            news_pool.append(url)
    time.sleep(t)
```

```
return(news_pool)
```

这是 get-news-pool 函数的部分代码,工作流程为循环遍历 <http://news.nankai.edu.cn/nky> 的每一个页面,向其发送 get 请求,将返回的 html 页面用 BeautifulSoup 处理后得到每条新闻的 url,将 url 加入 news-pool,工作完成后函数返回的是列表 news-pool

```
def
    crawl_news(news_pool,min_body_len,doc_dir_path,doc_encoding):
    i=1
    for news in news_pool:
        article=Article(news,language='zh')
        try:
            article.download()
            article.parse()
        except Exception as e:
            print("-----%s: %s-----"%(type(e), news))
            continue
        doc=ET.Element("doc")
        ET.SubElement(doc, "id").text = "%d"%(i)
        ET.SubElement(doc, "url").text = news
        ET.SubElement(doc, "title").text = article.title
        ET.SubElement(doc, "datetime").text =
            str(article.publish_date)
        ET.SubElement(doc, "body").text = article.text
        tree = ET.ElementTree(doc)
        tree.write(doc_dir_path + "%d.xml"%(i), encoding =
            doc_encoding, xml_declaration = True)
    i += 1
```

crawl-news 主要使用的是 newspaper 模块，提取出 news-pool 中每个 url 指向的新闻链接的标题、正文、时间，并连同 url 一起存进 xml 文件中。其中，Article 是 newspaper 模块用来存储文章的对象，article.download() 和 article.parse() 分别对该 article 对象进行下载和解析。实际爬取过程中可能出现异常，我们将异常情况输出至控制台并跳过该 url 继续执行下一条。

5 内容索引子系统

源码../deep-news/code/index.py

内容索引子系统是搜索引擎中至关重要的部分。index.py 主要实现了一个 index module 类，构建索引的所有操作都是该类的成员函数。

```
class IndexModule:
    stop_words = set()
    postings_lists = {}

    config_path = ''
    config_encoding = ''
```

IndexModule 类的成员变量。分别是停用词、记录列表、配置文件路径、编码。

```
def construct_postings_lists(self):
    config = configparser.ConfigParser()
    config.read(self.config_path, self.config_encoding)
    files = listdir(config['DEFAULT']['doc_dir_path'])
    AVG_L = 0
    for i in files:
        try:
            root = ET.parse(config['DEFAULT']['doc_dir_path'] +
                             i).getroot()
            title = root.find('title').text
            body = root.find('body').text
            docid = int(root.find('id').text)
            date_time = root.find('datetime').text
            seg_list = jieba.lcut(title + '。' + body,
                                   cut_all=False)
```

```

ld, cleaned_dict = self.clean_list(seg_list)

AVG_L = AVG_L + ld

for key, value in cleaned_dict.items():
    d = Doc(docid, date_time, value, ld)
    if key in self.postings_lists:
        self.postings_lists[key][0] =
            self.postings_lists[key][0] + 1 # df++
        self.postings_lists[key][1].append(d)
    else:
        self.postings_lists[key] = [1, [d]] # [df, [Doc]]
except Exception as e:
    print("-----%s: %s-----"%(type(e), i))
    continue
AVG_L = AVG_L / len(files)
config.set('DEFAULT', 'N', str(len(files)))
config.set('DEFAULT', 'avg_l', str(AVG_L))
with open(self.config_path, 'w', encoding =
    self.config_encoding) as configfile:
    config.write(configfile)
self.write_postings_to_db(config['DEFAULT']['db_path'])

```

倒排索引构建算法使用内存式单遍扫描索引构建方法（SPIMI），其实就是依次对每篇新闻进行分词，如果出现新的词项则插入到词典中，否则将该文档的信息追加到词项对应的倒排记录表中。

使用了 jieba 中文分词系统，效果还是很不错的，不用太操心。停用词列表是从别的搜索引擎项目中直接拿来的，应用中没有发现什么问题。

倒排索引构建完成后将其写入数据库中。

```

def write_postings_to_db(self, db_path):

```

```

conn = sqlite3.connect(db_path)
c = conn.cursor()

c.execute('''DROP TABLE IF EXISTS postings''')
c.execute('''CREATE TABLE postings
(term TEXT PRIMARY KEY, df INTEGER, docs TEXT)''')

for key, value in self.postings_lists.items():
    doc_list = '\n'.join(map(str,value[1]))
    t = (key, value[0], doc_list)
c.execute("INSERT INTO postings VALUES (?, ?, ?)", t)

conn.commit()
conn.close()

```

数据库采用的是 sqlite3，写入的记录格式下图所示。

term	df	docs
8862 12018-12-10 0000002273102018-12-07 000000128681002018-11-21 000000117710002018-03-20 00000020862100012011-07-01 0000000		
南开大学	120	12018-12-10 000000127310082011-06-16 0000001101100832011-06-15 000000227210262018-03-08 000000228100812011-03-01 00000
天津	2844	12018-12-10 00000002731002018-03-20 00000030402100012011-07-01 0000001181100012011-07-01 00000020280100042011-07-01 000
福建师范大学	892	12018-12-10 0000004273100902011-06-30 000000129100152011-06-29 0000001328100512011-06-23 0000001202100762011-06-17 000
河南	3317	12018-12-10 0000001273102018-12-07 000000186910002018-03-20 00000020862100012011-07-01 000000172100012011-07-01 00000
浙江	747	12018-12-10 0000001327310002018-03-16 0000001481012018-11-18 000000104010062011-06-04 00000044610122011-05-25 00000
湖南	4369	12018-12-10 000000172310002018-03-20 00000005862100012011-07-01 0000001017210002011-07-01 0000001381100012011-07-01 00000
江苏	2061	12018-12-10 0000004273102018-12-07 0000001869100012011-07-01 0000000840100042011-07-01 0000003406100052011-06-30 00000
特色	2337	12018-12-10 00000002731002018-03-20 0000003366210002011-07-01 00000011381100012011-07-01 0000000940100042011-07-01 000
四川	259	12018-12-10 0000004273100382011-06-25 0000001219104202011-05-23 000000130710082011-05-04 0000001184102032011-04-26 000
福建	1307	12018-12-10 0000005273102018-12-07 0000001869100012011-07-01 0000004128100052011-06-30 0000003348100062011-06-30 00000
湖北	3513	12018-12-10 00000002731002018-03-20 0000001869100012011-07-01 000000172100012011-07-01 0000001840100042011-07-01 000
河南	309	12018-12-10 000000227310002011-07-01 0000001281100062011-06-30 0000003177100102011-06-30 000000323100202011-06-27 000
南开	10850	12018-12-10 0000002273102018-12-07 0000003669100018-11-21 00000017710002018-03-20 0000004862100012011-07-01 00000031
福建	10566	12018-12-10 0000001273102018-12-07 0000001869100018-11-21 00000017710002018-03-20 0000001862100012011-07-01 00000011
新疆兵团	9096	12018-12-10 0000001273102018-12-07 0000001869100018-11-21 00000017710002018-03-20 0000001862100012011-07-01 00000011
湖北	8788	12018-12-10 0000001273102018-12-07 0000001869100018-11-21 00000017710002018-03-20 0000001862100012011-07-01 00000011
湖南	8502	12018-12-10 0000001273102018-11-21 0000001177100012011-07-01 00000017210002011-06-30 0000001281100102011-06-30 00000
南开大学	1	12018-12-10 0000001273
王学博	54	12018-12-10 000000127310992018-02-05 000000131613532017-11-20 000000146810002017-09-24 0000001325102017-09-24 0000000
江苏	601	12018-12-10 00000012731002018-03-20 000000128101242011-07-01 00000013801042011-05-21 000000118210152011-05-18 00000
中国	4794	12018-12-10 0000005273102018-12-07 0000001286810002018-03-20 0000002086210002011-07-01 00000051281100012011-07-01 00000
福建师范大学	847	12018-12-10 000000627310002011-07-01 0000001281100032011-07-01 0000001940100152011-06-29 0000001208100532011-06-23 000
李金	698	12018-12-10 0000000273102018-12-07 000000176910002011-07-01 0000001138110032011-06-20 000000154710042011-06-24 00000
分学	117	12018-12-10 000000327310112011-06-02 0000003407014202011-05-24 0000001112102202011-04-17 000000162510242011-04-14 000
工作	6178	12018-12-10 0000005273102018-12-07 0000003669100018-03-20 0000003862100012011-07-01 00000017210002011-07-01 0000000
平谷	212	12018-12-10 0000001273102018-12-07 000000086910012011-06-27 0000001241100012011-06-20 000000176210142011-05-21 00000
董	915	12018-12-10 0000001273102018-12-07 0000001869100192011-06-29 0000001284100262011-06-27 0000001480100272011-06-27 00000
第九组	137	12018-12-10 000000127310442011-05-23 0000003448102502011-04-12 000000158210382011-02-28 0000001364109932010-10-22 000

图 8: 倒排记录表可视化

6 链接结构分析子系统

进行到这一步的时候，我发现了严重的问题。南开大学新闻网并没有推荐系统。

以 <http://news.nankai.edu.cn/nkyw/system/2018/12/15/000423674.shtml> 这篇新闻为例，网页中不包含其他新闻的链接，导致 PageRank 无从做起。但是我曾经选过大数据计算及应用的课程，课程的大作业是实现 PageRank 算法..... 所以我只能将那次大作业一并上传，以示友好。PageRank 在../deep-news/source/Pagerank 目录下。

7 内容检索子系统

源码../deep-news/web/search engine.py

Deep News 支持三种检索类型，分别是相关度检索、时间倒序检索、热度检索。其中相关度检索采用经典的 tf-idf 算法，时间倒序是按新闻发布时间倒序排列，热度采用的是时间/相关度混合算法。

```
def result_by_tfidf(self, sentence):
    seg_list = jieba.lcut(sentence, cut_all=False)
    n, cleaned_dict = self.clean_list(seg_list)
    TFIDF_scores = {}
    for term in cleaned_dict.keys():
        r = self.fetch_from_db(term)
        if r is None:
            continue
        df = r[1]
        docs = r[2].split('\n')
        for doc in docs:
            docid, date_time, tf, ld = doc.split('\t')
            docid = int(docid)
            tf = int(tf)
            ld = int(ld)
            tf=tf/ld
            idf=math.log2(self.N/df)
            s = tf*idf
            if docid in TFIDF_scores:
                TFIDF_scores[docid] = TFIDF_scores[docid] + s
            else:
                TFIDF_scores[docid] = s
    TFIDF_scores = sorted(TFIDF_scores.items(), key =
        operator.itemgetter(1))
```



```
TFIDF_scores.reverse()
if len(TFIDF_scores) == 0:
    return 0, []
else:
    return 1, TFIDF_scores
```

tf-idf 公式

$$idf = \log_2\left(\frac{N}{df}\right)$$

$$tfidf = tf_1 * idf_1 + tf_2 * idf_2 + \dots + tf_n * idf_n$$

时间、热度检索代码与 tf-idf 大同小异，详见源码。

热度公式

$$hot_{score} = k1 * \log_2(tfidf) + \frac{k2}{time_{now} - time_{news}}$$

其中 k1、k2 为可变参数，初始设为 1（其实根本没调参，1 的效果还算勉强强）

8 总结

以上就是本次项目的主要内容。在一步一步完成的过程中，我收获了很多经验和技巧，因为前两次作业实现的特别水，导致这次完全是从零开始，爬虫、构建索引、检索模型，我对每一部分都有了深入的了解，并清楚它们之间的联系。美中不足的是本项目没有成功的应用 PageRank 算法，flask 前端部分也并非全部独立完成而是用别人的模板修改而来。latex 真的是个优秀的排版系统，这是我与 latex 的首次接触，但应该也是最后一次了，因为最终还是要投入到 markdown 的怀抱。

最后，感谢您的耐心阅读。