

信息检索第三次作业

孙靖淇

2018 年 12 月 16 日

目录	2
----	---

目录

第一部分 环境配置	3
1 scrapy安装配置	3
2 elasticSearch安装配置	4
2.1 logstash-input-jdbc 6.5.2的安装配置	5
2.1.1 logstash 上传数据并同步	5
2.2 elasticsearch-head-master的安装配置	6
2.3 analyzer-ik的安装配置	7
3 springboot整合mybatis+elasticsearch+jdbc+mysql+web	7
4 前端配置	10
 第二部分 项目需求及实现	 11
5 数据爬取	11
5.1 数据说明	11
5.2 scrapy爬取	11
5.2.1 Ip.py	12
5.2.2 settings.py	12
5.2.3 pipelines.py	13
5.2.4 items.py	14
5.2.5 Middlewares.py	15
6 链接分析pageRank	16
7 查询服务	17
7.1 高亮展示	17
7.2 通配查询	17
7.3 精确查询: site	19
7.4 分类查询	20
7.5 语音输入	21

目录	3
7.6 语音合成	22
7.7 网页快照	22
7.8 分页处理	23

摘要

本次作业要实现搜索引擎。搜索引擎包括四个子系统，即：数据爬取子系统、内容索引子系统、链接分析子系统、内容检索子系统。本次作业中，数据爬取自南开大学各学院网站，搜索主题为新闻，提供短语查询，通配查询，网页快照，语音识别，语音合成，制定site等查询服务。接下来将对上述的各部分进行详细说明。

第一部分 环境配置

1 scrapy安装配置

Scrapy是纯python开发实现的一个爬虫框架，包含爬取数据、提取结构性数据、应用框架。底层通过Twisted异步网络框架处理网络通讯，是一个可扩展、高性能、多线程、分布式爬虫框架。

scrapy是基于Python的框架，所以首先需要安装Python，此处使用的Python版本为3.7。具体配置步骤如下：

1. 安装python,下载版本3.7，安装时需要勾选加入系统环境变量选项。
2. 安装lxml lxml是lxml是一种使用Python 编写的库，可以迅速、灵活地处理XML。命令：`python -m pip install lxml`
3. 安装setuptools 一般都已经安装了的，可在cmd中用`python -m pip list`查看是否已经安装，如果没安装，用下载命令：`python -m pip install setuptools`
4. 与上面的方法相同，依次安装zope.interface, Twisted, pyOpenSSL, win32py, 然后安装scrapy, 如果顺序不一致会导致scrapy无法正常启动。
5. 安装scrapy, 可以在控制台输入scrapy查看是否安装成功,命令：`easy_install scrapy`

```
C:\Users\Administrator>scrapy
Scrapy 1.5.0 - no active project

Usage:
  scrapy <command> [options] [args]

Available commands:
  bench      Run quick benchmark test
  fetch      Fetch a URL using the Scrapy downloader
  genspider  Generate new spider using pre-defined templates
  runspider  Run a self-contained spider (without creating a project)
  settings   Get settings values
  shell      Interactive scraping console
  startproject Create new project
  version    Print Scrapy version
  view       Open URL in browser, as seen by Scrapy

[ more ]    More commands available when run from project directory

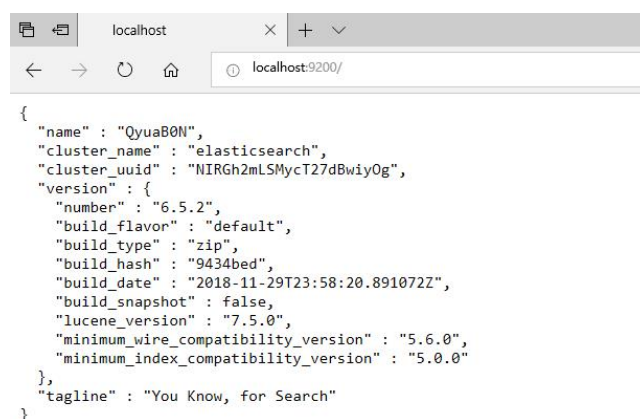
Use "scrapy <command> -h" to see more info about a command
```

图 1: 运行scrapy

2 elasticSearch安装配置

Elasticsearch是一个基于Apache Lucene(TM)的开源搜索引擎，是一个高可用开源全文检索和分析组件。提供存储服务，搜索服务，大数据准实时分析等。一般用于提供一些提供复杂搜索的应用。本项目使用的版本为6.5.2. 配置步骤：

1. elasticsearch官网下载压缩包,解压到指定目录
2. 运行./bin/elasticsearch.bat
3. 浏览器中输入http://localhost:9200 验证，如图为成功



```
{
  "name" : "QyuaB0N",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "NIRGh2mLSMycT27dBwiy0g",
  "version" : {
    "number" : "6.5.2",
    "build_flavor" : "default",
    "build_type" : "zip",
    "build_hash" : "9434bed",
    "build_date" : "2018-11-29T23:58:20.891072Z",
    "build_snapshot" : false,
    "lucene_version" : "7.5.0",
    "minimum_wire_compatibility_version" : "5.6.0",
    "minimum_index_compatibility_version" : "5.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

图 2: localhost:9200

为了查看节点的信息，索引状态信息，上传数据等操作，我们需要安装插件elasticsearchheadmaster和logstash。另外，elasticsearch默认的中文分词只能分成单个汉字，所以需要下载安装分词器analyzer-ik。

选择插件版本时注意要与elasticsearch的版本保持一致。

2.1 logstash-input-jdbc 6.5.2的安装配置

在官网可以下载插件。需要对其进行一些配置才能正常使用。步骤如下：

1. 使用mysql数据库，新建一张表news。
2. 在根目录下新建文件夹mysqletc，下载jar包mysql-connector-java，并放在该文件夹下
3. 新建文件mysql.conf， test.sql， mysql.conf是上传数据的配置信息， test.sql是要执行的sql语句。

2.1.1 logstash 上传数据并同步

mysql.conf的配置如图3所示。

```

1 input {
2   stdin {
3     }
4   }
5   jdbc {
6     # mysql 数据库链接,shop为数据库名
7     jdbc_connection_string => "jdbc:mysql://127.0.0.1:3306/database"
8     # 用户名和密码
9     jdbc_user => "root"
10    jdbc_password => "root"
11    # 驱动
12    jdbc_driver_library => "E:/logstash-6.5.2/mysqletc/mysql-connector-java-5.1.42-bin.jar"
13    # 驱动类名
14    jdbc_driver_class => "com.mysql.jdbc.Driver"
15    jdbc_paging_enabled => "true"
16    jdbc_page_size => "50000"
17    # 执行的sql 文件路径+名称
18    statement_filepath => "E:/logstash-6.5.2/mysqletc/test.sql"
19    # 设置监听间隔 各字段含义 (由左至右) 分、时、天、月、年，全部为*默认含义为每分钟都更新
20    schedule => "* * * * *"
21    # 索引类型
22    type => "news_ik_t"
23  }
24 }
25 filter {
26   json {
27     source => "message"
28     remove_field => ["message"]
29   }
30 }
31 output {
32   elasticsearch {
33     hosts => ["localhost:9200"]
34     index => "news_ik_t"
35     document_id => "%{id}"
36   }
37   stdout {
38     codec => json_lines
39   }
40 }

```

图 3: mysql.conf

test.sql中直接将对应表的所有信息查找出来即可，即：select * from news.ik。

打开elasticsearch后, 在logstash的bin目录下执行`./logstash -f ../mysql/etc/mysql.conf`即可。

2.2 elasticsearch-head-master的安装配置

本插件主要用于查看节点信息, 索引状态等信息, 配置步骤如下:

1. github上下载最新的分支, 注意版本问题, 地址: <https://github.com/mobz/elasticsearch-head>
2. 安装node,npm,grunt
3. 修改Elasticsearch 配置文件, `./elasticsearch-6.5.2/config/elasticsearch.yml`, 添加如下语句:

```
http.cors.enabled:true  
http.cors.allow-origin: '*'
```
4. 修改Gruntfile.js, 将hostname改为127.0.0.1, 端口号设置为9100.
5. 执行`npm install`, 然后安装grunt, 命令: `npm install -g grunt`
6. 启动head, 命令: `npm run start`
7. 浏览器访问localhost:9100, 配置成功, 如图4:



图 4: localhost:9100

3 SPRINGBOOT整合MYBATIS+ELASTICSEARCH+JDBC+MYSQL+WEB8

2.3 analyzer-ik的安装配置

用于替换elasticsearch的默认分词器，步骤如下：

1. 在git上下载对应的安装包
2. 使用maven将项目打包，命令：mvn clean; mvn compile; mvn package
3. 将生成的jar包解压到./elasticsearch/plugins 中
4. 重启elasticsearch, 显示plugin [analysis-ik] loaded, 如图5，配置成功
5. 测试分析器，使用postMan向9300端口发送Get请求，/_analyze?analyzer=ik_smarter

```
19:34:782[INFO ][o.e.p.PluginsService ] [QyuaBON] loaded module [x-pack-logstash]
19:34:784[INFO ][o.e.p.PluginsService ] [QyuaBON] loaded module [x-pack-ml]
19:34:785[INFO ][o.e.p.PluginsService ] [QyuaBON] loaded module [x-pack-monitoring]
19:34:787[INFO ][o.e.p.PluginsService ] [QyuaBON] loaded module [x-pack-rollup]
19:34:789[INFO ][o.e.p.PluginsService ] [QyuaBON] loaded module [x-pack-security]
19:34:791[INFO ][o.e.p.PluginsService ] [QyuaBON] loaded module [x-pack-sql]
19:34:792[INFO ][o.e.p.PluginsService ] [QyuaBON] loaded module [x-pack-upgrade]
19:34:793[INFO ][o.e.p.PluginsService ] [QyuaBON] loaded module [x-pack-watcher]
19:34:797[INFO ][o.e.p.PluginsService ] [QyuaBON] loaded plugin [analysis-ik]
19:34:570[INFO ][o.e.x.s.a.s.FileRolesStore] [QyuaBON] parsed [0] roles from file [E:\宸々赫瀛\笔记\代码\elasticsearch-6.5.2\config\roles.yml]
19:59:139[INFO ][o.e.x.m.j.p.l.CppLogMessageHandler] [QyuaBON] [controller/24224] [Main.cc@109] co
Version 6.5.2 (Build 767566e25172d6) Copyright (c) 2018 Elasticsearch BV
20:01:579[DEBUG][o.e.a.ActionModule ] [QyuaBON] Using REST wrapper from plugin org.elasticsearch
Security
20:04:589[INFO ][o.e.d.DiscoveryModule ] [QyuaBON] using discovery type [zen] and host provider
20:08:944[INFO ][o.e.n.Node ] [QyuaBON] initialized
20:08:945[INFO ][o.e.n.Node ] [QyuaBON] starting ...
```

图 5: analyzer_ik

3 springboot整合mybatis+elasticsearch+jdbc+mysql+web

项目的后端部分，使用IDEA intellj编译。创建项目的时候勾选上述五个选项即可创建项目，但是这时的项目还不够完整。需要做如下工作：

1. 配置application.yml 代码如图6所示。

3 SPRINGBOOT整合MYBATIS+ELASTICSEARCH+JDBC+MYSQL+WEB9

```
1 server:
2   port: 8080
3
4 spring:
5   datasource:
6     name: database
7     type: com.alibaba.druid.pool.DruidDataSource
8     #druid连接池配置
9     druid:
10      #监控统计拦截的filters
11      filters: stat
12      driver-class-name: com.mysql.jdbc.Driver
13      #基本属性
14      url: jdbc:mysql://127.0.0.1:3306/database?useUnicode=true&characterEncoding=UTF-8&allowMultiQueries=true&serverTimezone=GMT
15      username: root
16      password: root
17      #配置初始化大小/最小/最大
18      initial-size: 1
19      min-idle: 1
20      max-active: 20
21      #获取连接等待超时时间
22      max-wait: 60000
23      #间隔多久进行一次检测，检测需要关闭的空闲连接
24      time-between-eviction-runs-millis: 60000
25      #一个连接在池中最小生存的时间
26      min-evictable-idle-time-millis: 300000
27      validation-query: SELECT 'x'
28      test-while-idle: true
29      test-on-borrow: false
30      test-on-return: false
31      #打开PSCaches，并指定每个连接上PSCache的大小，mysql设为true，mysql设为false，分库分表较多推荐设置为false
32      pool-prepared-statements: false
33      max-pool-prepared-statement-per-connection-size: 20
34
35   data:
36     elasticsearch:
37       cluster-name: elasticsearch
38       cluster-nodes: 127.0.0.1:9200
39       repositories:
40         enabled: true
41
42   main:
43     allow-bean-definition-overriding: true
44
45   # 该配置节点为独立的节点，有很多同学容易将这个配置放在spring的节点下，导致配置无法被识别
46   mybatis:
47     mapper-locations: classpath:mapper/*.xml #注意：一定要对应mapper映射.xml文件的路径
48     type-aliases-package: com.news.springboot2.mybatisdemo.model #注意：对应实体类的路径
```

图 6: application.yml

2. 添加相关依赖，即pom.xml 添加如图7所示的依赖(其他依赖已经自动生成)。

3 SPRINGBOOT整合MYBATIS+ELASTICSEARCH+JDBC+MYSQL+WEB10

```
56
57
58 <dependency>
59   <groupId>com.fasterxml.jackson.core</groupId>
60   <artifactId>jackson-core</artifactId>
61 </dependency>
62 <dependency>
63   <groupId>com.fasterxml.jackson.core</groupId>
64   <artifactId>jackson-databind</artifactId>
65 </dependency>
66 <dependency>
67   <groupId>com.fasterxml.jackson.datatype</groupId>
68   <artifactId>jackson-datatype-joda</artifactId>
69 </dependency>
70 <dependency>
71   <groupId>com.fasterxml.jackson.module</groupId>
72   <artifactId>jackson-module-parameter-names</artifactId>
73 </dependency>
74 <!-- 分页插件 -->
75 <dependency>
76   <groupId>com.github.pagehelper</groupId>
77   <artifactId>pagehelper-spring-boot-starter</artifactId>
78   <version>1.2.5</version>
79 </dependency>
80 <!-- alibaba的druid数据库连接池 -->
81 <dependency>
82   <groupId>com.alibaba</groupId>
83   <artifactId>druid-spring-boot-starter</artifactId>
84   <version>1.1.9</version>
85 </dependency>
86
87 <dependency>
88   <groupId>org.springframework.boot</groupId>
89   <artifactId>spring-boot-starter-data-elasticsearch</artifactId>
90 </dependency>
91 <dependency>
92   <groupId>org.elasticsearch.plugin</groupId>
93   <artifactId>transport-netty3-client</artifactId>
94   <version>5.6.10</version>
95 </dependency>
```

图 7: pom.xml

3. 创建domain, dao,controller,service,serviceImpl,mapper等, 完善项目结构, 如图8所示。

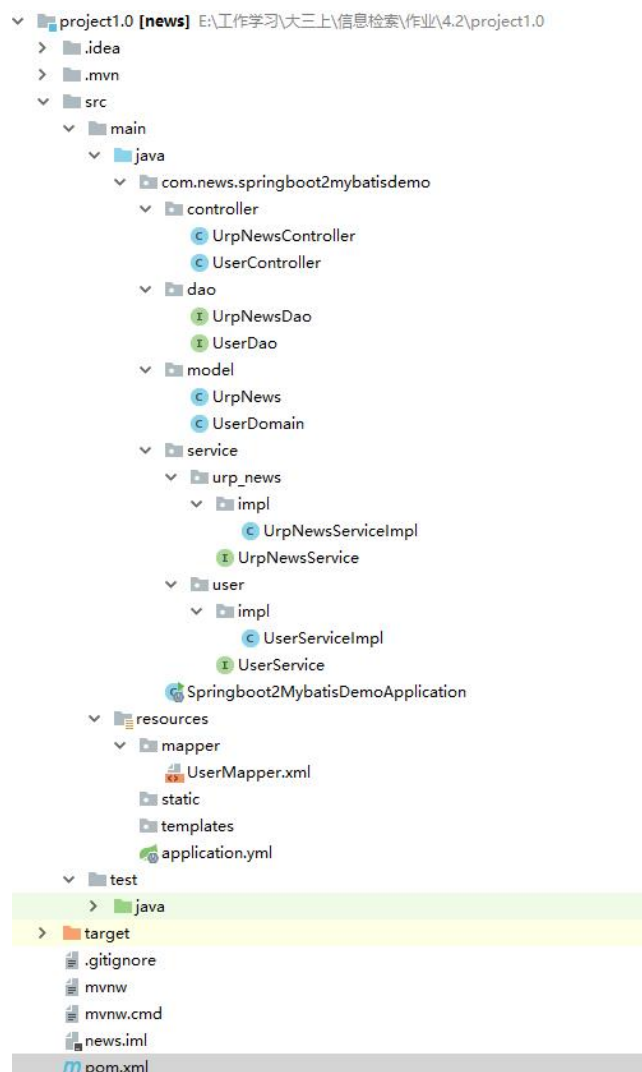


图 8: 项目结构

4 前端配置

使用vue.js2.0框架制作html页面，配置步骤如下：

1. 安装vue-cli
2. 创建vue前端项目，vue create front

3. 添加组件库npm i install element-ui -S, 在main.js中引用lib
4. 安装异步请求插件axios, npm install axios -save, 在main.js中定义全局变量\$axios并引用
5. npm run dev, 运行项目

第二部分 项目需求及实现

5 数据爬取

5.1 数据说明

按照要求，数据来自学校的10个网站，包括7个学院，信息门户以及南开大学新闻网的各类新闻，共10223个文档，如图9。

查看 5 个分片中共有的 5 个, 10223 命中, 耗时 0.025 秒

index	type	_id	_score	title	newsurl
news_r_1	doc	24	1	新华社：贯彻落实“两会”精神 开启全面建设社会主义现代化国家新征程	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	40	1	【学习贯彻十九大精神】“两会”精神进校园 南开大学举行报告会	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	44	1	我校师生在“五四”青年节之际 举行升旗仪式	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	48	1	共青团南开大学委员会 召开五四青年节座谈会	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	52	1	南开大学军事学研究生支部九月入团积极分子	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	60	1	杨振宁院士之妻：欣闻南开大学 百年华诞之际	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	84	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	92	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	108	1	周恩来同志诞辰100周年纪念大会在津举行	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	116	1	我校教师应邀参加“马克思主义学院专业设置三十周年纪念大会”	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	120	1	我校教师应邀参加“马克思主义学院专业设置三十周年纪念大会”	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	132	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	172	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	204	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	208	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	240	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	256	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	260	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	264	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	300	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	332	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	340	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	356	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	372	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?
news_r_1	doc	400	1	李俊卿院士应邀作题为“从马克思主义到新时代中国特色社会主义思想”的报告	http://cc.nankai.edu.cn/Pages/ArticlePage.aspx?

图 9: 新闻文档

5.2 scrapy爬取

使用scrapy创建爬虫项目, scrapy startproject Ip (由于安装Ctex的时候所有的环境变量都被清掉了，这里不再给出截图，另提醒：安装Ctex或者卸载之前都要备份以下环境变量。)

各个文件以及代码如下所示：

5.2.1 Ip.py

```
1 import scrapy
2 from liep.items import LiepItem
3
4
5 class LpSpider(scrapy.Spider):
6     name = 'lp'
7     allowed_domains = ['hyxy.nankai.edu.cn']
8     start_urls = ['http://hyxy.nankai.edu.cn/index.php/Home/News/newsMore/ci
9
10    # 主站链接 用来拼接
11    base_site = 'http://hyxy.nankai.edu.cn'
12
13    def parse(self, response):
14        news_urls = response.xpath('//ul[@class="articleList"]/li/a/@href')
15
16        for news_url in news_urls:
17            url = self.base_site + news_url
18            yield scrapy.Request(url, callback=self.getInfo)
19
20        # 获取下一页
21        next_page_url = self.base_site + response.xpath('//div[@class="manu
22
23        yield scrapy.Request(next_page_url, callback=self.parse)
24
25    def getInfo(self, response):
26        item = LiepItem()
27        # 提取信息
28        item['time'] = response.xpath('//*[@id="container"]/div/div/div/p/
29        item['title'] = response.xpath('/html/body/div[2]/div[2]/h1/text()')
30        item['newsUrl'] = response.url
31        item['pageHtml'] = response.body.decode('utf-8')
32        # item['viewNum'] = response.xpath('//*[@id="container"]/div/div/di
33        divs = response.xpath('/html/body/div[2]/div[2]/div')
34        body = ''
35        for p in divs.xpath('.//p/text()|.//p//span/text()'):
36            body += p.extract().strip()
37        item['content'] = body
38        yield item
```

图 10: Ip.py

5.2.2 settings.py

主要需要配置的是启动管道文件，控制爬虫的速率，添加代理等

```

8 BOT_NAME = 'liep'
9
10 SPIDER_MODULES = ['liep.spiders']
11 NEWSPIDER_MODULE = 'liep.spiders'
12
13 # Crawl responsibly by identifying yourself (and your website) on the user-agent
14 # USER_AGENT = 'liep (+http://www.yourdomain.com)'
15
16 # Obey robots.txt rules
17 ROBOTSTXT_OBEY = True
18
19 # Configure maximum concurrent requests performed by Scrapy (default: 16)
20 # CONCURRENT_REQUESTS = 32
21
22 # Configure a delay for requests for the same website (default: 0)
23 # See https://doc.scrapy.org/en/latest/topics/settings.html#download-delay
24 # See also autotrottle settings and docs
25 DOWNLOAD_DELAY = 1
26
27 # The download delay setting will honor only one of:
28 # CONCURRENT_REQUESTS_PER_DOMAIN = 16
29 # CONCURRENT_REQUESTS_PER_IP = 16
30
31 # Disable cookies (enabled by default)
32 COOKIES_ENABLED = False
33
34 DEFAULT_REQUEST_HEADERS = {
35     'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
36     'Accept-Language': 'en',
37     'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/42.0.2317.15 Safari/537.36'
38 }
39
40
41 USER_AGENTS = [
42     'Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; AcooBrowser; .NET CLR 1.1.4324.22; LBBROWSER)',
43     'Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; Acoo Browser; SLCC1; .NET CLR 2.0.50724; Media Center PC 6.0; .NET CLR 3.0.4606.546; .NET CLR 3.5.4606.546; LBBROWSER)',
44     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/6.0; LBBROWSER)',
45     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; LBBROWSER)',
46     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER)',
47     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; LBBROWSER; rv:11.0; LBBROWSER)',
48     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
49     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
50     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
51     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
52     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
53     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
54     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
55     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
56     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
57     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
58     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
59     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
60     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
61     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
62     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
63     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
64     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
65     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
66     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
67     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
68     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
69     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
70     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
71     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
72     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
73     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
74     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
75     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
76     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
77     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
78     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
79     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
80     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
81     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
82     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
83     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
84     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
85     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
86     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
87     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
88     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
89     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
90     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
91     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
92     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
93     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
94     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
95     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
96     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
97     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
98     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
99     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
100     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
101     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
102     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
103     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
104     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
105     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
106     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
107     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
108     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
109     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
110     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
111     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
112     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
113     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
114     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0; LBBROWSER; rv:11.0; LBBROWSER)',
115     'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.
```

图 11: settings.py

5.2.3 pipelines.py

管道文件，用于将爬到的数据写入数据库中

```
7 import json
8 import pymysql
9
10
11 class LiepPipeline(object):
12     def __init__(self):
13         # 建立数据库连接
14         self.connection = pymysql.connect(host='127.0.0.1', port=3306, user=
15                                         charset='utf8')
16
17         # 创建操作游标
18         self.cursor = self.connection.cursor()
19
20     def process_item(self, item, spider):
21         # 定义sql语句
22         self.cursor.execute(
23             """insert into database.hxyx_news(title,content,pageHtml,newsUrl)
24             value (%s, %s, %s, %s)""",
25             (
26                 item['title'],
27                 item['content'],
28                 item['pageHtml'],
29                 item['newsUrl'],
30             ))
31
32         # 保存修改
33         self.connection.commit()
34
35         return item
36
37     def __del__(self):
38         # 关闭操作游标
39         self.cursor.close()
40         # 关闭数据库连接
41         self.connection.close()
```

图 12: pipelines.py

5.2.4 items.py

创建对象，存储要采集的字段

```
1 #- coding: utf-8 -*-
2
3 # Define here the models for your scraped items
4 #
5 # See documentation in:
6 # https://doc.scrapy.org/en/latest/topics/items.html
7
8 import scrapy
9
10
11 class LiepItem(scrapy.Item):
12     # define the fields for your item here like:
13     # name = scrapy.Field()
14     #time = scrapy.Field()
15     title = scrapy.Field()
16     content = scrapy.Field()
17     pageHtml = scrapy.Field()
18     newsUrl = scrapy.Field()
19     #viewNum = scrapy.Field()
```

图 13: items.py

5.2.5 Middlewares.py

设置使用随机代理和随机浏览器访问

```
64 class UseragentMiddleware(object):
65
66     def process_request(self, request, spider):
67         print('-----QAQ-----')
68         useragent = random.choice(USER_AGENTS)
69         request.headers.setdefault('User-agent', useragent)
70         print('----->headers successful')
71         return None
72
73 class ProxyMiddleware(object):
74     def process_request(self, request, spider):
75         print('----->-----<-----')
76         proxy = random.choice(PROXIES)
77         print(proxy['ip_port'], proxy.get('user_password', None))
78         request.meta['proxy'] = "http://%s" % proxy.get('ip_port')
79
80         if proxy.get('user_password', None):
81             b64 = base64.b64encode(proxy.get('user_password'))
82             print(b64)
83             request.headers['Proxy-Authorization'] = 'Basic' + b64
84             print('----->proxy-----')
```

图 14: Middlewares.py

6 链接分析pageRank

pageRank体现了网页的相关性和重要性，在搜索引擎优化操作中是经常被用来评估网页优化的成效因素之一。令 R 表示所有 N 个网页的PageRank组成的列向量，令网页间的连接矩阵 $L = l_{ij}$ ， P_i 有链接指向 P_j 时， $l_{ij} = 1$ ，否则 $l_{ij} = 0$ 。对 L 的每行进行归一化，即用 P_i 的出度 N_i 去除得到矩阵 $A = a_{ij}$ ， $a_{ij} = l_{ij}/N_i$ ，则有：

$$R = cA_T R \iff c^{-1}R = A_T R$$

根据线性代数中有关特征向量和特征值的理论， R 是矩阵 A_T 的 c^{-1} 特征值对应的特征向量。下图给出了pagerank的计算：

```

255 private double[] pageRank() {
256     List<UrpNews> news = urpNewsDao.queryAllByIdExists();
257     double matrix[][] = new double[news.size()][news.size()];
258     double PR[] = new double[news.size()];
259     //初始化矩阵
260     for(int i = 0; i < news.size(); i++){
261         for(int j = 0; j < news.size(); j++){
262             matrix[i][j] = 0;
263         }
264     }
265     int j = 0;
266     for(UrpNews urpNews : news){
267         for(int i = 0; i < news.size(); i++){
268             int[] temp = getHref(urpNews.getPageHtml());
269             if(temp.length != 0){
270                 for(int t : temp){
271                     matrix[i][t] = 1/(double)t; //形成矩阵并归一化
272                 }
273             }
274         }
275     }
276     //计算pageRank
277     Matrix matrixPr = new Matrix(matrix);
278     matrix = matrixPr.eig().getD().getArray();
279     for(int i = 0; i < news.size(); i++){
280         PR[i] = matrix[i][i];
281     }
282     return PR;
283 }
284 //获取每个网页指向的网页
285 private int[] getHref(String pageHtml) {
286     Pattern p = Pattern.compile("href=\".*\""); //正则表达式
287     Matcher m = p.matcher(pageHtml); //匹配所有href="http://baidu.com" 的字符串
288     String url;
289     List<String> urls = new ArrayList<>();
290     if(m.find()){
291         url = m.group(0);
292         urls.add(url); //将m中的每个分组交给url
293     }
294     int j = 0;
295     int all[] = new int[urls.size()];
296     for(String i : urls){
297         all[j] = urpNewsDao.queryUrpNewsByUrl(i.split(" ")[1]).getId(); //提取出url
298         j++;
299     }
300     return all;
301 }
302 }
303 }

```

图 15: pageRank

7 查询服务

7.1 高亮展示

使关键词能够高亮显示，给用户以更好地视觉感受。在elasticSearch中提供了高亮搜索关键词的功能。但是有时content中并不含有关键词，也就是其返回值将会是空，这个时候，我们将原内容的前100个字符返回，作为展示预览的内容。实现及效果如下所示：



图 16: 高亮显示



图 17: 高亮实现

7.2 通配查询

在用户搜索“大学”时，他即有可能希望查找到“大学生”相关的信息，也有可能是想要查找“南开大学”，对于这种情况，如果只提供精确

查询，是难以做到的，这就需要使用通配查询来实现。通过查阅官方文档，在elasticsearch中提供了match方法，支持字符串的通配查询，根据官方文档所示，match方法在进行匹配的同时，还会根据词项出现的频率权重等以及用户的自定义设置对查询的文档进行评分，并返回排序后相应的文档。效果实现代码如图。



图 18: 通配查询

```
public List<UpNews> highlightQuery(@RequestParam String field, @RequestParam String type, @RequestParam String SearchMessage, @RequestParam Integer page) {
    NativeSearchQuery searchQuery = new NativeSearchQueryBuilder()
        .withQuery(QueryBuilders.boolQuery().must(QueryBuilders.matchQuery(field, SearchMessage)))
        .withPageable(PageRequest.of(page, 10))
        .withHighlightFields(new HighlightBuilder.Field(field).preTags("<span style='color: red'>").postTags("</span>"))
        .build();

    Page<UpNews> page = elasticSearchTemplate.queryForPage(searchQuery, UpNews.class, new SearchResultMapper() {
        @Override
        public <T> AggregatedPage<T> mapResults(SearchResponse searchResponse, Class<T> tClass, Pageable pageable) {
            ArrayList<UpNews> upNews = new ArrayList<UpNews>();
            SearchHits hits = searchResponse.getHits();

            for (SearchHit searchHit : hits) {
                if (hits.getHits().length == 0) {
                    return null;
                }
                else if (searchHit.getSourceAsMap() != null) {
                    UpNews upNews1 = new UpNews();
                    String highlightMessage = String.valueOf(searchHit.getSourceAsMap().get(field));

                    if (searchHit.getHighlightFields().size() != 0) {
                        highlightMessage = searchHit.getHighlightFields().get(field).fragments[0].toString();
                    }

                    upNews1.setId(Integer.parseInt(searchHit.getId()));
                    upNews1.setTitle(String.valueOf(searchHit.getSourceAsMap().get("title")));
                    upNews1.setContent(String.valueOf(searchHit.getSourceAsMap().get("content")));
                    upNews1.setNewsurl(String.valueOf(searchHit.getSourceAsMap().get("newsurl")));
                    upNews1.setPagehtml(String.valueOf(searchHit.getSourceAsMap().get("pagehtml")));
                    upNews1.setTime(String.valueOf(searchHit.getSourceAsMap().get("time")));
                    upNews1.setTotal(hits.getHits().length);
                    upNews1.setNewstype(String.valueOf(searchHit.getSourceAsMap().get("newstype")));
                }
            }
            return new AggregatedPage<UpNews>(upNews, pageable.getPageNumber(), pageable.getPageSize());
        }
    });

    return upNews;
}
```

图 19: 通配查询代码

7.3 精确查询: site

很多搜索引擎都会提供此项功能,用于使用户自定义搜索范围。此处设计的语法为: site:domain [keyword],在domain后接有空格,冒号为英文符号。采用此语法时,所有结果都来自于同一个域名。实现效果及代码如下所示:



图 20: site:news.nankai.edu.cn

```

search(searchMessage) {
  let real = searchMessage;
  this.newSearchMessage = searchMessage;
  if(tmp !== this.newSearchMessage) {
    this.currentPage = 1;
  }

  if(real.split(":")[0] === "site") {
    let temp = real.split(":")[1];
    this.url = temp.split(" ")[0];
    real = temp.split(" ")[1];
  }

  this.$axios.get(
    'http://localhost:8080/search', {
      params: {
        searchMessage:real,

```

图 21: 代码

7.4 分类查询

为了给用户更加明确的信息，设计了新闻的分类效果。按照新闻来自学院的不同进行了分类划分，共分为10个类型。参考elasticsearch的官方文档，提供了term方法，可以按照字段进行严格的查询，实现代码与效果如下所示：



图 22: 分类效果

```

if (newType.equals("all")) {
    searchQuery = new NativeSearchQueryBuilder()
        .withQuery(QueryBuilders.boolQuery().must(QueryBuilders.multiMatchQuery(searchMessage, _fieldNames: "title", "content").analyzer("ik_smart"))
        .must(QueryBuilders.matchQuery(_fieldName: "newstitle", url).operator(Operator.AND))
    )
    .withHighlightFields(new HighlightBuilder
        .Field(name: "title")
        .preTags("<span style = 'color : red;'>")
        .postTags("</span>"),
        new HighlightBuilder
        .Field(name: "content")
        .preTags("<span style = 'color:red;'>")
        .postTags("</span>"))
    .withPageable(pageable)
    .build();
} else {
    searchQuery = new NativeSearchQueryBuilder()
        .withQuery(QueryBuilders.boolQuery()
            .must(QueryBuilders.multiMatchQuery(searchMessage, _fieldNames: "title", "content").analyzer("ik_smart"))
            .must(QueryBuilders.matchQuery(_fieldName: "newstitle", url).operator(Operator.AND))
            .must(QueryBuilders.termQuery(_fieldName: "newstype", newType))
        )
        .withHighlightFields(new HighlightBuilder
            .Field(name: "title")
            .preTags("<span style = 'color : red;'>")
            .postTags("</span>"),
            new HighlightBuilder
            .Field(name: "content")
            .preTags("<span style = 'color:red;'>")
            .postTags("</span>"))
        .withPageable(pageable)
        .build();
}

```

图 23: 分类代码

7.5 语音输入

在chrome浏览器第11版之后，chrome就新加入了语音识别（speech recognition）功能，但这个功能并不支持所有的浏览器，而是仅仅针对chrome有效，所以声明对象的时候，添加了webkit前缀，能够使不支持该对象的浏览器也能跳过此功能，进行正确的解析。语音识别功能可以使用户更加方便的进行查询操作。功能演示将在录制的视频中展示。

```

}
speechInput() {
    let vm = this;
    recognition = new webkitSpeechRecognition();
    recognition.start();
    recognition.onresult = function(event) {
        if(event.results[0][0].confidence > 0.95) {
            console.log(event.results[0][0].transcript);
            vm.searchMessage = event.results[0][0].transcript;
            vm.openFullScreen();
            setTimeout(() => {
                vm.search(vm.searchMessage);
            }, 500);
        } else {
            vm.$message({
                type: 'warn',
                message: '没有听清楚哟'
            })
        }
    }
}
},

```

图 24: 语音输入代码

7.6 语音合成

与语音识别不同的是，语音合成功能支持所有的浏览器，所以在任何浏览器上运行本项目都可以体验到新闻朗读的功能，在这里考虑到用户的阅读问题，在开始新闻阅读的时候，会展示一个新的按钮——content，用来展示所有的新闻内容。

```
    speakNews(item) {  
      item.speechStop = true;  
      speaker.text = item.content;  
      window.speechSynthesis.speak(speaker);  
      console.log("speak news");  
    },  
  
    speakNewsStop(item) {  
      item.speechStop = false;  
      window.speechSynthesis.cancel();  
    },  
  
    snapshot(id) {  
      console.log(id);  
      const {href} = this.$router.resolve({  
        name: 'snapshot',  
        path: '/snapshot',  
        query: {  
          newsId:id  
        }  
      });  
      window.open(href, '_blank')  
      //this.$router.push({name:'snapshot',params:{newsHtml:html}});  
    },
```

图 25: 语音合成代码

7.7 网页快照

不通过访问链接，而是由搜索引擎直接返回页面的信息。在本项目中，由于爬虫的限制，仅提供静态页面，页面不会随着时间发生变化。点击url左侧三角符号，即可查看网页快照。实现及效果如下所示：

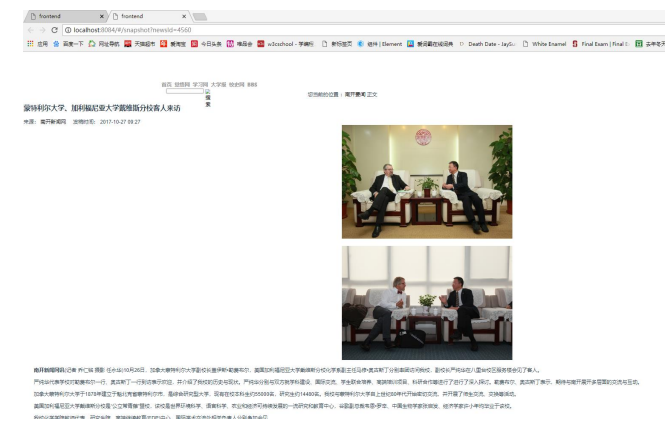


图 26: 快照

```
snapshot(id) {
  console.log(id);
  const {href} = this.$router.resolve({
    name: 'snapshot',
    path: '/snapshot',
    query: {
      newsId:id
    }
  });
  window.open(href, '_blank')
  //this.$router.push({name:'snapshot',params:{newsHtml:html}});
},
```

图 27: 实现

7.8 分页处理

elasticsearch提供参数pageable,pageable可以支持对每页的内容进行排序,接受page, size参数,分别设置页数和页大小,默认size=10,page=0。在创建接口时,设置了页数调整,但是没有设置页面size的调整,所以每次都是默认的10条记录。实现如下所示:


```
<div style="position: fixed;bottom: 0; margin-left: 135px;background-color: white;width: 100%;">
  <el-row>
    <el-col :span="6">
      <div class="block">
        <el-pagination
          @size-change="handleSizeChange"
          @current-change="handleCurrentChange"
          :current-page.sync="currentPage"
          :page-size="10"
          layout="total, prev, pager, next"
          :total="resultTotal">
        </el-pagination>
      </div>
    </el-col>
    <el-col :span="5" style="margin-top: 5px">
      <span class="time">当前 {{currentPage}} 页</span>
    </el-col>
  </el-row>
</div>
```

图 28: 分页处理