

信息检索导论课设作业

——新闻检索系统

时间：2017 秋季

课程：信息检索导论

贺翔宇 邓思艺 冯子朋 蒋正锴 解贺嘉 王琴琴 张继元

{hexiangyu17,dengsiyi17,fengzipeng17,jiangzhengkai17,

xiehejia17,wangqinqin17,zhangjiyuan17}

@mailsucas.ac.cn

目录

1 系统介绍.....	1
1.1 系统需求.....	1
1.2 系统思路与框架.....	1
2 设计方案.....	1
2.1 新闻爬取.....	1
2.1.1 算法简述.....	1
2.1.2 创新点.....	4
2.2 索引构建.....	4
2.3 检索模块.....	5
2.3.1 检索模式.....	5
2.3.2 检索排序.....	5
2.4 用户接口.....	6
2.4.1 查询自动补全.....	6
2.4.2 实时 snippet 生成.....	7
2.4.3 相关搜索推荐.....	7
2.4.4 热点新闻推荐.....	8
2.4.5 相似新闻推荐.....	8
2.4.6 情感分析.....	9
2.5 前端模块.....	9
3 测试报告.....	10
3.1 新闻爬取.....	10
3.2 索引测试.....	10
4 经验&不足.....	10
4.1 经验总结.....	10
4.2 不足之处.....	10
5 系统功能实现.....	11
5.1 基本功能.....	11
5.2 创新点.....	11
6 参考文献.....	11

1 系统介绍

1.1 系统需求

新闻检索系统：定向采集不少于 4 个中文社会新闻网站或频道，实现这些网站新闻信息及评论信息的自动爬取、抽取、索引和检索。本项目未使用 lucene，Goose 等成熟开源框架。

1.2 系统思路与框架

本系统总体的实现思路如图 1 所示：

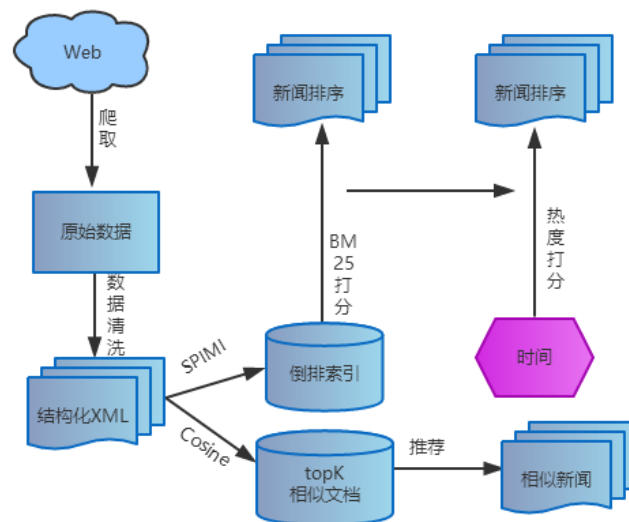


图 1 系统总体框架图

一个完整的搜索系统主要的步骤是：

- (1) 对新闻网页进行爬虫得到语料库；
- (2) 抽取新闻的主体内容，得到结构化的 xml 数据；
- (3) 内存式单遍扫描索引构建方法构建倒排索引，供检索模块使用；
- (4) 用户输入查询，得到相关文档返回给用户。

2 设计方案

2.1 新闻爬取

2.1.1 算法简述

该模块针对搜狐，网易，腾讯三大主流新闻网站及官方的参考消息网站进行

了新闻获取。并基于其网站结构，设计了不同的爬取模式。由于网站架构两两相似，以下选取两种类型的典型代表进行介绍：

（1）搜狐新闻

搜狐新闻除正常主页外，存在隐藏的列表式新闻页，如 <http://news.sohu.com/1/0903/62/subject212846206.shtml>，其新闻组织方式如下图所示：

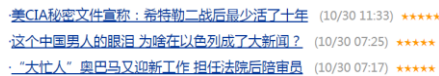


图 2 列表式新闻

源代码中以：

```
<a test=a href='http://www.sohu.com/a/201108961_115479' target='_blank'>美CIA秘密文件宣称：希特勒二战后最少活了十年</a><span> (10/30 11:33)</span>
```

图 3 新闻分割源代码

<a>作为各条新闻的分割，且含有新闻标题，发表时间，通过 Beautiful soup 该款开源格式解析工具，我们获取到列表中所有新闻的 URL，并记录其相应的标题，发表时间。注意 URL 的最后数字串对应于该条评论的 newsID。

新闻内容页中，正文部分以：

```
<div class="text">
  <div class="text-title">
    <h1>美国CIA秘密文件宣称：希特勒二战后最少活了十年<span class="article-tag">
```

图 4 新闻正文

<div class="text"> 包含正文部分，而评论消息正文部分记录于 <http://quan.sohu.com/pinglun/cyqemw6s1/+newsID> 所在网页，且通过 js 动态进行加载。针对这一问题，我们采用了 Selenium + PhantomJS 伪浏览器获取网页动态内容的方式，冒充真实用户请求，获取动态加载的评论。其评论以 的方式包裹，逐条进行呈现。而评论的数量，需要通过访问：

[http://changyan.sohu.com/api/3/topic/liteload?callback=jQuery1709683075909326675_1500009616671&client_id=cyqemw6s1&topic_url="+newsID](http://changyan.sohu.com/api/3/topic/liteload?callback=jQuery1709683075909326675_1500009616671&client_id=cyqemw6s1&topic_url=) 进行获取，返回结构体

```
{"cmt_sum":0,"comments":[],"hots":[],"mode":6,"outer_cmt_sum":0,"participation_sum":0,"topic_id":4285033212});
```

图 5 评论结构体

其中 cmt_sum 即为所求。

（2）网易新闻

可以将网易新闻及腾讯新闻归结为一般类型的新闻主页，我们采用了自新闻主页开始的广度优先的递归爬取策略。注意到新闻的正文页往往是静态网页.html，因此，我们将网页中出现的所有以.html 结尾的网页的 URL 均记录下来，在爬取到一定量时，进行一次去重。

对于一些不是新闻的错分网页，容错处理即通过检查新闻正文标签 <div post_content_main> 时会被剔除。

新闻正文页中我们重点关注内容，时间，评论获取：

正文起始：

```
<div class="post_content_main" id="epContentLeft">
<h1>外交部：法国总统马克龙将于1月8日至10日访问中国</h1>
```

新闻时间：

```
<div class="post_time_source">
2018-01-02 10:38:01 来源：
```

评论相关：

网易新闻为直接通过新闻的 URL 的末尾编号 (NewsID) 与关联评论的网址的组合：<http://comment.news.163.com/api/v1/products/a2869674571f77b5a0867c3d71db5856/threads/+NewsID> 获取评论数量及内容：

```
{
  "against": 0,
  "boardId": "tech_bbs",
  "channelId": "0009",
  "cmtAgainst": 27,
  "cmtVote": 7633,
  "createTime": "2018-01-02 09:00:53",
  "docId": "D74N3KK400097U7T",
  "isAudit": false,
  "modifyTime": "2018-01-02 10:29:24",
  "pdocId": "D74N3KK400097U7T",
  "recount": 688,
  "status": "normal",
  "source": "web",
  "joincount": "on",
  "web": "on",
  "audio": "off",
  "against": "on",
  "app": "on",
  "tcount": 562,
  "title": "特斯拉的尴尬：96万元的新车 三年后残值仅30多万",
  "url": "http://tech.163.com/18/0102/09/D74N3KK400097U7T.html",
  "vote": 40
}
```

图 6 新闻评论数量与内容

这里的 cmt 相关的数字，我们认为其评论的大致数量。

加上 /comments/newList?offset 即返回含有评论的结构体：

```
{
  "against": 0,
  "anonymous": false,
  "buildLevel": 1,
  "commentId": 514246,
  "content": "花10万以上买电动车的，都是脑子坏的。",
  "createTime": "2018-01-02 16:29:58",
  "favCount": 0,
  "ip": "139.228.*.*",
  "isDel": false,
  "postId": "D74N3KK400097U7T_514246",
  "productKey": "a2869674571f77b5a0867c3d71db5856",
  "shareCount": 0,
  "siteName": "网易",
  "source": "web",
  "unionState": false,
  "user": {
    "avatar": "http://img.cache.netease.com/timg/default100.png",
    "location": "上海市",
    "nickname": "user1",
    "realNameInfo": {
      "userId": 2436945,
      "vote": 0,
      "454508": {
        "against": 0,
        "anonymous": false,
        "buildLevel": 5,
        "commentId": 454508,
        "content": "大众的车本田丰田只要不是国产车和冷门车都差不多",
        "createTime": "2018-01-02"
      }
    }
  }
}
```

图 7 新闻评论结构体

腾讯新闻通过新闻正文主页中的 comment_id 实现对评论的检索：

```
document.domain = 'qq.com';
cmt_site = 'news';
cmt_id = 2331528973;
cmt_is_group = 0;
cmt_count_id = 'comment_count|comment_count2';
```

图 8 评论检索

之后仍然是跳转与评论相关的网址：http://coral.qq.com/article/+cmt_id

```
{
  "errorCode": 0,
  "data": {
    "appid": "10002",
    "targetid": "2331528973",
    "groupid": "",
    "orgcommentnum": "13251",
    "commentnum": "1510",
    "time": "1514882365"
  }
}
```

图 9 评论数量

返回结构体中的 commentnum 即为数量，加上 /comment?reqnum=num，即得到 json 形式的评论，

```
{
  "errorCode": 0,
  "data": {
    "targetid": "2331528973",
    "display": 1,
    "total": 1512,
    "reqnum": 50,
    "retnum": 51,
    "hasnext": true,
    "commentid": [
      {
        "id": "6353756050260927153",
        "rootid": "0",
        "targetid": "2331528973",
        "parent": "0",
        "timeDifference": "今天 08:38:09",
        "time": "1514853489",
        "content": "互利共赢，携手共进。",
        "title": "",
        "up": "3",
        "rep": "0",
        "type": "1",
        "hotScale": "0",
        "checktype": "0",
        "checkstatus": "1",
        "isdeleted": "0"
      }
    ]
  }
}
```

图 10 评论

我们对其转码进行解析即可，其中 content 即为所求：

```
"commentId": [
  {
    "id": "6353756050260927153",
    "rootid": "0",
    "targetid": "2331528973",
    "parent": "0",
    "timeDifference": "今天 08:38:09",
    "time": "1514853489",
    "content": "互利共赢，携手共进。",
    "title": "",
    "up": "3",
    "rep": "0",
    "type": "1",
    "hotScale": "0",
    "checktype": "0",
    "checkstatus": "1",
    "isdeleted": "0"
  }
]
```

图 11 解析转码

2.1.2 创新点

(1) 实现了对新闻网页**动态加载**的评论进行爬取，如搜狐新闻评论爬取。

(2) 未借助开源新闻爬取工具，自己实现了对新闻标题，正文，时间，评论内容，评论数目的高效爬取。[项目已开源](#)，被此次作业的其他组参考借鉴。

2.2 索引构建

(1) 分词，我们借助开源的 jieba 中文分词组件来完成，jieba 分词能够将一个中文句子切成一个个词项，这样就可以统计 tf, df 了；

(2) 去停用词，去停词的步骤在 jieba 分词之后完成；

(3) 倒排记录表存储，词典用 B-树或 hash 存储，倒排记录表用邻接链表存储方式，这样能大大减少存储空间。

倒排索引构建算法使用内存式单遍扫描索引构建方法 (SPIMI)，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。SPIMI 的伪代码如下：

```
SPIMI-INVERT(Token_stream)

    Output_file=NEWFILE()
    dictionary = NEWHASH()

    while (free memory available)

        do token <-next(token_stream) //逐一处理每个词项-文档ID对

            if term(token)∉dictionary

/*如果词项是第一次出现，那么加入hash词典，同时，建立一个新的倒
排索引表*/

                then postings_list = ADDTODICTIONARY(dictionary,term(token))

/*如果不是第一次出现，那么直接返回其倒排记录表，在下面添加其后*/

                else postings_list = GETPOSTINGLIST(dictionary,term(token))

                if full(postings_list)

                    then postings_list =DOUBLEPOSTINGLIST(dictionary,term(token))

/*SPIMI与BSBI的区别就在于此，前者直接在倒排记录表中增加此项新纪录
*/

                ADDTOPOSTINGSLIST (postings_list,docID(token))

sorted_terms <- SORTTERMS(dictionary)

WRITEBLOCKTODISK(sorted_terms,dictionary,output_file)

return output_file
```

运行代码之后会在 ./data/ 下生成一个 ir.db 数据库文件，这就是构建好的索引数据库。

2.3 检索模块

2.3.1 检索模式

(1) 关键词检索

查询即根据用户输入的关键字，返回其相应的新闻。首先根据用户的查询进行 jieba 分词，记录分词后词项的数量以字典形式进行存储。

```
def clean_list(self, seg_list):
    cleaned_dict = {}
    n = 0
    for i in seg_list:
        i = i.strip().lower()
        if i != '' and not self.is_number(i) and i not in self.stop_words:
            n = n + 1
            if i in cleaned_dict:
                cleaned_dict[i] = cleaned_dict[i] + 1
            else:
                cleaned_dict[i] = 1
    return n, cleaned_dict
```

图 12 关键词检索核心代码

根据每个词项返回的结果，利用 BM25 计算公式得到每个文档的分数，得到最终的文档排名。

(2) 布尔检索

对于布尔检索，这里只简单实现支持 OR, AND 操作，首先根据用户的查询，得到 OR 还是 AND，分别得到相应的文档序号，然后进行返回文档的 OR 和 AND 得到最终的文档结果。

```
def process_bool(self, seg_list):
    if 'OR' in seg_list:
        return 'OR'
    elif 'AND' in seg_list:
        return 'AND'
    else:
        return False
def intersection(self, doc1, doc2):
    doc = [val for val in doc1 if val in doc2]
    return doc
def unionset(self, doc1, doc2):
    return list(doc1.union(doc2))
```

图 13 布尔检索核心代码

2.3.2 检索排序

(1) 按相关度

算法简述

按新闻的相关度排序，相关度越高的新闻排名越高。检索模型中检索效果最好的是基于概率的 BM25 模型。

(2) 按时间

类似的，我们还可以对所有文档按时间先后顺序排序，越新鲜新闻排名越高。

(3) 按热度

1) 算法简述

按新闻的热度排序，越热门的新闻排名越高。关于热度公式，我们认为主要和三个因素有关：相关度，这个不需要过多解释，列出的新闻一定要和查询相关；用户评论数，评论数和点击数是呈正相关的，若一篇新闻下的评论越多，则表明该新闻更受用户关注；同时也要考虑时间因素，时间越久远，新闻的热度就会越低。所以热度公式是 BM25 打分、评论数打分和时间打分的一个综合。

比较有名的热度公式有两个，一个是 Hacker News，另一个是 Reddit。他们均为将新闻/评论的一个原始得分和时间组合起来，只是一个用除法，一个用加法。我们借鉴 reddit 的热度公式，将其稍作简化，并添加评论数打分，开创如下的热度公式：

$$hot_{score} = k_1 \log(BM25_{score}) + k_2 num_{comments} + \frac{k_3}{t_{now} - t_{news}}$$

将 BM25 得分、评论数、新闻时间和当前时间的差值的倒数线性加权，k1、k2、k3 均是可调参数。

2) 创新点

与典型的热度公式相比，添加了评论数的指标，更精准的刻画“热度”的含义。

2.4 用户接口

2.4.1 查询自动补全

(1) 功能介绍

1) 用户在输入框输入搜索内容时，根据用户的输入内容，自动完成数据匹配和前端展示，最多展示十条。

2) 用户可以通过键盘方向键浏览。

3) 用户可以通过鼠标的点击和键盘的回车选中需要的数据项并更新显示在输入框中，同时实现搜索。

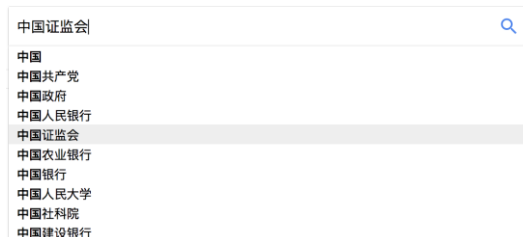


图 14 查询自动补全效果

(2) 算法简述

1) 使用 html 和 js 实现前端展示和用户行为监听。其中，ajax 实现前端和后端的数据传输。

2) 匹配数据部分, 我们使用 SQL 语句, SELECT term FROM postings WHERE term like 'term%' ORDER BY df DESC LIMIT 10, 来对数据进行匹配。其中, 我们以用户输入的内容为前缀在倒排索引表内进行数据匹配, 同时根据文档频率进行降序排序, 也就是说, 出现在文档中次数多的词条会被提示。同时使用 ' ? ' 赋值的方式, 防止 SQL 注入, 保证了数据库的安全。

3) 键盘方向键浏览补全, 鼠标点击提示内容或者键盘回车, 实现自动查询。

2.4.2 实时 snippet 生成

(1) 算法简述

该模块给用户提供了一个更加方便快捷的视角查看检索结果中检索词出现的语境和上下文, 从而快速排查是否为所需信息。其核心思想是对检索词所在句子进行整合和凸显。具体做法如下:

1) 利用 jieba 对检索词进行切分, 并去除停用词, 过滤, 最终得到检索切分词表。

2) 在新闻正文中用 “。|?|!|#|,|……|~|;|:|~|,” 对中文句子进行简单切分, 对每个分句进行判断, 若存在任一检索切分词则将其突出, 其他分句均用省略号表示。

2.4.3 相关搜索推荐

(1) 算法简述

该模块针对用户的搜索给用户提供更相关的搜索词或者用户可能想了解的相关信息。其核心思想是提取用户检索结果中除检索词之外的关键词。具体做法如下:

1) 提取出前二十个检索结果的标题, 若检索结果没有 20 条则取全部检索结果的标题;

2) 用 TextRank 算法对所取标题组成的整个文本进行关键词的抽取;

3) 按照重要值取前 10 个关键词作为推荐搜索词。

(2) 创新点

1) 在没有大量用户日志数据的情况下, 也能够较准确地推荐相关检索词, 且耗费成本较少, 根据排序依据不同, 推荐检索词也会随之变化, 较为灵活。

2) 在无法直接得知用户所需信息时, 转换思路, 利用用户初次搜索的结果中的重要信息作为相关推荐, 并经过了经验和实践的认证。

3) 使用 TextRank 算法进行关键词抽取。TextRank 算法是一种用于文本的基于图的排序算法。其基本思想来源于谷歌的 PageRank 算法, 通过把文本分割成若干组成单元(单词、句子)并建立图模型, 利用投票机制对文本中的重要成分

进行排序，仅利用单篇文档本身的信息即可实现关键词提取、文摘。和 LDA、HMM 等模型不同，TextRank 不需要事先对多篇文档进行学习训练，因其简洁有效而得到广泛应用。

2.4.4 热点新闻推荐

(1) 算法简述

该算法与按热度排序的算法相似，只是去掉与查询相关度的信息，表示为评论数、新闻时间和当前时间的差值的倒数线性加权。

另外与按热度排序的算法不同的是，按热度排序的算法应用于检索出的相关文档，而该算法应用于全部的文档，选出评分最高的前 K 个新闻放在主页上显示。公式如下：

$$hot_{score} = k_1 num_{comments} + \frac{k_2}{t_{now} - t_{news}}$$

其中，k1、k2 均是可调参数。

(2) 创新点

将最热新闻置于网站首页，更大限度的让用户捕捉到热点信息。

2.4.5 相似新闻推荐

算法简述

当用户浏览某条具体新闻时，我们在页面底端给出 5 条和该新闻相关的新闻。推荐模块的思路是度量两两新闻之间的相似度，取相似度最高的前 5 篇新闻作为推荐阅读的新闻。

一篇文档可以用一个向量表示，向量中的每个值是不同词项 t 在该文档 d 中的词频 tf 。但是一篇较短的文档（如新闻）的关键词并不多，所以我们可以提取每篇新闻的关键词，用这些关键词的 $tf*idf$ 值构成文档的向量表示，这样能够大大减少相似度计算量，同时保持较好的推荐效果。

jieba 分词组件自带关键词提取功能，并能返回关键词的 $tf*idf$ 值。所以对每篇新闻，我们先提取 $tf*idf$ 得分最高的前 25 个关键词，用这 25 个关键词的 $tf*idf$ 值作为文档的向量表示。由此能够得到一个 $1000*m$ 的文档词项矩阵 M ，矩阵每行表示一个文档，每列表示一个词项， m 为 1000 个文档的所有互异的关键词（大概 10000 个）。 M 是稀疏矩阵。

得到文档词项矩阵 M 之后，我们利用 sklearn 的 `pairwise_distances` 函数计算 M 中行向量之间的 cosine 相似度，对每个文档，得到与其最相似的前 5 篇新闻 id，并把结果写入数据库。

2.4.6 情感分析

(1) 算法简述

该模块给用户提供了对每条新闻评论的情感极性分析,并且对每篇新闻所有新闻的总体评价极性趋势以及情绪波动情况进行了统计分析。其核心思想是利用情感词典给评论进行情感分析。具体做法如下:

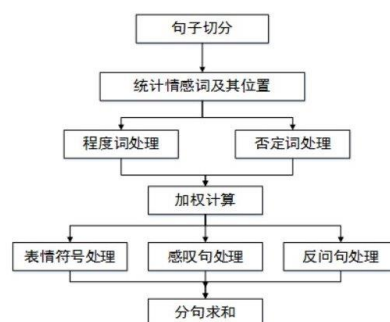


图 15 评论情感分析流程

1) 切分句子。利用标点符号(, . ! ? ; ~ , . ! ? ; ~ …)对句子进行切分。

2) 切词并清理。对每个句子进行 jieba 分词,并去除停用词,过滤。

3) 判断句子中是否出现情感词典中的词,以及多个情感词之间的位置;若出现,判断前后一定范围内是否出现程度词表中的词,判断句子是否是感叹句反问句等情感强烈的句式以及是否存在表情符号。根据以上判断,分别赋予相应的分值加权。累计每个分句得到每条评论的极性分析。

4) 统计所有评论的情感分析数据,统计正向评论占比和负向评论占比,以及情感分析值的均值和方差,对应总体极性偏向和评论之间的情感波动情况。

(2) 创新点

1) 利用情感词典,无需进行监督学习即可快速进行极性判断,且正确率还可以。该方法快速便捷,适应网络用户的需要。

2) 情感词典涵盖较丰富,综合现有开放的资源,包括褒贬词及其近义词、汉语情感词极值表、清华大学李军中文褒贬义词典、台湾大学 NTUSD 简体中文情感词典以及知网 Hownet 情感词典。

3) 不仅对每条句子进行了极性分析,也对总体评论进行了分析统计,能给用户一个较为直观的分析结果。

2.5 前端模块

前端部分采用 flask 框架搭建,用 bootstrap 框架进行美化,模板匹配采用 jinja2 模板引擎搜索界面风格及配色模仿谷歌,前端效果展示见视频。

3 测试报告

3.1 新闻爬取

达到了 24h 内爬取 10W 条新闻及其评论的要求。实际测试中，该速度在很大程度上取决于带宽及对方服务器的响应速度。如不考虑评论更新问题，除腾讯新闻外，其他三类网站的 30W 条新闻可以在 5h 以内完成爬取。（笔者机器：Intel Xeon(R) E3-1220v5 @3.00GHz x 4; 32GB Memory; 1000Mbit/s 网卡）

3.2 索引测试

查询时间达到了平均不超过 2 秒。界面简洁大方，将热点新闻置于首页，用户查询后将排序方式置于检索框下方方便用户选择，点击 view 可以预览，新闻正文界面清晰明了，用户体验良好。

4 经验&不足

4.1 经验总结

（1）新闻爬取

出于礼貌爬取的考虑，我们并没有采用多线程爬取的策略。在实际中即使是单线程爬取，腾讯新闻也对爬取的速度进行了限制。

（2）自动补全

前缀补全可以满足基本需求。但是可以有更加智能的补全方式，例如加入句子的补全，不只限于词条。

（3）相关搜索推荐

有关用户推荐，在数据量并不足够的情况下可以寻求别的思路来代替无法直接获得的信息。

（4）评论情感分析

对网络搜索引擎而言，快速地提供信息是最主要的目的，因此在正确率尚可的情况下应采用快速而便捷的方法。

4.2 不足之处

（1）热度排序

为提高运算效率，热度排序公式只是简单地线性加权，没有更复杂的考虑。

（2）相关搜索推荐

缺少足够的用户数据,无法对用户做个性化推荐。

大部分功能依赖于分词效果,尤其在网络环境中,用户检索词多样化且新词快速出现,要适应不断增大的网络词库,分词模块应该不断优化改良。

(3) 热点新闻推荐

热点新闻推荐需要对全部新闻计算分值,当新闻数据量过大时效率会比较低。

5 系统功能实现

5.1 基本功能

检索系统功能中**没有实现通配符检索并用布尔检索代替**,其他功能全部实现。

5.2 创新点

(1) 新闻爬取

实现了对新闻网页动态加载的评论进行爬取,如搜狐新闻评论爬取。未借助开源新闻爬取工具,自己实现了对新闻标题,正文,时间,评论内容,评论数目的高效爬取。

(2) 检索排序

与典型的热度公式相比,添加了评论数的指标,更精准的刻画“热度”的含义。

(3) 相关搜索推荐

在没有大量用户日志数据的情况下,也能够较准确地推荐相关检索词,根据排序依据不同,推荐检索词也会随之变化;在无法直接得知用户所需信息时,利用用户初次搜索的结果中的重要信息作为相关推荐;使用 TextRank 算法进行关键词抽取

(4) 评论情感分析:

利用情感词典,无需进行监督学习即可快速进行极性判断;情感词典涵盖较丰富,综合现有开放的资源;不仅对每条句子进行了极性分析,也对总体评论进行了分析统计。

6 参考文献

[1] 开源代码使用:

BeautifulSoup: 开源 XML 或 HTML 解析库, 参考解析爬取网页内容

Jieba: 开源中文分词, 参考分词模块及 tf-idf 计算

textrank4zh: 开源中文语句关键词解析, 参考新闻关键词提取

news-search-engine: 开源中文搜索引擎, 参考架构设计及相关新闻推荐

[2] 王斌. 信息检索导论 [M]. 北京: 人民邮电出版社, 2010.

[3] 朱广文. 信息检索系统的设计与实现[D]. 哈尔滨工业大学, 2007.