体育新闻整合与检索系统设计文档

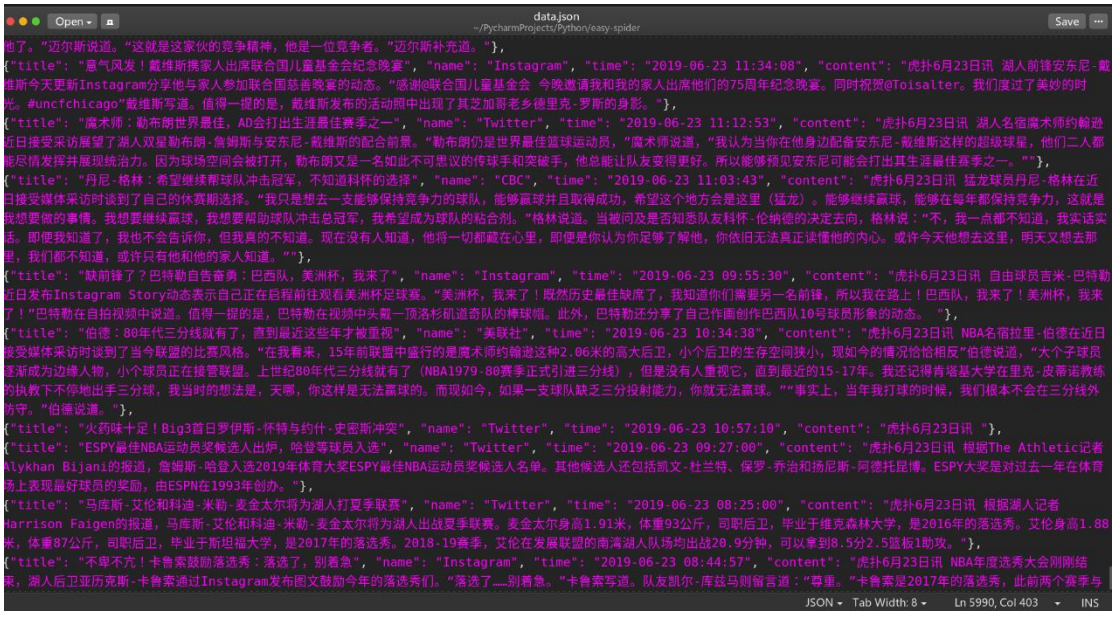
励永辉

一. 系统介绍

体育新闻整合与检索系统主要包含球队热度榜与搜索界面，可以满足用户搜索相关新闻以及浏览 NBA 球队详细信息的需求。

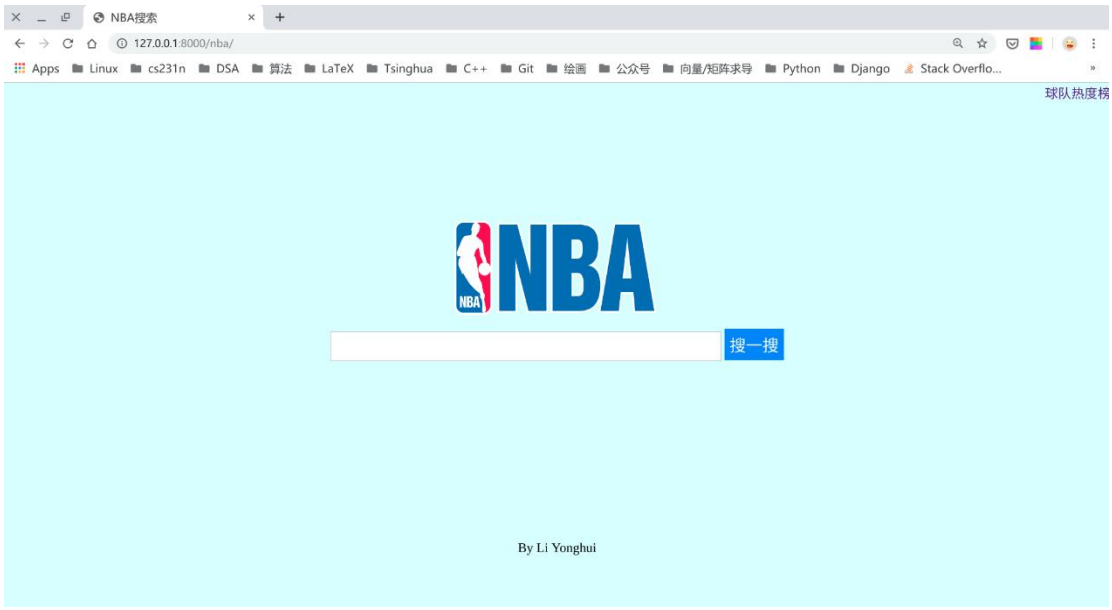
二. 功能展示

a) 数据爬取



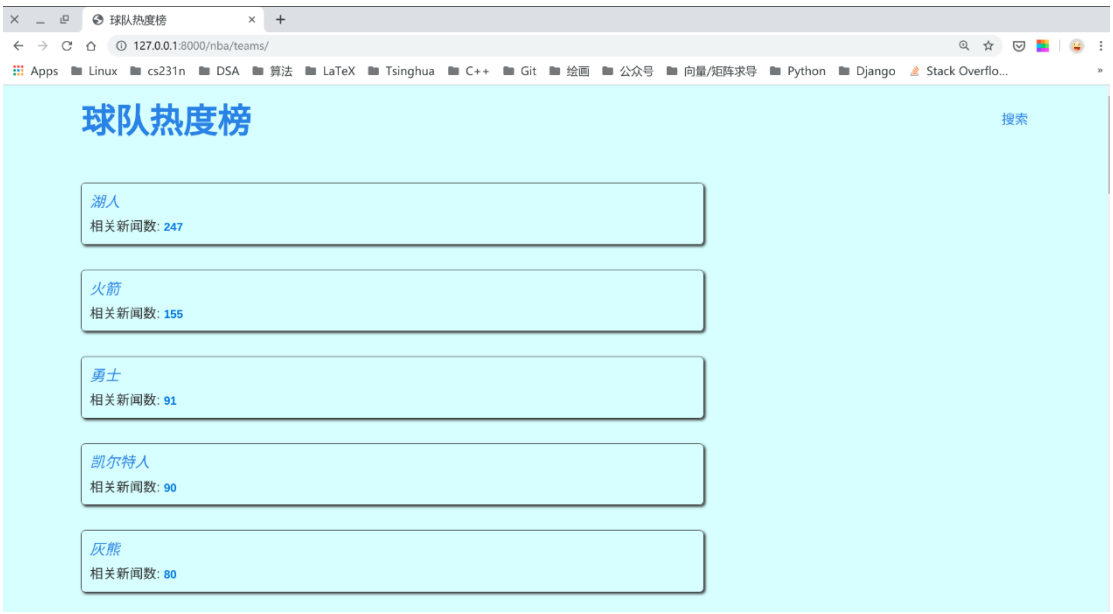
用 scrapy 总计爬取新闻 5998 条，保存在.json 文件中。

b) 搜索



在搜索栏输入目标词汇，可获取相关新闻。支持多关键词搜索。匹配的词汇会高亮显示，便于查找。

c) 球队热度版



页面包含一个列表，显示球队名称以及球队相关新闻数量等信息，并按照球队的相关新闻数量对球队排序。点击页面中的球队名称可跳转到球队主页。

d) 球队主页

球队主页包含两部分内容：基本信息与相关新闻。新闻数量较多时，进行分页等操作以方便用户交互

e) 新闻详情页

每条新闻有一个新闻详情页，新闻详情页中显示新闻标题与全文。为新闻正文中包含的球队名与运动员姓名添加超链接，跳转到本系统的球队主页。

三. 性能统计信息

总共用爬虫爬取新闻总数 5998 条。但由于建立倒排索引的时候过于耗时，抽取其中 1500 条新闻建立倒排索引，所以搜索功能只能查询到 1500 条新闻中的内容。

高级搜索查询时间稳定在毫秒量级。

建立了 30 个球队的个人主页，记录了 554 名球员信息。

四．设计思路

利用 Scrapy 爬取虎扑新闻，过滤其中域名开头为 https://voice.hupu.com/nba/的，得到 NBA 新闻，保存在 JSON 文件中。

Django 读取 JSON 中的文件，保存在 Sqlite3 数据库中。

Django 有三个模型 Team、Person、News，分别记录队伍信息、球员信息、新闻信息。

```
class Team(models.Model):
    name_text = models.CharField(max_length=200)
    time_text = models.CharField(max_length=200)
    city_text = models.CharField(max_length=200)
    news_int = models.IntegerField(default=0)
    def __str__(self):
        return self.name_text


class Person(models.Model):
    question = models.ForeignKey(Team, on_delete=models.CASCADE)
    firstname_text = models.CharField(max_length=200, null=True)
    lastname_text = models.CharField(max_length=200, null=True)

    def __str__(self):
        return self.lastname_text


class News(models.Model):
    team = models.ForeignKey(Team, on_delete=models.CASCADE)
    title_text = models.TextField(max_length=200)  # Field name made lowercase.
    posttime_text = models.TextField(max_length=200)  # Field name made lowercase.
    author_text = models.TextField(max_length=200)  # Field name made lowercase.
    content_text = models.TextField(max_length=200000)  # Field name made lowercase.

    def __str__(self):
        return self.title_text
```

五．具体技术说明

a) 分词

利用 jieba 库的 lcut 中对正文和标题进行分词，建立正排表，统计出文本中出现过的所有词汇。

```
all_words = []
for i in News.objects.all():
    cut = lcut(i.title_text+i.content_text)
    all_words.extend(cut)

set_all_words = set(all_words)
```

b) 索引

再遍历一遍建立倒排索引，把倒排索引的表储存在 JSON 文件中。

```
invert_index = dict()
for b in set_all_words:

    temp = []
    for j in range(1, News.objects.count()):

        field = News.objects.get(pk=j)

        split_field = lcut(field.title_text+field.content_text)

        if b in split_field:
            temp.append(j)
    invert_index[b] = temp
```

c) 搜索匹配程度

首先建立 TF-IDF，用来分析字词的重要性指标，评价多关键词搜索时文档与用户查询之间的相关程度。

GET 到用户的查询字符串后，利用 jieba 分词，再读取 TF-IDF 中列表中每个词的权重。然后对于每一遍文章，计算其所包含所有被搜索字符的总权重。对文章的总权重排序，降序输出。