

# 系统说明和设计文档

## 运行方式

- 运行 `sduspider/run.py` 来进行网络爬虫
- 运行 `indexbuilder/index_builder.py` 来对数据库中的数据构建索引
- 运行 `indexbuilder/query.py` 来测试搜索功能。
- 运行 `searchengine/run_server.py` 打开搜索网页服务器，在浏览器中打开 **127.0.0.1:8000** 进入搜索页面执行搜索。

## 所需python库

- scrapy
- requests
- pymongo
- whoosh
- jieba
- django

## 所需数据库

- MongoDB
- Studio 3T 导出csv文件

## 爬虫特性

爬取百度贴吧 `nba吧` 的内容。

爬虫代码位于 `Tieba/` 目录下。

由于待爬取的网页结构较简单, 只需要修改 `pn=` 后面的数字即可爬取。

通过查看网页结构, 发现每一页都有50个待爬数据, 每一页的 `pn` 都是50的倍数, 因此设置起始站点的代码为

```
1 start_urls = [f'https://tieba.baidu.com/f?kw=nba&ie=utf-8&pn={page * 50}' for page in range(0, 2000)]
```

再利用待爬取的数据的 `xpath` 爬取数据。

## 爬取的字段

## 爬取的字段

title

introduction

author

reply

last\_reply\_time

url

```
1  # 爬取当前网页
2      print('start parse : ' + response.url)
3      print("开始了开始了")
4
5      selectors = response.xpath('//*[id="thread_list"]/li')
6      print(selectors)
7      if response.url.startswith("https://tieba.baidu.com/"):
8          item = items.TiebaItem()
9          for selector in selectors[2:]:
10             url = selector.xpath(
11                 '._//div[@class="threadlist_title pull_left j_th_tit
12                 "]/a/@href').get()
13             if url: # 会员情况与非会员的xpath不一样, 判断一下非会员的是否读成功, 失败的话
14                 # 就表示是会员的, 要重新读一遍
15                 url = "https://tieba.baidu.com" + url
16             else:
17                 url = selector.xpath(
18                     '._//div[@class="threadlist_title pull_left j_th_tit
19                     member_thread_title_frs "]/a/@href').get()
20                 url = "https://tieba.baidu.com" + url
21                 md5url = md5(url)
22                 if self.binary_md5_url_search(md5url) > -1: # 存在当前MD5
23                     print("有重复!!!!!!!!!!!!!!")
24                     pass
25                 else:
26                     title = selector.xpath(
27                         '._//div[@class="threadlist_title pull_left j_th_tit
28                         member_thread_title_frs "]/a/text()').get()
29                     if not title:
30                         title = selector.xpath('._//div[@class="threadlist_title
31                         pull_left j_th_tit "]/a/text()').get()
32                     introduction = selector.xpath(
33                         '._//div[@class="threadlist_abs threadlist_abs_onlyline
34                         "]/text()').get()
35                     introduction = introduction.strip()
36                     author = selector.xpath(
37                         '._//span[@class="tb_icon_author
38                         "]/a[@rel="noreferrer"]/text()').get()
39                     reply = selector.xpath(
40                         '._//span[@class = "threadlist_rep_num
41                         center_text"]/text()').get()
42                     last_reply_time = selector.xpath(
43                         '._//span[@class = "threadlist_reply_date pull_right
44                         j_reply_data"]/text()').get()
45                     last_reply_time = last_reply_time.strip()
46                     item['title'] = title
47                     item['introduction'] = introduction
48                     item['author'] = author
49                     item['reply'] = reply
50                     item['last_reply_time'] = last_reply_time
51                     item['url'] = url
52                     item['urlmd5'] = md5(url)
53                     # 索引构建flag
```

```

45         item['indexed'] = 'False'
46         self.binary_md5_url_insert(md5url)
47         self.destination_list.append(url)
48         print('已爬取网址数: ' + (str)(len(self.destination_list)))
49         # yield it
50         yield item
51
52         # print("title: " + title, count)
53         # print("introduction: " + introduction)
54         # print("author: ", author)
55         # print("reply number: " + reply)
56         # print("last reply time = " + last_reply_time)
57         # print("url = ", url)
58         # print(" \n")
59
60     print("结束了")

```

但是在爬取的过程中发现有很多重复的数据，仔细检查之后发现是百度贴吧本身有很多重复的数据，在200页之后的每一页几乎都是相同的，从而导致了大量数据重复。

因此加了去重的代码。

## 将 url 加密为md5

```

1  def md5(val):
2      import hashlib
3      ha = hashlib.md5()
4      ha.update(bytes(val, encoding='utf-8'))
5      key = ha.hexdigest()
6      return key

```

## 二分法 md5 集合排序插入 self.url\_md5\_set--16 进制 md5 字符串集

```

1  def binary_md5_url_insert(self, md5_item):
2      low = 0
3      high = len(self.url_md5_seen)
4      while (low < high):
5          mid = (int)(low + (high - low) / 2)
6          if self.url_md5_seen[mid] < md5_item:
7              low = mid + 1
8          elif self.url_md5_seen[mid] >= md5_item:
9              high = mid
10     self.url_md5_seen.insert(low, md5_item)

```

二分法查找 url\_md5 存在于 self.url\_md5\_set 的位置，不存在返回 -1

```

1  def binary_md5_url_search(self, md5_item):
2      low = 0
3      high = len(self.url_md5_seen)
4      if high == 0:
5          return -1
6      while (low < high):

```

```

7         mid = (int)(low + (high - low) / 2)
8         if self.url_md5_seen[mid] < md5_item:
9             low = mid + 1
10        elif self.url_md5_seen[mid] > md5_item:
11            high = mid
12        elif self.url_md5_seen[mid] == md5_item:
13            return mid
14        if low >= self.url_md5_seen.__len__():
15            return -1
16        if self.url_md5_seen[low] == md5_item:
17            return low
18        else:
19            return -1

```

在每次存入数据之前,先检查当前的 `url` 是否已经被存入, 如果已经被存入, 那么不存入; 如果没有被存入, 那么存入。

```

1         if self.binary_md5_url_search(md5url) > -1: # 存在当前MD5
2             print("有重复!!!!!!!!!!!!!!")
3             pass
4         else:
5             ...

```

在管道文件 `pipelines.py` 中设置数据库接口, 存入数据。

```

1 class MongoDBPipeline(object):
2     def __init__(self):
3         host = settings["MONGODB_HOST"]
4         port = settings["MONGODB_PORT"]
5         dbname = settings["MONGODB_DBNAME"]
6         sheetname = settings["MONGODB_SHEETNAME"]
7         # 创建MONGODB数据库链接
8         client = pymongo.MongoClient(host=host, port=port)
9         # 指定数据库
10        mydb = client[dbname]
11        # 存放数据的数据库表名
12        self.post = mydb[sheetname]
13
14    def process_item(self, item, spider):
15        data = dict(item)
16        # self.post.insert(data) # 直接插入的方式有可能导致数据重复
17        # 更新数据库中的数据, 如果upsert为True, 那么当没有找到指定的数据时就直接插入, 反之不执行
18        插入
19
20        self.post.update({'urlmd5': item['urlmd5']}, data, upsert=True)
21        return item

```

通过 `Studio 3T` 可以查看爬取下来的数据

Quickstart × news ×								
(localhost:27017) > new_db > news								
Query 0								
Projection 0								
Sort 0								
Skip								
Limit								
Result Query Code Explain								
Documents 1 to 50								
Table View								
news > title								
_id	title	introduction	author	reply	last_reply_time	url	urlmd5	indexed
5f2f42305c74...	这样重排2008年选秀大会， 同曾吗？	这样重排2008年...	巧克力之...	75	13:06	https://tieba.baidu.com/p/6868694134	a1c366dfb89e...	False
5f2f42305c74...	这里是NBA球迷总群， 星球迷的聊天殿堂...	这里是NBA球迷...	接卸	358	13:05	https://tieba.baidu.com/p/6870052674	26e8c79acce4...	False
5f2f42305c74...	请问这是哪场比赛？ 第几节？		意料后进	9	13:03	https://tieba.baidu.com/p/6869921367	9d59171a56d7...	False
5f2f42305c74...	步行者给力点！ 拿下这场比赛送湖人一...		手放开...	319	13:05	https://tieba.baidu.com/p/6870068470	5e71598ba8c8...	False
5f2f42305c74...	已经买了开拓者票， 送一把温床， 给湖人...	已经买了开拓者...	永远	153	09:02	https://tieba.baidu.com/p/6868510765	c1b60d399712...	True
5f2f42305c74...	我卡走步绝杀如同	我卡走步绝杀同...	赛之冠	11	12:50	https://tieba.baidu.com/p/6869867658	2c794cd5575e...	True
5f2f42305c74...	盖帽鱼雷 NBA历史上五大选秀新秀：哪一属...	盖帽鱼雷NBA历...	邪恶契约	95	08:49	https://tieba.baidu.com/p/6866158227	e038e4aa6a1e...	True
5f2f42305c74...	加内特性格什么水平		加内特性格什么...	223	12:57	https://tieba.baidu.com/p/6868677426	ad1dece9f213...	False
5f2f42305c74...	为错误的球员轮休纠正正确轮休， 而换到老...	为错误的球员轮...	吴徐徐徐...	16	10:24	https://tieba.baidu.com/p/6866230884	9a8a3b348f10...	False
5f2f42305c74...	盖帽鱼雷21世纪最成功的NBA球员前30榜...	盖帽鱼雷21世纪...	贴吧用户...	7	13:04	https://tieba.baidu.com/p/6869834226	562cd769b43f...	False
5f2f42305c74...	不懂就问 历史上除了浓眉还有哪个球员一...	不懂就问 历史...	曙光大道	86	13:03	https://tieba.baidu.com/p/6868635035	ba513d39a184...	False
5f2f42305c74...	都说乔丹那么厉害， 为啥在历史仅有的9...		想字字...	375	11:05	https://tieba.baidu.com/p/6862497795	1ec49dd32ce8...	False
5f2f42305c74...	客观的来说， 如果开拓者能进季后赛， 湖...		第九先生	25	11:50	https://tieba.baidu.com/p/6855860198	8b587437275a...	True
5f2f42305c74...	乐福是不是NBA颜值天花板？	乐福是不是NBA...	不要早餐...	204	11:47	https://tieba.baidu.com/p/6857711066	254cebae1fe69...	True
5f2f42305c74...	【真大白】 警方公布调查结果：科比是...	众所周知， 那个...	常山机翻...	13	08:17	https://tieba.baidu.com/p/6714341215	748635b63a41...	False
5f2f42305c74...	詹姆斯能睡到张曼源吗？		魏晨魏你...	169	08:16	https://tieba.baidu.com/p/6702236962	b25d3644dae7...	False
5f2f42305c74...	曾经在很长一段时间99-07年左右 邓肯取...	曾经在很长一段...	白衣秀儿	173	11:35	https://tieba.baidu.com/p/6843570112	093d6ab822b6f...	True
5f2f42305c74...	NBA现役球员进入名人堂的概率一览	第一：詹姆斯10...	陈虎虎originat	269	12:58	https://tieba.baidu.com/p/6869060963	246fa418333cf...	False
5f2f42305c74...	如果詹姆斯给湖人带来2个总冠军， 詹姆斯...	如果詹姆斯给湖...	一梦生辰	93	10:16	https://tieba.baidu.com/p/6860766624	5ff23bfdf55f...	False
5f2f42305c74...	说实话很久没逛n吧， 得有6-7年没有结果...	说实话很久没逛...	叙述到的...	571	13:06	https://tieba.baidu.com/p/6866076366	66014b1ef44e...	False
5f2f42305c74...	科比职业生涯超级数据， 你怎么看？	科比职业生涯超...	气泡水	80	09:02	https://tieba.baidu.com/p/6856776352	1dc5ced8581...	True
5f2f42305c74...	湖人又被虐了	大家怎么看？		13	11:54	https://tieba.baidu.com/p/6870293163	0eb4edec108f...	True
5f2f42305c74...	湖人击败步行者， 谁的锅。	湖人击败步行者...	null	49	11:41	https://tieba.baidu.com/p/6870294693	db34eeae37e4...	True
5f2f42305c74...	整整以来的球员场均得分：1.哈登 - 33...	整整以来的球员...	N吧第一...	82	11:46	https://tieba.baidu.com/p/6868536236	9fa562e990ce...	True
5f2f42305c74...	你认为NBA最被高估的现役球员是谁？	你认为NBA最被...	cocoss	162	8-8	https://tieba.baidu.com/p/6857719506	052e14ed8e1a...	False
5f2f42305c74...	步行者狂虐湖人116:111我宣布， 淘汰我永...	步行者狂虐湖人...	哈哈敬爱...	26	12:09	https://tieba.baidu.com/p/6870300410	fdaf583178c05...	True
5f2f42305c74...	湖人真有希望， 领先几个， 不讨论谁谁谁	湖人真有希望...	王者荣耀...	6	08:22	https://tieba.baidu.com/p/6870296850	6a2d36f94d92...	False

## 索引构建特性

索引构建代码位于 `indexbuilder/` 目录下。

## 中文分词

Whoosh自带的Analyzer分词仅针对英文文章，而不适用于中文。从jieba库中引用的ChineseAnalyzer保证了能够对Documents进行中文分词。同样，ChineseAnalyzer在search时也能够对中文查询query提取关键字并进行搜索。

```
1 analyzer = ChineseAnalyzer()
2
3 # 创建索引模板
4 schema = Schema(
5     Id=ID(stored=True),
6     title=TEXT(stored=True, analyzer=analyzer),
7     url=ID(stored=True),
8     reply=NUMERIC(stored=True, sortable=True),
9     author=TEXT(stored=True),
10    last_reply_time=TEXT(stored=True),
11    introduction=TEXT(stored=True, analyzer=analyzer),
12 )
```

## Query类提供搜索API

Query类自动执行了从index索引文件夹中取倒排索引来执行搜索，并返回一个结果数组。

```
1 if __name__ == '__main__':
2     q = Query()
3     q.standard_search('')
```

## 搜索引擎特性

搜索引擎代码位于 `searchengine/` 目录下。

## Django搭建Web界面

Django适合Web快速开发。result页面继承了main页面，搜索结果可以按照result中的指示显示在页面中。在django模板继承下，改变main.html中的页面布局，result.html的布局也会相应改变。

```
1 def search(request):
2     res = None
3     if 'q' in request.GET and request.GET['q']:
4         res = q.standard_search(request.GET['q']) # 获取搜索结果
5         c = {
6             'query': request.GET['q'],
7             'resAmount': len(res),
8             'results': res,
9         }
10    else:
11        return render_to_response('main.html')
12
13    return render_to_response('result.html', c) # 展示搜索结果
```

示例界面



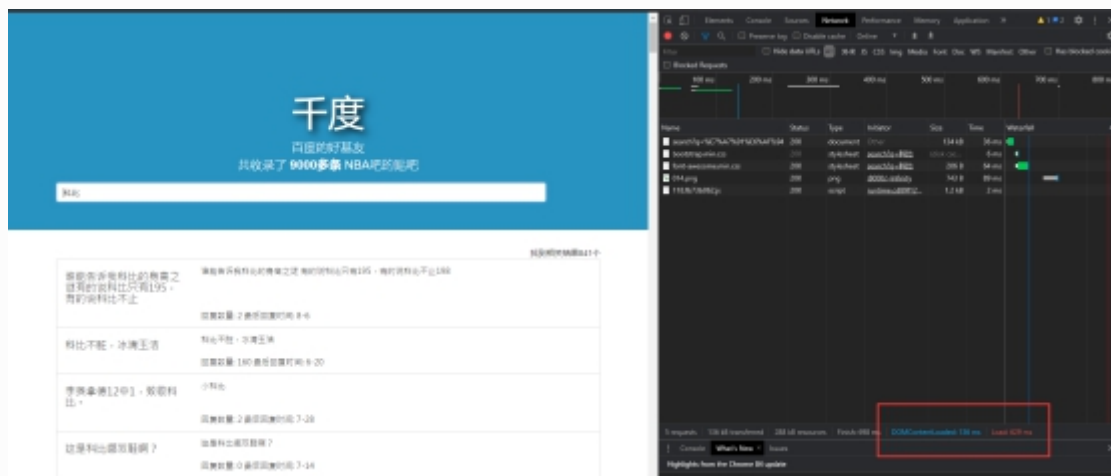
## 搜索引擎评估结果

### 1、Query1：科比

	URL	是否相关
1	<a href="https://tieba.baidu.com/p/6861920352">https://tieba.baidu.com/p/6861920352</a>	是
2	<a href="https://tieba.baidu.com/p/5568318409">https://tieba.baidu.com/p/5568318409</a>	是
3	<a href="https://tieba.baidu.com/p/6843823788">https://tieba.baidu.com/p/6843823788</a>	是
4	<a href="https://tieba.baidu.com/p/6811342090">https://tieba.baidu.com/p/6811342090</a>	是
5	<a href="https://tieba.baidu.com/p/6485852642">https://tieba.baidu.com/p/6485852642</a>	是

Precision@5: 5/5=1

Responding time: 629ms

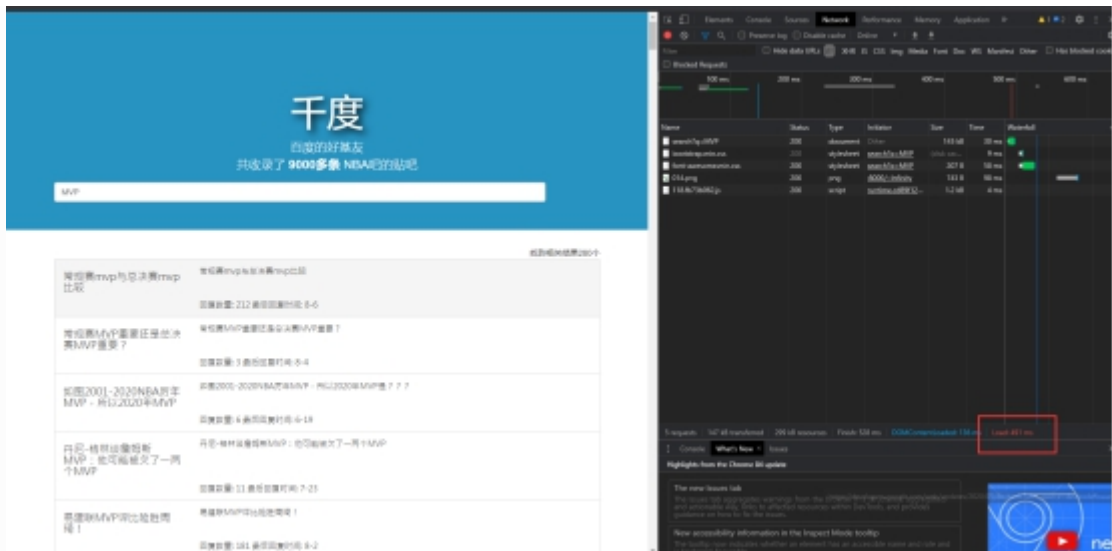


## 2、Query2: MVP

	URL	是否相关
1	<a href="https://tieba.baidu.com/p/6710552149">https://tieba.baidu.com/p/6710552149</a>	是
2	<a href="https://tieba.baidu.com/p/6859952281">https://tieba.baidu.com/p/6859952281</a>	是
3	<a href="https://tieba.baidu.com/p/6403404483">https://tieba.baidu.com/p/6403404483</a>	是
4	<a href="https://tieba.baidu.com/p/6830017137">https://tieba.baidu.com/p/6830017137</a>	是
5	<a href="https://tieba.baidu.com/p/6849737670">https://tieba.baidu.com/p/6849737670</a>	是

Precision@5: 5/5=1

Responding time: 491ms

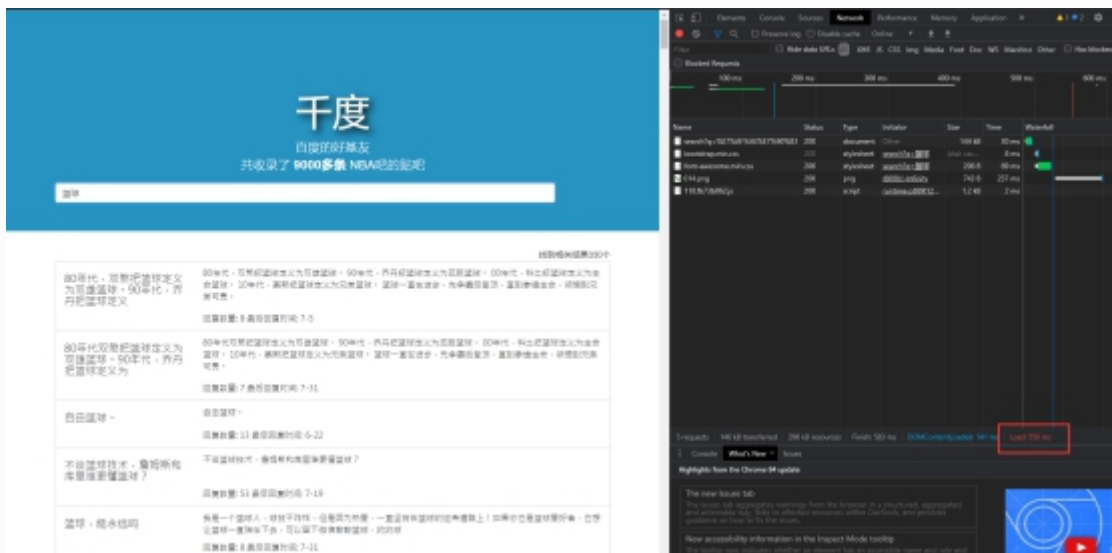


### 3、Query3：篮球

	URL	是否相关
1	<a href="https://tieba.baidu.com/p/6783779720">https://tieba.baidu.com/p/6783779720</a>	是
2	<a href="https://tieba.baidu.com/p/6847910160">https://tieba.baidu.com/p/6847910160</a>	是
3	<a href="https://tieba.baidu.com/p/6738287761">https://tieba.baidu.com/p/6738287761</a>	是
4	<a href="https://tieba.baidu.com/p/6172464331">https://tieba.baidu.com/p/6172464331</a>	是
5	<a href="https://tieba.baidu.com/p/6340825427">https://tieba.baidu.com/p/6340825427</a>	是

Precision@5: 5/5=1

Responding time: 550ms



### 4、Query4：科比和邓肯



	URL	是否相关
1	<a href="https://tieba.baidu.com/p/6795232184">https://tieba.baidu.com/p/6795232184</a>	是
2	<a href="https://tieba.baidu.com/p/6433023087">https://tieba.baidu.com/p/6433023087</a>	是
3	<a href="https://tieba.baidu.com/p/5728242690">https://tieba.baidu.com/p/5728242690</a>	是
4	<a href="https://tieba.baidu.com/p/6785343202">https://tieba.baidu.com/p/6785343202</a>	是
5	<a href="https://tieba.baidu.com/p/6838650416">https://tieba.baidu.com/p/6838650416</a>	是

Precision@5: 5/5=1

Responding time: 437ms



## 5、Query5: 篮板王

	URL	是否相关
1	<a href="https://tieba.baidu.com/p/6868635035">https://tieba.baidu.com/p/6868635035</a>	是
2	<a href="https://tieba.baidu.com/p/6741871545">https://tieba.baidu.com/p/6741871545</a>	是
3	<a href="https://tieba.baidu.com/p/6870308031">https://tieba.baidu.com/p/6870308031</a>	是
4	<a href="https://tieba.baidu.com/p/6812117274">https://tieba.baidu.com/p/6812117274</a>	否
5	<a href="https://tieba.baidu.com/p/6833705399">https://tieba.baidu.com/p/6833705399</a>	否

Precision@5: 4/5 = 0.8

Responding time: 440ms



	URL	是否相关
1	<a href="https://tieba.baidu.com/p/6806665073">https://tieba.baidu.com/p/6806665073</a>	是
2	<a href="https://tieba.baidu.com/p/6852223264">https://tieba.baidu.com/p/6852223264</a>	是
3	<a href="https://tieba.baidu.com/p/6144345739">https://tieba.baidu.com/p/6144345739</a>	是
4	<a href="https://tieba.baidu.com/p/6849654796">https://tieba.baidu.com/p/6849654796</a>	是
5	<a href="https://tieba.baidu.com/p/6864602741">https://tieba.baidu.com/p/6864602741</a>	是

Precision@5:  $5/5 = 1$

Responding time: 524ms



## 8、Query8: 湖人总决赛

	URL	是否相关
1	<a href="https://tieba.baidu.com/p/6834114334">https://tieba.baidu.com/p/6834114334</a>	是
2	<a href="https://tieba.baidu.com/p/6857330801">https://tieba.baidu.com/p/6857330801</a>	是
3	<a href="https://tieba.baidu.com/p/6119351434">https://tieba.baidu.com/p/6119351434</a>	否
4	<a href="https://tieba.baidu.com/p/6772497317">https://tieba.baidu.com/p/6772497317</a>	是
5	<a href="https://tieba.baidu.com/p/6785474060">https://tieba.baidu.com/p/6785474060</a>	是

Precision@5:  $4/5 = 0.8$

Responding time: 440ms





# 数据来源链接

<https://tieba.baidu.com/f?ie=utf-8&kw=nba>