

Hadoop wordcount demo

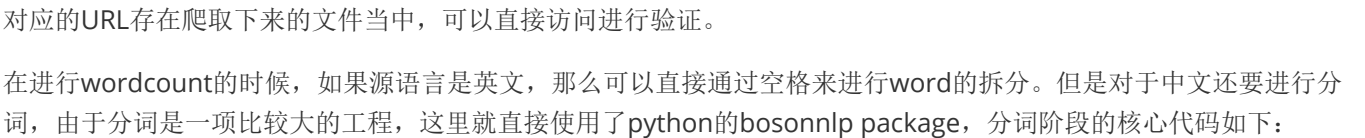
小组成员



关于数据爬取，爬取了20条网易新闻，采用了python scrapy框架，核心代码如下：

```
1  # -*- coding: utf-8 -*-
2  import scrapy
3  import re
4  from news_spider.items import News
5
6  class NeteaseSpiderSpider(scrapy.Spider):
7      name = "netease_spider"
8      PAGE_SIZE = 25
9      # start_urls = ['http://3g.163.com/touch/article/list/BBM54PGAwangning/0-5000.html']
10     start_urls = ['http://3g.163.com/touch/article/list/BBM54PGAwangning/0-20.html']
11     header = {"User-Agent": "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36\
12             (KHTML, like Gecko) Ubuntu Chromium/56.0.2924.76 Chrome/56.0.2924.76 Safari/537.36"}
13     news_count = 0
14
15     def start_requests(self):
16         yield scrapy.Request(url=self.start_urls[0], headers=self.header, callback=self.parse)
17
18     def parse(self, response):
19         content = response.xpath('//body/p/text()').extract()[0]
20         # with io.open('news.txt', 'w') as f:
21         #     f.write(content)
22         urls = re.findall(
23             'http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[*\\(\\),]|(?:%[0-9a-fA-F][0-9a-fA-F]))+',
24             content)
25
26         for url in urls:
27             if url.endswith('.html'):
28                 yield scrapy.Request(url=url, callback=self.parse_news)
29
30
31     def parse_news(self, response):
32         item = News()
33         item['url'] = response.url
34         item['title'] = response.xpath('/html/body/h1/text()').extract()[0]
35         content = ''
36         ps = response.xpath('//body/div[@class="content"]/p')
37         for p in ps:
38             content += p.xpath('./text()').extract()[0]
39         item['content'] = content
40
41         return item
```

The screenshot shows a Linux desktop environment. At the top is a menu bar with options: File, Edit, View, Go, Bookmarks, and Help. Below the menu bar is a panel with icons for various applications: a terminal, a file manager, a web browser, a music player, a video player, a presentation viewer, a calculator, a clock, and a system monitor. The main window is a file manager showing the contents of a folder named 'news_20'. The folder contains 12 sub-folders, each with a document icon and a title. The titles are: '为何呼格案和聂树斌案的精神损害赔偿金破百万', '假老公?女子结婚五年与丈夫无夫妻生活 初吻都在', '外交部回应日媒质疑 印尼雅万高铁:准备工作正推进', '居民家辐射值突然增高 原是邻居家藏放射性装置', '巴黎华人副区长:死者或患精神病 邻居称其"疯子"', '广州加强住房限购:须取得不动产证满2年后才可转让', '新疆乌恰县发生3.3级地震 震源深度16千米', '朴槿惠女警卫引发关注:身高175cm 武术高强', '湖北男子东莞"勇斗劫匪"被刺身亡 嫌疑人在逃', '男子不满女友分手将其掐死"扶"尸体到朋友家大睡', '男子持刀砍人被警方开枪制服 警察手部被刀划伤', '美国务院雇员受隋向中国提供情报?外交部回应', '这位副部级官员为了购选送出30个苹果手机', '重庆一大厦发生火灾致13人伤 其中2人不治身亡', '韩国国会通过呼吁中国停止"反萨"措施决议案', '韩法院对朴槿惠审问结束 是否批捕31日凌晨见分晓', and '马方:允许在马朝鲜公民离境并交还朝鲜男子遗体'.



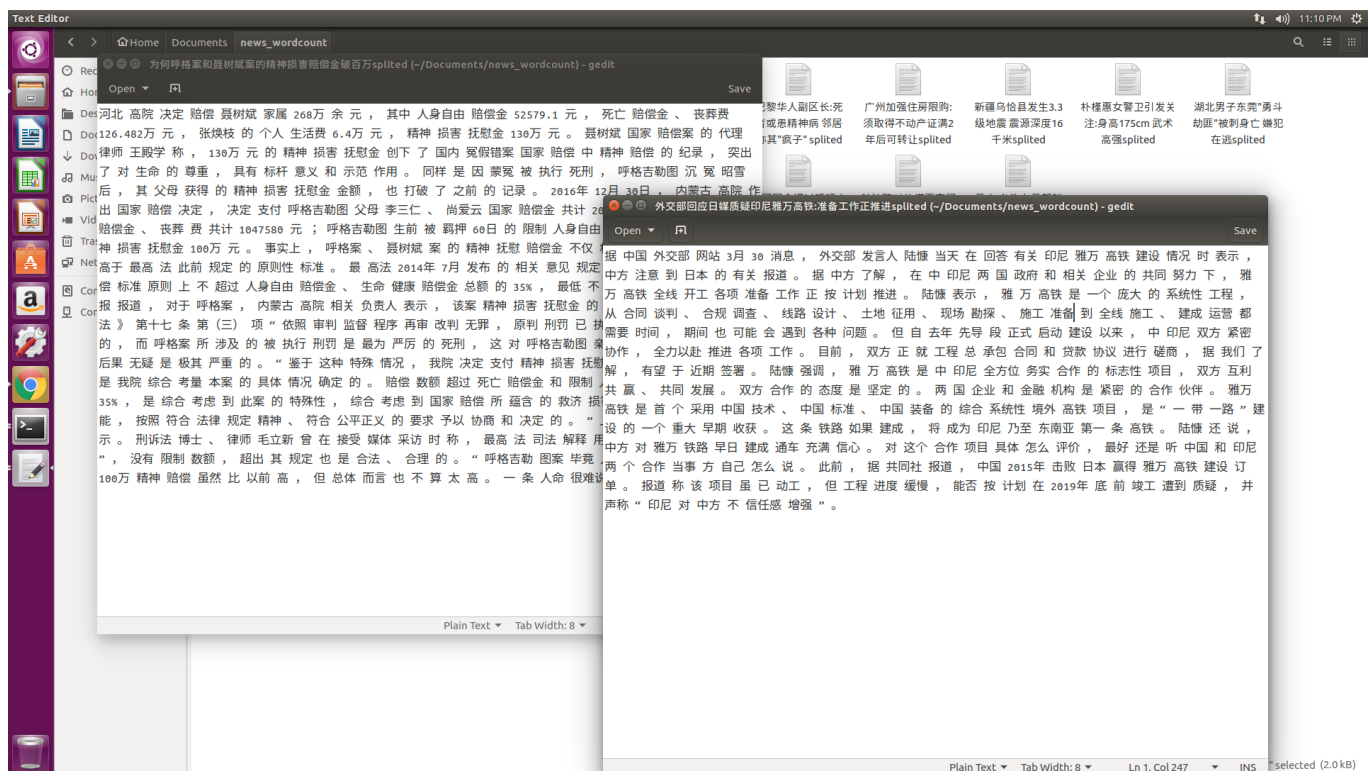
在进行wordcount的时候，如果源语言是英文，那么可以直接通过空格来进行word的拆分。但是对于中文还要进行分词，由于分词是一项比较大的工程，这里就直接使用了python的bosonnlp package，分词阶段的核心代码如下：

```

1 import os
2 import io
3 from bosonnlp import BosonNLP
4
5 # Get news file path
6 news_dir = '/home/ewan/Documents/news_spider/news_20/'
7 file_list = os.listdir(news_dir)
8
9 # Get the news dict
10 news = {}
11 for index, file in enumerate(file_list):
12     with io.open(news_dir + file, 'r') as f:
13         news[index] = file.decode('utf-8') + '\n' + f.read()
14
15 nlp = BosonNLP('FuHSE7Vf.13924.jadflTdrQLWx')
16 splitted_news = {}
17
18 SIZE = len(news)
19
20 for key, value in news.items():
21     result = nlp.tag([value])
22     words = ''
23     for index, word in enumerate(result[0]['word']):
24         words += word + ' '
25     splitted_news[key] = words
26
27 for index, content in splitted_news.items():
28     with io.open(str(index), 'w') as f:
29         f.write(content)

```

最后得到分词后的文件



下面贴出mapreduce的java代码:

```

1  /**
2   * Licensed to the Apache Software Foundation (ASF) under one
3   * or more contributor license agreements. See the NOTICE file
4   * distributed with this work for additional information
5   * regarding copyright ownership. The ASF licenses this file
6   * to you under the Apache License, Version 2.0 (the
7   * "License"); you may not use this file except in compliance
8   * with the License. You may obtain a copy of the License at
9   *
10  *     http://www.apache.org/licenses/LICENSE-2.0
11  *
12  * Unless required by applicable law or agreed to in writing, software
13  * distributed under the License is distributed on an "AS IS" BASIS,
14  * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
15  * See the License for the specific language governing permissions and
16  * limitations under the License.
17  */
18  package org.apache.hadoop.examples;
19
20  import java.io.IOException;
21  import java.util.StringTokenizer;
22
23  import org.apache.hadoop.conf.Configuration;
24  import org.apache.hadoop.fs.Path;
25  import org.apache.hadoop.io.IntWritable;
26  import org.apache.hadoop.io.Text;
27  import org.apache.hadoop.mapreduce.Job;
28  import org.apache.hadoop.mapreduce.Mapper;
29  import org.apache.hadoop.mapreduce.Reducer;
30  import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
31  import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
32  import org.apache.hadoop.util.GenericOptionsParser;
33
34  public class WordCount {
35
36      public static class TokenizerMapper
37          extends Mapper<Object, Text, Text, IntWritable>{
38
39          private final static IntWritable one = new IntWritable(1);
40          private Text word = new Text();
41
42          public void map(Object key, Text value, Context context
43              ) throws IOException, InterruptedException {
44              StringTokenizer itr = new StringTokenizer(value.toString());
45              while (itr.hasMoreTokens()) {
46                  word.set(itr.nextToken());
47                  context.write(word, one);
48              }
49          }
50      }
51
52      public static class IntSumReducer
53          extends Reducer<Text,IntWritable,Text,IntWritable> {
54          private IntWritable result = new IntWritable();
55

```

```

56     public void reduce(Text key, Iterable<IntWritable> values,
57                        Context context
58                        ) throws IOException, InterruptedException {
59         int sum = 0;
60         for (IntWritable val : values) {
61             sum += val.get();
62         }
63         result.set(sum);
64         context.write(key, result);
65     }
66 }
67
68 public static void main(String[] args) throws Exception {
69     Configuration conf = new Configuration();
70     String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
71     if (otherArgs.length < 2) {
72         System.err.println("Usage: wordcount <in> [<in>...] <out>");
73         System.exit(2);
74     }
75     Job job = Job.getInstance(conf, "word count");
76     job.setJarByClass(WordCount.class);
77     job.setMapperClass(TokenizerMapper.class);
78     job.setCombinerClass(IntSumReducer.class);
79     job.setReducerClass(IntSumReducer.class);
80     job.setOutputKeyClass(Text.class);
81     job.setOutputValueClass(IntWritable.class);
82     for (int i = 0; i < otherArgs.length - 1; ++i) {
83         FileInputFormat.addInputPath(job, new Path(otherArgs[i]));
84     }
85     FileOutputFormat.setOutputPath(job,
86         new Path(otherArgs[otherArgs.length - 1]));
87     System.exit(job.waitForCompletion(true) ? 0 : 1);
88 }
89 }

```

Pig Latin:

```

1  A = load '/tmp/demo.txt';
2  B = foreach A generate flatten(TOKENIZE((chararray)$0)) as word;
3  C = filter B by word matches '\\w+';
4  D = group C by word;
5  E = foreach D generate COUNT(C), group;
6  store E into '/tmp/wordcount_output';

```

word count结果

1	"	6	
2	(2	
3)	2	
4	1	6	
5	1000	1	
6	100万	3	
7	1047580	1	
8	10分	1	
9	10日	1	
10	10时	2	
11	10月	1	
12	10点	1	
13	11	1	
14	110万	1	
15	11时	1	
16	11点	1	
17	12041.40	1	
18	126.482万	1	
19	12月	2	
20	13	2	
21	130万	2	
22	15.6万	1	
23	16千	2	
24	175	2	
25	188	1	
26	189	1	
27	19	1	
28	1979年	1	
29	19时	1	
30	19点10分	1	
31	1月	1	
32	2	7	
33	20	1	
34	2011年	1	
35	2012年	4	
36	2013年	3	
37	2014年	1	
38	2015年	2	
39	2016年	3	
40	2017	1	
41	2017年	2	
42	2019年	1	
43	2059621.40	1	
44	207	1	
45	21日	1	
46	23日	2	
47	25日	2	
48	26	2	
49	260万	1	
50	268万	1	
51	26日	1	
52	27	2	
53	27日	1	
54	28日	2	
55	29日	2	

56 3 3
57 3.3 2
58 30 5
59 300 1
60 30分 1
61 30日 14
62 31日 4
63 35 2
64 35% 2
65 38 1
66 39.91 1
67 3月 12
68 4 2
69 40% 1
70 42 1
71 45 1
72 45 3
73 4日 1
74 4月 2
75 5 3
76 5196亿 1
77 52579.1 1
78 5点 1
79 6.4万 1
80 60日 1
81 65 1
82 6日 2
83 75.13 1
84 76 1
85 7分 1
86 7日 1
87 7月 2
88 7点 2
89 7点43分 1
90 8 1
91 850 1
92 9 5
93 9月 1
94 : 5
95 ; 2
96 ? 2
97 APP 1
98 X 3
99 XQ2505 2
100 cm 2
101 <http://3g.163.com/news/17/0330/17/CGPSDTS000187VE.html> 1
102 <http://3g.163.com/news/17/0330/17/CGPSE2700018750.html> 1
103 <http://3g.163.com/news/17/0330/17/CGPTKEHI000187VE.html> 1
104 <http://3g.163.com/news/17/0330/17/CGPU6S0700018990.html> 1
105 <http://3g.163.com/news/17/0330/18/CGPV7ERR00018750.html> 1
106 <http://3g.163.com/news/17/0330/18/CGPVL90Q00018750.html> 1
107 <http://3g.163.com/news/17/0330/18/CGPV052P0001875M.html> 1
108 <http://3g.163.com/news/17/0330/18/CGPVTEIT00018AOR.html> 1
109 <http://3g.163.com/news/17/0330/18/CGQ0HNPF00018AOR.html> 1
110 <http://3g.163.com/news/17/0330/19/CGQ2UVML000187VE.html> 1

111 <http://3g.163.com/news/17/0330/19/CGQ3NSRI00018AQQ.html> 1
112 <http://3g.163.com/news/17/0330/19/CGQ3U5F00001899N.html> 1
113 <http://3g.163.com/news/17/0330/19/CGQ501NF00018AOR.html> 1
114 <http://3g.163.com/news/17/0330/19/CGQ5E9UR00018990.html> 1
115 <http://3g.163.com/news/17/0330/20/CGQ6BIKG0001875P.html> 1
116 <http://3g.163.com/news/17/0330/20/CGQ740VN00018990.html> 1
117 <http://3g.163.com/news/17/0330/20/CGQ860RH0001875M.html> 1
118 www.thepaper.cn 1
119 ‘ 1
120 ’ 1
121 “ 39
122 ” 39
123 、 52
124 。 181
125 。 。 。 1
126 《 2
127 》 2
128 【 1
129 】 1
130 (1
131) 1
132 一 25
133 一个 6
134 一些 2
135 一大早 1
136 一样 3
137 一直 1
138 一经 1
139 一起 3
140 一跃而起 1
141 一路 1
142 一边 1
143 一道 1
144 七 1
145 万 4
146 万一 1
147 丈夫 1
148 丈母娘 1
149 三 5
150 上 11
151 上午 5
152 上述 1
153 下 2
154 下午 2
155 下台 1
156 下同 4
157 下属 2
158 下来 1
159 下达 1
160 不 20
161 不予 1
162 不仅 2
163 不公 1
164 不动产 4

165 不同 1

166	不少	1
167	不幸	1
168	不得	3
169	不得了	1
170	不断	1
171	不满	2
172	不见	1
173	不过	2
174	与	8
175	与否	1
176	丑闻	1
177	专业	3
178	专家	1
179	专案组	1
180	专访	1
181	专门	2
182	且	1
183	东南亚	1
184	东经	1
185	东莞	2
186	东莞市	5
187	两	15
188	严厉	1
189	严格	2
190	严肃	1
191	严重	2
192	丧葬	1
193	...	
194	...	
195	...	
196	...	
197	采用	1
198	采访	2
199	释放	1
200	里	1
201	重	1
202	重大	4
203	重庆	2
204	重要	1
205	重视	1
206	金属	2
207	金秋	1
208	金章洙	2
209	金融	3
210	金额	1
211	鉴于	1
212	针对	4
213	钱	1
214	铁岭	1
215	铁岭市	4
216	铁路	2
217	银行业	2
218	销售	2
219	销售商	1
220	销毁	1

221	锄头	1
222	锅	1
223	长	1
224	长安街	1
225	问	3
226	问话	2
227	问询	3
228	问题	5
229	间	1
230	防控	1
231	陆慷	4
232	陆续	1
233	限制	3
234	限购	4
235	除	2
236	随即	1
237	随后	2
238	随着	1
239	随身	1
240	隐匿	1
241	隐隐	1
242	雅	2
243	雅万	3
244	雅万高铁	1
245	雇员	1
246	需	2
247	需要	6
248	震惊	1
249	震源	2
250	青瓦台	2
251	非常	2
252	非本市	2
253	靠	2
254	面	1
255	面向	1
256	面部	1
257	韩	6
258	韩国	22
259	韩媒	1
260	韩方	1
261	项	1
262	项目	8
263	顺差	1
264	须	5
265	预计	1
266	领域	1
267	领导	4
268	领导人	1
269	领导人员	1
270	风险	1
271	首	4
272	首尔	3
273	马	4
274	马方	2
275	马来西亚	8

276	驻华	2
277	骗	1
278	骗局	1
279	高	2
280	高于	1
281	高层	1
282	高强	2
283	高法	1
284	高铁	8
285	高院	4
286	鳌头人	1
287	鳌头镇	2
288	麻果	1
289	黄英	3
290	黑	1
291	黑人	1
292	！	6
293	！！	1
294	（	22
295	）	22
296	，	397
297	：	21
298	；	7
299	？	2