1 Boilerpipe

The question 1 of the assignment is to run boilerpipe on 10,000 files downloaded using wget. Boilerpipe removes all the html templates from the downloaded files.

I downloaded Boilerpipe and imported the threee .jar files to my classpath.I then used the boilerpipe.java program to remove the html templates. I did this on command prompt and it took me a lot of time to run just about 500 URIs. I then used net beans to run the 10 separate boilerpipe.java programs, each program with different inputs and it saved me a lot of time.

The file size of 9500 URIs downloaded using wget is approx.900MB. The file size of 9500 URIs after using boilerpipe is approx.16MB. Total words after successful boilerpipe removal is 1,535,723. I estimated the total words before boilerpipe removal to be approximately equal to 86,384,458 words based on the file size. Boilerpipe was successful for about 6101 URIs out of the total 9000 URIs I used. The remaining URIs threw exceptions and below are some of the frequent exceptions received:

Ex 1:de.l3s.boilerpipe.BoilerpipeProcessingException: java.net.SocketException: Unexpected end of file from server http://www.nbcmiami.com/news/local/Plane-Crashes-in-SW-Miami-Dade-County-291559351.html I received the above error for the above URI. I received such errors a long time after the servers were contacted. These exceptions were the main reason the boilerpipe.java took a lot of time to run.[2]

Ex 2:de.l3s.boilerpipe.BoilerpipeProcessingException: java.net.UnknownHostException: merumo.info http://merumo.info/15pp37/

Ex 3:BoilerpipeProcessingException: java.io.FileNotFoundException: http://err.lolipop.jp/404.html http://truffle.upper.jp/ahv21

Ex 4:de.l3s.boilerpipe.BoilerpipeProcessingException: java.io.IOException: Server returned HTTP response code: 403

Ex 5:de.l3s.boilerpipe.BoilerpipeProcessingException: java.io.IOException: Server returned HTTP response code: 503 for URL:http://topic.xvs.jp/bbp

Ex 6: http://item.rakuten.co.jp/spatecno/10000567/? For websites that ended with '.jp' the boiler pipe was unsuccessful as well.

Ex7: I found that for most of the news websites the boiler pipe was unsuccessful.

An example of a successful boilerpipe removal http://www.missfoodwise.com/2015/02/food-`photography-magic-soup-book.html?utm_content=bufferd528d&utm_medium=social&utm_source=twitter.com&utm_campaign=buffer`

2 Extracting unique terms

To extract unique terms and their frequencies I wrote a java program with a mapper and a reducer and ran it in a Hadoop environment. The mapper generates a key and value pair for every word in the input file. The reducer then generates total frequency for every unique word.[1]

To run this java program in hadoop the input file should be kept in HDFS and the output generated is also in HDFS. The output should be copied from HDFS into the local machine.

Out of the top 50 unique terms generated, after boilerpipe removal, about 10-11 terms are found in the stop words list and for the top 50 terms before boiler pipe removal none of those are found in the stop words list.

I've used the WordCount.java program to extract the unique words. I've attached directions to run the WordCount.java file as well.

I used R to plot the graph for the above two tables. I have two R scripts. Run each script to view graph for before boiler pipe removal and the other for after boiler pipe removal.

Zipf distribution:

Zipf's law states that given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table. Thus the most frequent word will occur approximately twice as often as the second most frequent word, three times as often as the third most frequent word, etc.

From the frequencies i received Table 2.1 doesn't follow zipf distribution. The first term should occur twice as much as the second and thrice as much as the third for it to follow zipf distribution but the firs term occurs approximately 1.5 times the second and 1.75 times the third. This clearly shows that it doesn't follow zipf distribution.

Table 2.2 doesn't follow zipf distribution either. First term occurs 1.1 times the second and 1.75 times the third and this shows that it doesn't follow zipf distribution.

References

[1] Dennis Dawson. Hadoop mapreduce tutorial. 2014.

[2] Christian Kohlschtter. boilerpipe - boilerplate removal and fulltext extraction from html pages. .

Table 1: Top 50 unique terms before boiler pipe

Term	Frequency
-	3503702
=”	2126021
.	2003431
a	1875991
/	1380493
¿	1254712
class	1231829
p	1115596
div	1025258
i	939684
d	908005
l	878443
e	872345
b	871899
m	853392
c	839751
j	829309
h	819502
k	805402
0	799214
f	761501
g	754305
n	744046
r	733237
1	681085
u	636432
s	601923
t	600116
:	599755
2	581574
q	574632
,	564748
¡/	560449
4	556199
o	546042
3	509090
8	507972
¡	505992
;	500797
5	500737
=	472302
9	462552
6	433063
span	425628
7	399257
com	375718
li	361690
href	343715
://	328463

Table 2: Top 50 unique terms after boiler pipe

.	58212
,	53538
the	35043
a	25557
to	20563
-	17890
and	17345
of	16143
:	14641
i	13208
in	12191
/	12077
s	12025
’	11910
de	10640
p	10421
y	9988
(	9713
t	9045
e	8973
b	8844
?	8807
o	8636
)	8567
you	8564
!	8066
w	8055
u	8010
d	8008
q	7953
is	7883
j	7842
c	7659
play	7491
n	7488
]	7487
h	7361
x	7356
z	7317
k	7228
for	7182
f	7129
v	7013
l	7003
on	6790
this	6637
[	6451
1	6420
that	6018
it	5834