Assignment 3 : CS 751

HARISH

April 3, 2015

0.1 Part 1

For the text you saved for the 10000 URIs from A1, Q2: Use the boilerpipe software to remove the HTML templates from all HTML pages (document how many pages link from the tweets were non-HTML and had to be skipped) https://code.google.com/p/boilerpipe/ WSDM 2010 paper: http://www.l3s.de/ kohlschuetter/boilerplate/ For how many of the 10000 URIs was boilerpipe successful? Compare the total words, unique words, and byte sizes before and after use of boilerpipe For what classes of pages was it successful? For what classes of pages was it unsuccessful? Provide examples of both successful and unsuccessful removals and discuss at length.

I have used jusText library to remove HTML templates. It removes headers, footers and navigation links from the page. I have written a code in python named "htmlnonhtml.py" which removes boilerplate content from the links.

Total words before removing boiler content plate is 64221898 and after removing boiler content plate is 91666 . Total unique words before removing boiler content plate is 2979715. Total unique words after removing boiler content is "19418".

Only 6100 links were unique , out of this 1496 are successful . Most of them are non-HTML skipped because of non content , Images on links , 404, 302 errors etc.

The size of the file before removing boiler content is "573,080 KB" and after removing boiler content is "553 KB".

jusText was unsuccessful for pages that did not follow HTML standards. Pages which are not properly organized .

Below figure1 represents unsuccessful class of page.

Unsuccessful Class of Page

jusText was successful for pages that followed HTML standards. Pages which are properly organized .

Below figure2 represents successful class of Page.

successful Class of Page

Collection1: Extract all the unique terms and their frequency from the 10000 files* Collection2: Extract all the unique terms and their frequency of the 10000 files* after running boilerpipe Construct a table with the top 50 terms from each collection. Find a common stop word list. How many of the 50 terms are on that stop word list? For both collections, construct a graph with the x-axis as word rank, and y-axis as word frequency. Do either follow a Zipf distribution? Support your answer. I have extracted extracted all the unique terms and their frequency from the 10000 files. This is done using "getwordcount.py" . All the words before applying boliler pipe are stored in "wordcountbeforeboiler.txt".

Also , i have extracted all the unique terms and their frequency and stored in "wordcountafterboiler.txt".

I have also extracted top 50 words with their frequency into "fiftywordcountbeforeboiler.txt" before applying boiler pipe .This file "fiftywordcountafterboiler.txt" contains top 50 words with their frequency after applying boilerpipe.

Below figure 3 has top 50 words with their frequencies before applying boiler pipe.

Below figure 3 has top 50 words with their frequencies after applying boiler pipe.

41 words are found common from text file from the stop word list which I got from internet.

14 words are found common from words extracted from html file .

I have plotted the top 50 words frequency distribution using RStudio for the words which i have extracted brfore applying boiler plate and after applying boiler plate.
The script "frequencyrankComparsion.R" exports the frequency distribution graph for words after applying boiler plate.It is shown in Figure 3.

The script "htmlData.R" exports the frequency distribution graph for words before applying boiler plate.It is shown in Figure 4.

Figure3,4 doesn't follow zipf distribution. Most frequent word that occurred did not approximately equal to twice as often as the second most frequent word ,three times as often as the third most frequent word and so on.

Table 1: Rank, word and frequency using jusText

RANK	WORD	FREQUENCY
1	the	4444
2	and	3037
3	to	2753
4	of	2015
5	a	1862
6	in	1653
7	you	1330
8	is	1037
9	for	968
10	are	847
11	that	764
12	on	754
13	with	699
14	this	693
15	it	569
16	or	562
17	i	555
18	your	554
19	be	534
20	by	525
21	if	504
22	from	495
23	as	494
24	have	459
25	we	451
26	will	413
27	at	384
28	not	382
29	our	354
30	an	342
31	more	325
32	new	319
33	can	314
34	all	302
35	was	282
36	but	260
37	he	232
38	their	219
39	about	213
40	my	211
41	they	207
42	has	194

Table 2: Top 50 terms extracted from HTML files

RANK	TERM	FREQUENCY
1	div	5921
2	=	3933
3	the	3711
4	¡a	3511
5	to	3408
6	and	3390
7	a	3270
8	of	3110
9	{	3032
10	<a	2913
11	<span	2878
12		2773
13	in	2683
14	{.	2583
15	for	2480
16	-	2310
17	<li	2237
18	at	2137
19	var	2031
20	point:false,	1997
21	on	1896
22	+	1793
23		1633
24	is	1489
25	target='self'	1211
26	?	1031
27	position:'left'},	937
28	yous	820
29	your	708
30		681
31	class="user-in"	608
32	with	585
33	I	565
34	user	550
35	target="	520
36	&[6]	482
37	2014	472
38		460
39	December	452
40	0	400
41	rel='fll'	368
42	<img	355

Frequency Distribution of top 50 words before boiler plate

Frequency Distribution of top 50 words after boiler plate

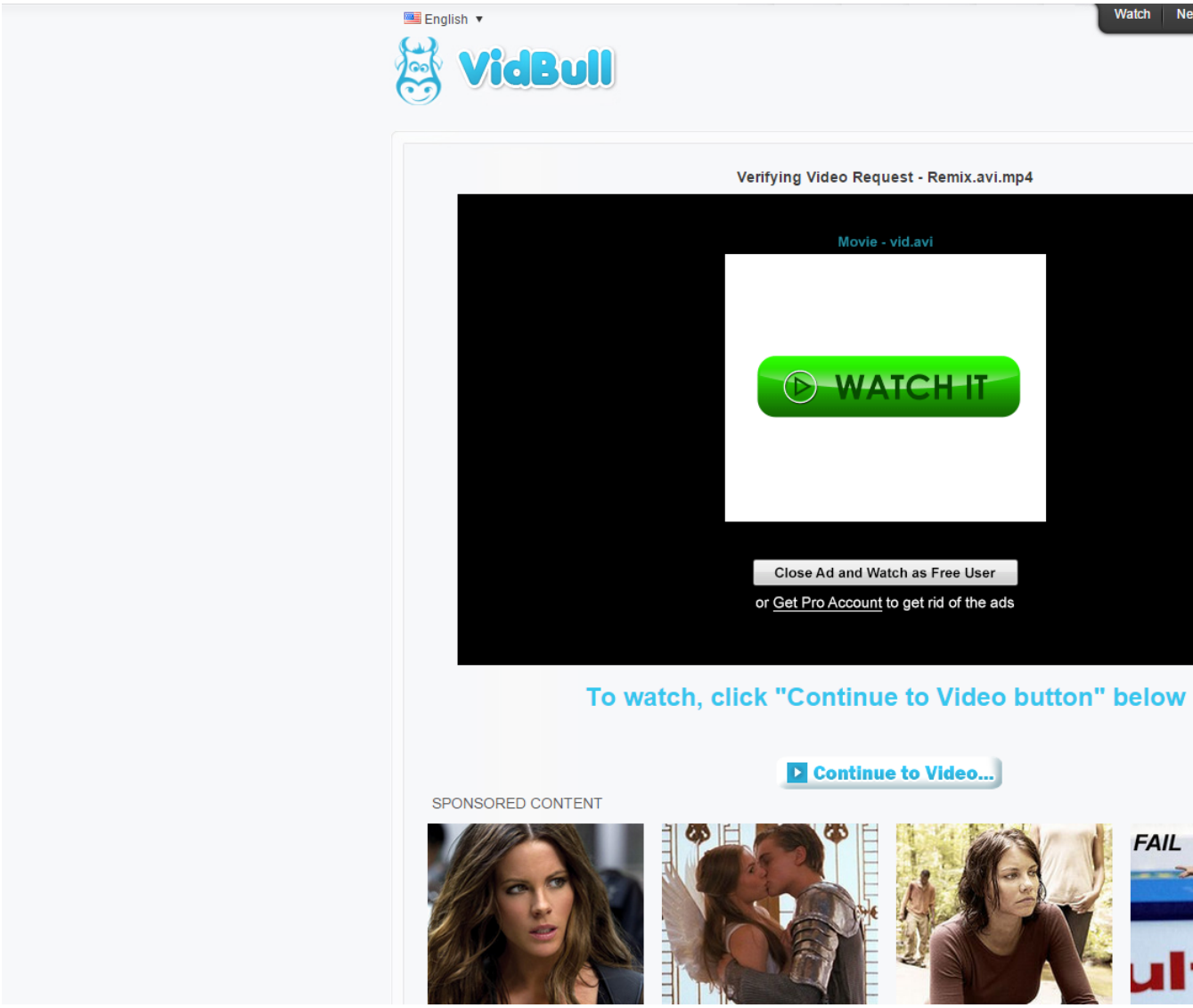

Figure 1: Unsuccessful Class of Page

Figure 2: successful Class of Page

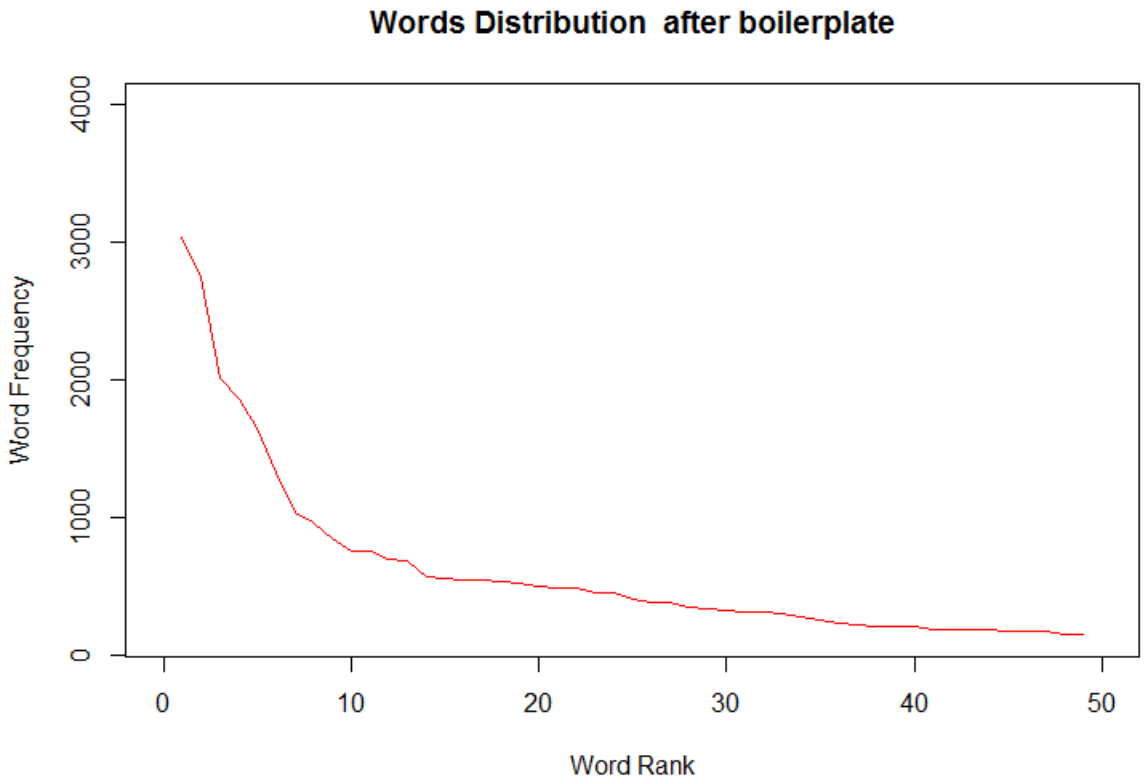
Words Distribution after boilerplate
Word Frequency
0
1000
2000
3000
4000
0
10
20
30
40
50
Word Rank

Figure 3:

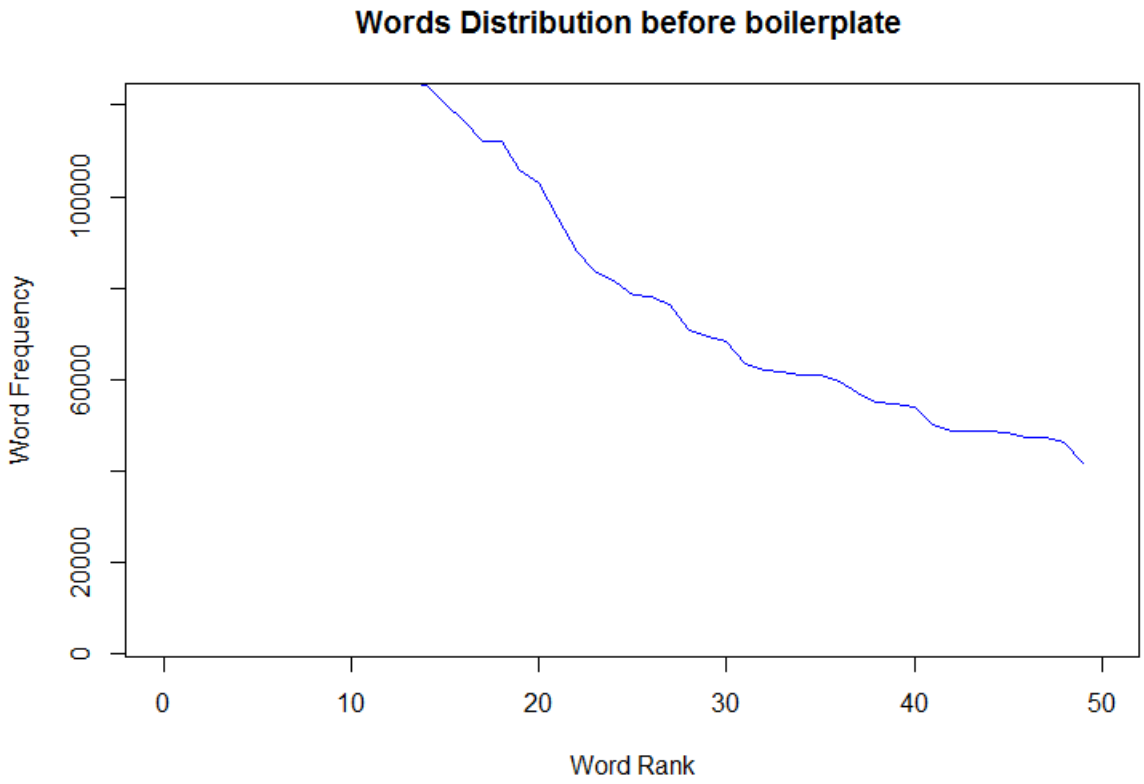
Words Distribution before boilerplate
Word Frequency
0
20000
60000
100000
0
10
20
30
40
50
Word Rank

Figure 4: