Web Science cs532: Assignment #3

Due on Thursday, February 18, 2016

Dr.Michael.L.Nelson 4:20pm

Zetan Li

Contents

Problem 1

Download the 1000 URIs from assignment #2. "curl","wget", or "lynx" are all good candidate programs to use. We want just the raw HTML, not the images, stylesheets, etc.

from the command line:

% curl `http://www.cnn.com/>www.cnn.com`

% wget -O `www.cnn.com` `http://www.cnn.com/`

% lynx -source `http://www.cnn.com/>www.cnn.com`

"www.cnn.com" is just an example output file name, keep in mind that the shell will not like some of the characters that can occur in URIs (e.g., "?", "&"). You might want to hash the URIs, like:

% echo -n "`http://www.cs.odu.edu/show\_features.shtml?72`" | md5 41d5f125d13b4bb554e6e31b6b591eeb

("md5sum" on some machines; note the "-n" in echo – this removes the trailing newline.)

Now use a tool to remove (most) of the HTML markup. "lynx" will do a fair job:

% lynx -dump -force_html `www.cnn.com>www.cnn.com.processed`

Use another (better) tool if you know of one. Keep both files for each URI (i.e., raw HTML and processed).

SOLUTION

Copy the url list from assignment 2 and run shell script blow to download web page from every url in the list.

Listing 1: Shell script for web downloading

```
#!/bin/bash
dataDir="../webpages/"
rm -f hashtable
touch hashtable
counter=0
for line in `cat url_list.txt`; do
     fname=`echo -n $line | openssl md5 | cut -d" " -f2`
     curl  --silent -A "Mozilla/5.0 (X11; Linux i686 (x86_64); rv:2.0b4pre) Gecko
        /20100812 Minefield/4.0b4pre" "$line" > ${dataDir}${fname}
     echo -e $fname"\t"$line >>hashtable
     counter=$[$counter+1]
     percent=$[$counter/10]
     echo -e -n "\r"$percent"%"
done
```

The use lynx in description above to remove all the html tags.

Listing 2: Shell script to convert raw web page

```
#!/bin/bash
dataDir="../webpage_processed/"
raw_dir="../webpages/"
counter=0
for line in `cat url_list.txt`; do
     fname=`echo -n $line | openssl md5 | cut -d" " -f2`
     lynx  -dump -force_html ${raw_dir}${fname} > ${dataDir}${fname}
     counter=$[$counter+1]
     percent=$[$counter/10]
     echo -e -n "\r"$percent"%"
done
```

Problem 2

Choose a query term (e.g., “shadow”) that is not a stop word (see week 5 slides) and not HTML markup from step 1 (e.g., “http”) that matches at least 10 documents (hint: use “grep” on the processed files). If the term is present in more than 10 documents, choose any 10 from your list. (If you do not end up with a list of 10 URIs, you’ve done something wrong).

As per the example in the week 5 slides, compute TFIDF values for the term in each of the 10 documents and create a table with the TF, IDF, and TFIDF values, as well as the corresponding URIs. The URIs will be ranked in decreasing order by TFIDF values. For example:

Table 1. 10 Hits for the term “shadow”, ranked by TFIDF.

```
TFIDF TF IDF URI
----- -- --- ---
0.150 0.014 10.680 http://foo.com/
0.044 0.008 5.510 http://bar.com/
```

You can use Google or Bing for the DF estimation. To count the number of words in the processed document (i.e., the deonminator for TF), you can use “wc”:

```
% wc -w www.cnn.com.processed
2370 www.cnn.com.processed
```

It won’t be completely accurate, but it will be probably be consistently inaccurate across all files. You can use more accurate methods if you’d like.

Don’t forget the log base 2 for IDF, and mind your significant digits!

SOLUTION

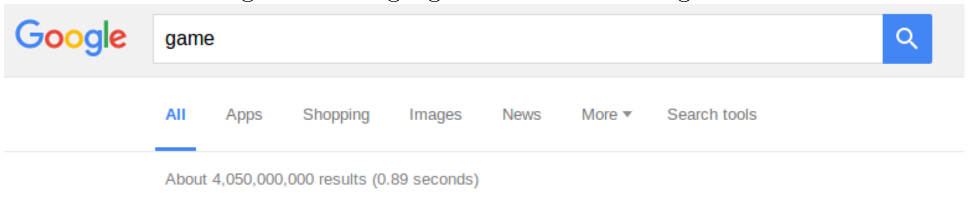
To get IDF, we should search on the google to get the total amount of file contain the term we required. Here, we choose the term “game”.

Note: The data we use is from Feb 16, and the image we captured is in Feb 17, You can see the result increased.

Then search hard for total amount of web site indexed in google, and get the answer[1]

Problem 2 continued on next page. . .

Figure 1: The google search result for “game”



About 4,050,000,000 results (0.89 seconds)

So now we have DF and total number of documents to calculate IDF.
Next, grep each file in the folder count the term using wc, and calculate the normalized TF.
During calculating, we use bc to enable float calculation in bash, and awk to do the log2 work. In order to get 10 different domain for problem 3, we listed 20 urls.
Finally multiply them together to get TF-IDF, then use sort to get ranked list.

Listing 3: Code to get TF and IDF

```
#!/bin/bash
dataDir="../webpage_processed/"
fileCount=0
rm -f TFdata
touch TFdata
amount=`grep -rl "$1" $dataDir| wc -l`
IDF=$(echo "scale=6;30000000000000/3970000000"|bc)
IDF=`echo $IDF | awk '{printf("%.6f",log($1)/log(2))}'`
for file in `ls $dataDir`; do
     count=`cat ${dataDir}${file} | grep -o "$1" | wc -l`
     if [[ $count -gt 0 ]]; then
          words=`cat ${dataDir}${file} | wc -w`
          TF=$(echo "scale=6;$count/$words"|bc)
          TFIDF=`echo "scale=6;$TF*$IDF"| bc`
          url=`grep "$file" hashtable | cut -f2`
          echo -e $TFIDF"\t"$TF"\t"$IDF"\t"$url >> TFdata
          fileCount=$[$fileCount+1]
          if [[ $fileCount -gt 20 ]]; then
               break
          fi
     fi
done
sort -r -n -k1 TFdata > TFRankdata
rm TFdata
```

Problem 2 continued on next page. . .

Table 1: The table of TF value of top ten url

TF-IDF	TF	IDF	URI
8.589015	0.666666	12.883536	http://mmanor.17bullets.com/twitter_post.php?messageId=quest &values=quests.2584.title&tw_locale=en_US%0A
1.004542	0.077971	12.883536	http://games-fors.ru/top/art/edge-of-tomorrow-game/%0A
0.460122	0.035714	12.883536	http://www.freem.ne.jp/win/game/9914%0A
0.080303	0.006233	12.883536	http://www.vegascoverage.com/2016/02/08/quinnipiac-vs-saint -peters-ncaa-sports-betting-odds-pick-and-prediction/?utm_campaign= Vegas+Coverage&utm_source=twitterfeed&utm_medium=twitter
0.068527	0.005319	12.883536	https://www.facebook.com/jaycecabell1/posts/1645995465662075
0.067728	0.005257	12.883536	http://www.ebay.com/itm/like/331772926586?item=331772926586 &lgeo=1&utm_medium=twitter&utm_source=dlvr.it&vectorid=229466&rmvSB=true
0.059637	0.004629	12.883536	http://d-comic.net/movie_view.php?id=35529
0.059019	0.004581	12.883536	https://www.youtube.com/watch?v=rNwWpIRX7Pw&feature=youtu.be&aBEST%0A
.052152	.004048	12.883536	http://endzoneblog.com/denvers-defense-overwhelms-carolina-super-bowl-50/
0.048068	0.003731	12.883536	http://PETERSUN19.rsscb.com/asapin/datingtothebank/02081606/ the-love-game-how-to-date-the-most-beautiful-women-you- ever-wanted.php?utm_source=dlvr.it&utm_medium=twitter

Problem 3

Now rank the same 10 URIs from question #2, but this time by their PageRank. Use any of the free PR estimaters on the web, such as:

```
http://www.prchecker.info/check_page_rank.php
http://www.seocentro.com/tools/search-engines/pagerank.html
http://www.checkpagerank.net/
```

If you use these tools, you'll have to do so by hand (they have anti-bot captchas), but there is only 10. Normalize the values they give you to be from 0 to 1.0. Use the same tool on all 10 (again, consistency is more important than accuracy).

Create a table similar to Table 1:

Table 2. 10 hits for the term "shadow", ranked by PageRank.

PageRank URI
-------- ---
0.9 http://bar.com/
0.5 http://foo.com/

Briefly compare and contrast the rankings produced in questions 2 and 3.

SOLUTION

We choose `http://www.seocentro.com/tools/search-engines/pagerank.html` for the tool for this problem. As all the user-specific page will result in "undefined" from search, we use the domain as PR search term instead (not accurate but this is one way to solve the undef problem).

But still, some of the domain result in a n/a answer.

Table 2: Page rank from website estimation

URI	Page Rank
http://mmanor.17bullets.com	N/A
http://games-fors.ru	N/A
http://www.freem.ne.jp	5/10
http://www.vegascoverage.com	N/A
https://www.facebook.com	9/10
http://www.ebay.com	8/10
http://d-comic.net	1/10
https://www.youtube.com	9/10
http://endzoneblog.com	N/A
http://PETERSUN19.rsscb.com	N/A

Table 3: Normalized page rank

URI	Page Rank
http://mmanor.17bullets.com	0
http://games-fors.ru	0
http://www.freem.ne.jp	0.5
http://www.vegascoverage.com	0
https://www.facebook.com	0.9
http://www.ebay.com	0.8
http://d-comic.net	0.1
https://www.youtube.com	0.9
http://endzoneblog.com	0
http://PETERSUN19.rsscb.com	0

Compare
Because we use the domain name for page rank, so the accuracy is not so good. A domain may be popular but certain user page may not well-known. Additionally, as we can see, the facebook and youtube has high page rank, just because many websites have share link point to them, but if we point to certain video or post in their site, like the page contain "game" we picked from our own web database, they do not have so much link refer to them, or even none, because our web tool cannot even find them, but they are more relevant to our search term(e.g the top 1 ranked web site in our selection, though with zero page rank, but it is really a game site).

References

[1] Statistic brain. *Total Number of Pages Indexed by Google*, 2014 (accessed Febrary 16, 2016).