HW 3 - Ranking Webpages

Adeniran Adeniyi

Sunday, March 7th 2021 by 11:59pm

Q1

For the following tasks, consider which items could be scripted, either with a shell script or with Python. You may even want to create separate scripts for different tasks. It's up to you to determine the best way to collect the data.

- curl, wget, or lynx are all good candidate programs to use. We want just the raw HTML, not the images, stylesheets, etc.

Download the content of the 1000 unique URIs you gathered in HW2. Plan ahead because this will take time to complete.

You'll need to save the content returned from each URI in a uniquely-named file. The easiest thing is to use the URI itself as the filename, but it's likely that your shell will not like some of the characters that can occur in URIs (e.g., "?", "&"). You might want to hash the URIs to associate them with their respective filename.

For example, https://www.cnn.com/world/live-news/nasa-mars-rover-landing-02-18-21 hashes to 2fc5f9f05c7a69c6d658eb680c7fa6ee:

```
% echo -n "https://www.cnn.com/world/live-news/nasa-mars-rover-landing
   -02-18-21" | md5
2fc5f9f05c7a69c6d658eb680c7fa6ee
```

(md5 might be md5sum on some machines; note the -n in echo – this removes the trailing newline.)

Using this as an example, here are some ways you could download the HTML from that URI using the hash as the filename:

- ```
  % curl "https://www.cnn.com/world/live-news/nasa-mars-rover-
  landing-02-18-21" > 2fc5f9f05c7a69c6d658eb680c7fa6ee.html
  ```

  ```
  % wget -O 2fc5f9f05c7a69c6d658eb680c7fa6ee.html https://www.cnn
  .com/world/live-news/nasa-mars-rover-landing-02-18-21
  ```

- ```
  % lynx -source https://www.cnn.com/world/live-news/nasa-mars-
  rover-landing-02-18-21 > 2fc5f9f05c7a69c6d658eb680c7fa6ee.html
  ```

Now use a tool to remove (most) of the HTML markup for all 1000 HTML documents. python-boilerpipe will do a fair job, see
http://ws-dl.blogspot.com/2017/03/2017-03-20-survey-of-5-boilerplate.html *:*

```
    from boilerpipe.extract import Extractor
    extractor = Extractor(extractor='ArticleExtractor', html=html)
    extractor.getText()
```

Keep both files for each URI (i.e., raw HTML and processed). Upload both sets of files to your GitHub repo.

Answer

```
$Textfie = "C:\Users\adeni\CS532\Week5\hw2-archiving-aaden001\Q1\dict.txt"
$arrayUrlMd5 = @()
foreach($line in [System.IO.File]::ReadLines($Textfie)){
  #Build the output file for each URL
  $md5 = New-Object -TypeName System.Security.Cryptography.MD5CryptoServiceProvider
  $utf8 = New-Object -TypeName System.Text.UTF8Encoding
    #hash the each URLs in the dict.txt
  $hash = [System.BitConverter]::ToString($md5.ComputeHash($utf8.GetBytes($line)))
  $hash = $hash.Replace('-','') #has md5hash values

    #User for question 1
  docker container run -it --rm curlimages/curl -L  $line  > C:\Users\adeni\CS532\Week8\hw-ranking-aaden001\Q1\html\$hash.html
  #user for Question 2
    #Generate csv file for url hash
    $hash = "$hash.txt"
    $arrayUrlMd5 += New-Object pscustomobject -Property @{'URL'=$line; 'hash' = $hash}

    $count++

}
#save object to csv file
$arrayUrlMd5 | Export-Csv -Append -Path C:\Users\adeni\CS532\Week8\hw-ranking-aaden001\Q2\urlHash.csv -NoTypeInformation
```

Listing 1: dataCollections.ps1

```
# -*- coding: utf-8 -*-
"""
Created on Mon Mar  1 10:05:54 2021
```

```
4
5 @author: adeni
6 """
7
8 from boilerpy3 import extractors
9 import re
10 import os
11
12 pattern = re.compile(r"Q1\/html\/([\w]*)[.html]")
13 count = 0
14 directory ="Q1/html/"
15 for filename in os.listdir(directory):
16     if filename.endswith(".html"):
17
18         try:
19             f =open(os.path.join(directory, filename),"r")
20             #where my problem all began
21             #reconfigure file encoding to 16
22             f.reconfigure(encoding="utf-16")
23             text = f.read()
24             f.close()
25             #Find a match and group in regex compile
26             tempName = str(os.path.join(directory, filename))
27             m = pattern.match(tempName)
28             #change file extension by buildng the string
29             textFile_path = "Q1/processed/" + str(m.group(1)) +".txt"
30
31             extractor = extractors.ArticleExtractor()
32             content=extractor.get_content(text)
33             #print(content)
34             p = open(textFile_path,"a")
35             p.write(content)
36             p.close()
37         except Exception as r:
38             print(r)
```

Listing 2: boiler-plate-removal.py

Discussion

I followed these instruction below:

- Download the content of the 1000 unique URIs you gathered in HW2. Plan ahead because this will take time to complete.

Answer

- I used power-shell scripting doing this task. I used a docker image of curl to use the curl command on windows power-shell to in line 12 of dataCollection.ps1. The curl option L to follow redirects and the variable $line has the part to the location of dict.txt which has the list of a 1000 url. This command gets only the html document from the URI-R, it also creates these html files in Q1\html folder.

```
12  docker container run -it --rm curlimages/curl -L  $line  > C:\
     Users\adeni\CS532\Week8\hw-ranking-aaden001\Q1\html\$hash.html
```

Listing 3: snapshot of dataCollections.ps1

- You'll need to save the content returned from each URI in a uniquely-named file. The easiest thing is to use the URI itself as the filename, but it's likely that your shell will not like some of the characters that can occur in URIs (e.g., "?", "&"). You might want to hash the URIs to associate them with their respective filename.
  For example, https://www.cnn.com/world/live-news/nasa-mars-rover-landing-02-18-21 hashes to 2fc5f9f05c7a69c6d658eb680c7fa6ee:

Answer

- I converted the url gotten from the dict.txt to an MD5 hash in line 5 to 9

```
5  $md5 = New-Object -TypeName System.Security.Cryptography.
    MD5CryptoServiceProvider
6  $utf8 = New-Object -TypeName System.Text.UTF8Encoding
7    #hash the each URLs in the dict.txt
8  $hash = [System.BitConverter]::ToString($md5.ComputeHash($utf8.
    GetBytes($line)))
9  $hash = $hash.Replace('-','') #has md5hash values
```

Listing 4: snapshot of dataCollections.ps1

- *Now use a tool to remove (most) of the HTML markup for all 1000 HTML documents. python-boilerpipe will do a fair job, see hrefhttp://ws-dl.blogspot.com/2017/03/2017-03-20-survey-of-5-boilerplate.htmlhttp://ws-dl.blogspot.com/2017/03/2017-03-20-survey-of-5-boilerplate.html*

Answer

- I used boilerpy3 which is a better version of boilerpipe to do the extraction of the html. I grabbed all the html document files in \Q1 \html extracted them with boilerpy3 and deposited the result in \Q1 \processed folder.

  One major issues I had encountered was UTF encoding problems. This caused the boilerpy3 not work on the html string I feed it. So i noticed the output of the processed html files to text file automatically came out from power-shell as UTF UTF8BOM when using the file in dataCollections.ps1 In other for the extraction tool to do its job,

I had to reconfigure the text files gotten from processed folder to UTF-16 encoding . In line 22 of boiler-plate-removal.py

```
22                f.reconfigure(encoding="utf-16")
```

Listing 5: snap shot of boiler-plate-removal.py

- I used the getDistinctDomain.py to clean the final.csv file to produce the actualFinal.csv result.Only ten list was printed to the csv file, this had 10 distinct URL domains

Q2

Answer

```
#Set the directoryPath
$directoryPath = "C:\Users\adeni\CS532\Week8\hw-ranking-aaden001\Q1\processed\"
#Save the file names in an array
$files = Get-ChildItem -Name -Path "$directoryPath"

$pattern = "coronavirus"
#to track the number of documents that had the term coronavirus
$stopDocument = 0

#declare an array of Object
$testObject = @()

#final path to Export-Csv
$finalPath = "C:\Users\adeni\CS532\Week8\hw-ranking-aaden001\Q2\final.csv"
#run a for loop
foreach ($filenames in $files){
    #concatenate the full file path
    $directoryfilenames = "$directoryPath$filenames"
    #get the number of times, the term coronavirus is found in the document
    $termFrequency= (Get-Content $directoryfilenames| Select-String -pattern "$pattern").length
    #Get the total word per document
    $t = Get-Content $directoryfilenames| Measure-Object -word | Select -Object -expandproperty Words


    #avoid dividing by zero
    #Get TF value frequency/ total word count in document
    if($t -eq 0){
```

```
        $tf = 0
    }else {
        $tf = ($termFrequency/$t)
    }

    #Get-Content $directoryfilenames| Measure-Object  Word |Select-Object -expand Words
    #get only document that has a frequency of at least 1
    #Answering Q4 when the frequency of  the word in document is at least one then we can count it once
    if($termFrequency -gt 1){
        $stopDocument++

        $testObject += New-Object pscustomobject -Property @{'Frequency' = $termFrequency;'TotalWord' =$t; 'TF'= $tf;'IDF' = "";'Name'= $filenames; 'URL'=""}
    }
}
$filePath = "C:\Users\adeni\CS532\Week8\hw-ranking-aaden001\Q2\urlHash.csv"
#get Generated url hash csv
#and make a hashtable
$mytable = Import-CSV -Path $filePath
$headers = $mytable[0].psobject.properties.name
$key = $headers[0]
$value = $headers[1]
$hashTable = @{}
$mytable | %{$hashTable[$_."$key"] = $_."$value"}


#use the look up table to link to the corresponding Url to test object
$testObject | ForEach-Object {
    $_.URL = $hashTable[$_.Name]
}| Set Object
<# Question 4 #>
echo "This is the total document with at least One occurence of the term coronavirus"
echo $stopDocument
#export finalresorting outputs
$testObject | Export-Csv -Append -Path C:\Users\adeni\CS532\Week8\hw-ranking-aaden001\Q2\final.csv -NoTypeInformation
```

Listing 6: TF.ps1

```
# -*- coding: utf-8 -*-
"""
Created on Sun Mar  7 00:36:05 2021

```

```
@author: adeni
"""
import pandas as pd
from urllib.parse import urlparse # for requesting parse


filePath = "Q2/final.csv"


storeSingleD =[]
myList = []


df = pd.read_csv(filePath,index_col=False, encoding='utf-8')
#convert to list
myList = df.values.tolist()



"""
Get through the URL
get the domain of url domain = urlparse('http://www.example.test/foo/
   bar').netloc
If array doesnt contain the domain
store in a new pandas datast(list becasueit much better)

"""
#skip same domain and take first ten of the list
newList = []
count = 1
for item in myList:
    domain = urlparse(item[4]).netloc

    if (domain in storeSingleD):
        pass
    else:
        try:
            count = count +1
            storeSingleD.append(domain)
            newList.append([item[3],item[2],round(item[0],5),"","",
   item[4]])
        except Exception as r:
            print(r)

    if count > 10:
        break
#convert to pandas dataFrame
```

```
50 dp= pd.DataFrame(newList, columns=['Frequency','Total Word','TF','IDF',
      'TF-IDF','URL'])
51 dp.to_csv("Q2/actualFinal.csv",index=False)
```

Listing 7: getDistinctDomain.py

Discussion

I followed these steps below:

- I choose a query term coronavirus as seen on line 6 of TF.ps1
- I Opened each of the generated text document to search for the term. Power-shell as some easy ways to get the number of times the term appears in a textfile, seen in line 20 of TF.ps1

```
20     $termFrequency= (Get-Content $directoryfilenames| Select-String
       -pattern "$pattern").length
```

Listing 8: snap shot TF.ps1

- I also got the total word count in the document using Measure-Object -word, line 22

```
22     $t = Get-Content $directoryfilenames| Measure-Object -word |
     Select-Object -expandproperty Words
```

Listing 9: snap shot TF.ps1

- I was able to compute the the TF using the formula

  TF(coronavirus)= (total occurrence of the word coronavirus in a document)/(the total word count in the document)

- I avoided dividing by Zero since some of the text files had no content. I set the TF to be Zero when this occurs, see in line 25-31

```
25     #avoid dividing by zero
26     #Get TF value frequency/ total word count in document
27     if($t -eq 0){
28         $tf = 0
29     }else {
30         $tf = ($termFrequency/$t)
31     }
```

Listing 10: snap shot TF.ps1

- using a hash table I created from a urlHash.csv file - i generated, contains filename and corresponding hashed values - I was able to associate matching text file to there URL. The urlHash.csv file is found in \Q2 folder

- The computation for every document with at least one term frequency was created to a csv file called final.csv. This file is located in \Q2 folder
- IDF was calculated using the size of google's corpus of 55B and the search result obtained by using google browser which was 2.03 billion result.IDF = log2((Googles total size of corpus)/(search resutl of term on google browser)).The Calculation gave 4.76. -IDF was gotten by multiplying the result of TF and IDF above.

I manually configured the table below

Table 1: Hits for the term "coronavirus", ranked by TF-IDF

TF	IDF	TF-IDF	URL
0.01538	4.76	0.073	https://www.newscientist.com/article/mg24933233-500-first-universal-coronavirus-vaccine-will-start-human-trials-this-year/
0.00814	4.76	0.039	https://www.cbsnews.com/live-updates/coronavirus-china-outbreak-death-toll-infections-latest-updates-2020-02-17/
0.00592	4.76	0.028	https://www.donaldjtrump.com/media/study-hydroxychloroquine-significantly-lowers-coronavirus-death-rate/
0.00556	4.76	0.026	https://pittsburgh.cbslocal.com/2021/02/24/allegheny-county-3-uk-variants-covid-19/?utm_source=dlvr.it&utm_medium=twitter
0.00535	4.76	0.025	https://thewest.com.au/news/coronavirus/two-patients-given-wrong-dose-of-coronavirus-vaccine-ng-b881805132z?utm_campaign=share-icons&utm_source=twitter&utm_medium=social&tid=1614210946210
0.0049	4.76	0.023	http://veronews.com/2021/02/24/coronavirus-in-irc-feb-24-update/
0.00477	4.76	0.023	https://www.wcvb.com/article/biogen-conference-bostons-first-coronavirus-superspreader-event/35618145
0.0042	4.76	0.02	https://www.manorisd.net/site/default.aspx?PageType=3&DomainID=4&ModuleInstanceID=1600&ViewID=6446EE88-D30C-497E-9316-3F8874B3E108&RenderLoc=0&FlexDataID=10464&PageID=1
0.0029	4.76	0.014	https://www.cnbc.com/2021/02/24/house-democrats-aim-to-pass-1point9-trillion-covid-relief-bill-on-friday.html?utm_source=dlvr.it&utm_medium=twitter
0.00158	4.76	0.008	https://www.politicshome.com/thehouse/article/coronavirus-public-health-local-responsibility

Q3

Answer

Table 2: Page ranking per domain gotten from Question 2

Pageranking	URL
0.7	https://www.cbsnews.com
0.7	https://www.cnbc.com
0.6	https://www.newscientist.com
0.6	https://pittsburgh.cbslocal.com
0.6	https://www.donaldjtrump.com
0.5	https://www.politicshome.com
0.5	http://veronews.com
0.5	https://www.manorisd.net/
0.5	https://www.wcvb.com
0.5	https://thewest.com.au

Discussion

- I used this https://dnschecker.org/pagerank.php to obtain the page ranking for each domain
- Comparing the result of the Table 2 to Table 1, depended of a number of factors
  - I noticed that using the various Page ranking website you gave us generated completely different result to one another.

    dnschecker.org in particular doesn't allow you to view domains that do not use secured http. I notice that some of the Page Ranking site allowed only secured http, for example https://www.prchecker.info/check$_page_rank.php$
  - How well did the boiler plate removal tool extract the html content to text files.
  - On Average some of the domain that ranked higher also had a high TD-IDF. In conclusion it various result would differ on if the tools are technology used gave 100 % efficiency. We can only derive few correlations to how authentic the webpage is to the domain of the webpage.
  - Personally how authentic a webpage or it materials are is not solely dependent of how google or bing ranks the webpage alone. Over a large area I believe it very hard to quantify the relative importance/ranking of a webpage.

References

- `https://stackoverflow.com/questions/10521061/how-to-get-an-md5-checksum-in-powershell`
- `https://stackoverflow.com/questions/29889495/count-specific-string-in-text-file-using-powershell`
- `https://adamtheautomator.com/export-csv/`

- https://docs.microsoft.com/en-us/powershell/scripting/learn/deep-dives/everything-about-if?view=powershell-7.1
- https://ss64.com/ps/measure-object.html
- https://kimconnect.com/powershell-convert-csv-into-hashtable/
- https://devblogs.microsoft.com/scripting/use-a-powershell-cmdlet-to-count-files-words-and-lines/