HW 2 - Web Archiving

Adeniran Adeniyi

Sunday, Feb 28th 2021 by 11:59pm

Q1

Collect 1000 unique links from tweets in Twitter.
There are several steps involved in this:

- Obtain a Twitter Developer Account, create a standalone app in the Developer Portal, and generate consumer keys (API key and secret) and authentication tokens (access token and secret) – you should have done this in HW0
  - thread in Piazza related to this
- Write a Python program that collects links shared in tweets.
  - exclude links from the Twitter domain (twitter.com)
- Resolve all URIs to their final target URI (i.e., the one that responds with a 200) – some may be shortened links (dlvr.it, bit.ly, etc.)
- Verify that the final URIs are unique.
  - if after this step, you don't have 1000 unique URIs, go back and gather more until you are able to get to 1000 unique URIs
- Save this collection of links and upload it to your repo in GitHub – we'll use it again in HW3

Collecting Links in Tweets
There are several Twitter API wrappers available:

- Tweepy -used in the example collect-tweets.py that was provided in HW0
  - Tweepy documentation

    Cursor Tutorial
  - Search Method
  - Python - Status object in Tweepy
  - Tweet Object -Twitter API data structure returned

There are many other similar resources available on the web
Note that you'll likely need to collect more than 1000 links initially to get 1000 unique links.
There are rate limits associated with different types of API calls to Twitter. The search API has a larger limit than the streaming API, so I suggest using that. Choose a few keywords and use the search API to collect links for each of those keywords. Use keywords that you might actually search for (ex: coronavirus, election, vaccine) rather than "stopwords" (ex: test, the, tweet).
I've provided starter code, collec-links-cursor.py for collecting links from Twitter using the tweepy library.

- accepts a command-line parameter with the search term and uses "coronavirus" if no term is provided
- uses Cursor() and the search API to search for English language tweets containing the given search term
- attempts to collect 1200 links
  - if you get fewer than 1200 links and see Tweepy Error: Twitter error response: status code = 429 at the end of your file, you've hit the rate limit, so wait 15 minutes and try again

    you may want to reduce the number of links that are collected and use multiple search terms so that you can space out the requests
- prints the URIs to stdout
- does not exclude links containing twitter.com – you should add code to do this
- you will need to supply your Twitter API consumer_key and consumer_secret

```
% python3 collect-links-cursor.py archive > links.txt
```

This will run the program with the search term "archive" and write the output to the file links.txt. If you find a link you consider to be inappropriate for any reason, just discard it and get some more links.

Resolve URIs to Final Target URI

Many of the links that you collect will be shortened links (dlvr.it, bit.ly, buff.ly, etc.). We want the final URI that resolves to an HTTP 200 (not a redirection). For example:

```
$ curl -IL --silent https://t.co/DpO767Md1v | egrep -i "(HTTP/1.1|HTTP
   /2|^location:)"
HTTP/2 301
location: https://goo.gl/40yQo2
HTTP/2 302
```

```
location: https://soundcloud.com/roanoketimes/ep-95-talking-hokies-
   recruiting-one-week-before-signing-day
HTTP/1.1 200 OK
```

We want https://soundcloud.com/roanoketimes/ep-95-talking-hokies-recruiting-one-week-before-signing-day, not https://t.co/DpO767Md1v or https://goo.gl/40yQo2

You can either write a Unix shell script that uses curl to do this, or write a Python program using the requests library. Save Only Unique URIs

You can write Python code for this part, but I'd strongly recommend using the Unix tools sort and uniq. Back to Basics: Sort and Uniq is a nice introduction to this.

Answer

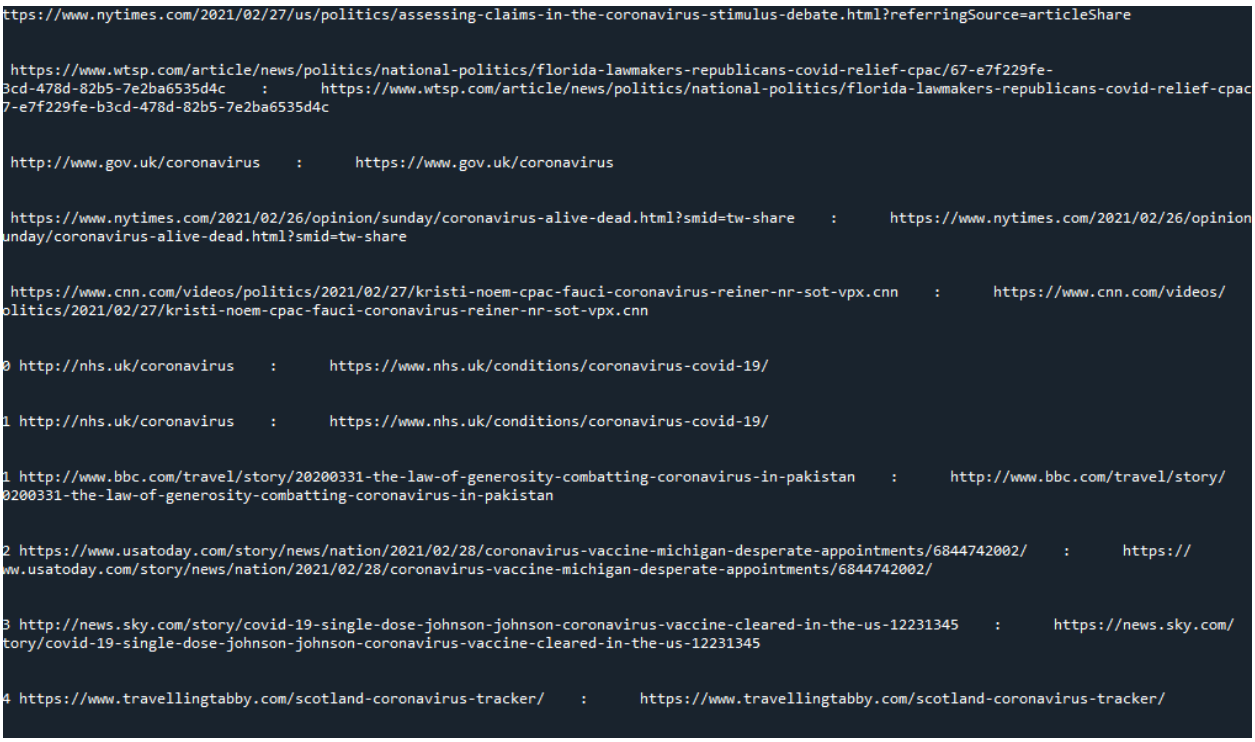

Figure 1: sample running output shows(count,URI-R gotten, resolved URI-R)

```
1 https://www.halifaxtoday.ca/coronavirus-covid-19-local-news/potential-
   covid-exposure-at-multiple-businesses-and-on-halifax-transit-routes
   -3456131
2 https://www.dallasnews.com/news/public-health/2021/02/24/dallas-county-
   reports-789-coronavirus-cases-25-deaths-tarrant-adds-470-cases-11-
   fatalities/?utm_content=buffer1ab99&utm_medium=social&utm_source=
   twitter.com&utm_campaign=buffer
```

```
https://globalnews.ca/
https://www.theguardian.com/world/2021/feb/24/covid-ad-campaign-launches-to-urge-england-to-stay-at-home?utm_source=dlvr.it&utm_medium=twitter
https://www.wltx.com/article/news/health/coronavirus/vaccine/more-vaccine-is-on-the-way-for-south-carolina-covid19/101-5485ab8a-a4d0-4666-9021-ca2c23870b1f?utm_source=dlvr.it&utm_medium=twitter
https://www.burnabynow.com/coronavirus-covid-19-local-news/bc-wont-change-school-protocols-in-the-face-of-covid-19-variants-yet-3456012?utm_source=dlvr.it&utm_medium=twitter
https://globalnews.ca/news/7661128/outbreak-declared-vernon-ltc-coronavirus/?utm_source=dlvr.it&utm_medium=twitter
https://www.kpbs.org/news/2021/feb/24/coronavirus-san-diego-live-updates-covid-19/
https://www.newwestrecord.ca/coronavirus-covid-19-local-news/bc-wont-change-school-protocols-in-the-face-of-covid-19-variants-yet-3456012?utm_source=dlvr.it&utm_medium=twitter
https://www.okotokstoday.ca/coronavirus-covid-19-local-news/okotoks-area-covid-19-update-3456148?utm_source=dlvr.it&utm_medium=twitter
https://www.thelocal.se/20210219/swedish-officials-report-escalated-threats-and-hate-in-coronavirus-debate
https://www.nytimes.com/2021/02/24/health/coronavirus-variant-nyc.html
https://www.guelphtoday.com/coronavirus-covid-19-local-news/outbreak-at-cargills-watson-parkway-facility-up-to-24-cases-3455357?utm_source=dlvr.it&utm_medium=twitter
https://www.baltimoresun.com/coronavirus/bs-md-mt-bank-mass-covid-vaccination-site-20210223-liq23iitinarjbhm6wqyfvzhv4-story.html
https://apnews.com/article/joe-biden-politics-poverty-coronavirus-pandemic-671cc3ce893e5accc0193ddacbbd72b5
https://www.abc.net.au/news/2021-02-25/victoria-coronavirus-contact-tracers-concerns-cake-shop/13186678?utm_source=abc_news_web&utm_medium=content_shared&utm_content=twitter&utm_campaign=abc_news_web
https://www.nbcsandiego.com/news/coronavirus/teachers-police-some-others-can-get-vaccine-starting-saturday-county/2531304/?amp
https://localnews8.com/health/coronavirus/2021/02/24/423-new-idaho-covid-19-cases/
https://www.msnbc.com/onassignment
https://www.jamestownsun.com/newsmd/coronavirus/6903805-South-Dakota-to-open-COVID-19-vaccines-to-those-with-1-medical-condition
https://www.cbc.ca/news/world/coronavirus-covid19-canada-world-february24-2021-1.5925859?__vfz=medium%3Dsharebar
https://www.newyorkupstate.com/coronavirus/2021/02/covid-in-ny-statewide-positive-test-rate-drops-below-3.html?utm_source=twitter&utm_campaign=newyorkupstate_sf&utm_medium=social
https://www.nbcwashington.com/news/coronavirus/virus-updates-biden-to-honor-500k-who-died-from-covid-19/2581973/?_osource=
```

```
    db_npd_nbc_wrc_twt_shr
24 https://tapnewswire.com/2021/02/latest-youtube-ban-man-made-coronavirus
    -and-people-begging-for-dna-altering-vaccines-in-mind-bending-2014-
    interview/
25 https://abc7ny.com/10365580/?ex_cid=TA_WABC_TW&taid=6035
    b900093c480001e4b0fe&utm_campaign=trueAnthem:+New+Content+(Feed)&
    utm_medium=trueAnthem&utm_source=twitter
26 https://apple.news/A72fueBexTPiV26r6uNjtug
27 https://www.wkbw.com/news/coronavirus/suny-fredonia-partners-with-
    chautauqua-county-department-of-health-on-mass-covid-19-vaccine-site
28 https://www.opb.org/article/2021/02/24/oregon-coronavirus-vaccines-long
    -term-care-facilities-covid-19-doses/
29 https://www.reuters.com/article/us-health-coronavirus-israel-vaccine/in
    -boost-for-covid-19-battle-pfizer-vaccine-found-94-effective-in-real
    -world-idUSKBN2AO2UA
30 https://www.nytimes.com/2021/02/23/health/coronavirus-california-
    variant.html
31 https://www.theguardian.com/commentisfree/2020/dec/26/ten-reasons-we-
    got-covid-19-vaccines-so-quickly-without-cutting-corners
32 https://www.wenatcheeworld.com/news/coronavirus/back-at-school-pandemic
    -continues-but-life-goes-on-at-wenatchee-and-eastmont-highs/
    article_b9d7ae06-762e-11eb-abef-bbb60008f62a.html?fbclid=
    IwAR2qXj4gmSVq6J91Tj12o7uvNq06FzpxVntUWGrrxpEXFnKJKvuZca8rrpA
33 https://www.cnbc.com/amp/2020/05/05/coronavirus-trump-says-blue-state-
    bailouts-unfair-to-republicans.html
34 https://www.nj.com/coronavirus/2021/02/nj-vice-principal-45-dies-from-
    coronavirus.html
35 https://www.youtube.com/watch?v=9ymKocjsmUo&feature=youtu.be
36 https://www.nbcphiladelphia.com/news/coronavirus/blood-banks-struggle-
    to-keep-up-with-demand-amid-pandemic/2718186/?_osource=
    SocialFlowTwt_PHBrand
37 https://apnews.com/article/personal-taxes-health-georgia-coronavirus-
    pandemic-2eb34f8526fe5b2d16a65a5ab71b0686
38 https://www.sabcnews.com/sabcnews/us-house-plans-vote-on-covid-19-aid-
    bill-on-friday/
39 https://www.reuters.com/article/health-coronavirus-israel-vaccine/in-
    boost-for-covid-19-battle-pfizer-vaccine-found-94-effective-in-real-
    world-idUSL1N2KU1T4
40 https://globalnews.ca/news/7655225/alberta-covid-coronavirus-hinshaw-
    feb-22/?utm_medium=Twitter&utm_source=%40GlobalEdmonton
41 https://www.foxnews.com/politics/biden-coronavirus-pandemic-best-thing-
    anita-dunn
42 https://samovartea.com/your-guide-to-the-ancient-wellness-practice-of-
    mindful-tea-time/?utm_campaign=your-guide-to-the-ancient-wellness-
    practice-of-mindful-tea-time&utm_medium=social_link&utm_source=
    missinglettr-twitter
43 https://pathofex.com/coronavirus-in-illinois-updates-pritzker-expects
```

```
   -100k-daily-vaccine-doses-by-next-month-as-2022-new-covid-19-cases-
   and-44-more-deaths-reported-wednesday/
https://news.yahoo.com/californias-coronavirus-strain-looks-
   increasingly-130055544.html?soc_src=social-sh&soc_trk=tw&tsrc=twtr
https://www.theguardian.com/world/2020/sep/12/coronavirus-closures-
   threaten-future-of-papua-new-guineas-only-animal-rescue-centre?
   utm_term=Autofeed&CMP=twt_gu&utm_medium&utm_source=Twitter#Echobox
   =1599871371
https://www.rivm.nl/coronavirus-covid-19/actueel/wekelijkse-update-
   epidemiologische-situatie-covid-19-in-nederland
https://www.businessinsider.com/trump-response-coronavirus-mimics-
   authoritarian-regimes-2020-2
https://www.rgj.com/story/news/2021/02/24/johnson-and-johnson-covid-
   vaccine-nevada-coronavirus-cases/6804495002/?utm_campaign=snd-
   autopilot
https://www.wtkr.com/news/national/coronavirus/israel-studies-pfizer-
   vaccine-prevents-severe-illness-stops-virus-transmission
https://www.lbc.co.uk/news/covid-vaccine-polling-hesitancy-support-
   brexit-voter-leave-oxford-university/
```

Listing 1: Part of Output text file

```
# collect-links-cursor.py
# MCW - 2/11/2021
import sys
import tweepy
import requests # for  requesting the url
from urllib.parse import urlparse # for requesting parse
import time #for time pauses to avoid max retries exceeded

def resovle_url(base_url,count):
    #print("\nProcess:    " +base_url +"\n")
    value =0
    url=""
    try:
        #request the link gotten set time out 2.5 seconds
        # This was actually useful to ensure that linked
        #gotten would not take to much time during redirection
        request = requests.head(base_url,allow_redirects=True,timeout
   =2.5)

        #Ensure a 200 responds
        if (request.status_code == 200):
            url  = request.url
            value =1
            #Gets the domain of the URI
            domain = urlparse(request.url).netloc
            #Ensures its not a twitter domain
```

```
            if domain.find("twitter.com") ==-1:
                url  = request.url
                value =1
                print("{} {}    :      {}".format( count,base_url,
   request.url))
                print("\n")
            else:
                #Reduce the counter because link wasnt resolved bc it a
    twitter domain
                count = count -1
        else:
            #Reduce the counter because link wasnt resolved
            count = count -1
        request.raise_for_status()
    except requests.exceptions.HTTPError as http_err:
        pass
        #print(f'HTTP error occurred: {http_err}')
    except Exception as err:
        #Reduce the counter because link wasnt resolved due to errors
        #from the link given
        count = count -1
        #print(f'Other error occurred: {err}')
    return  count,value, url

# use coronavirus as default search term unless one provided
search_term = "coronavirus"
if len(sys.argv) > 1:
     search_term = str(sys.argv[1])

# number of links to collect
MAX_COUNT = 1000
count = 1
url =""
value =0
start = 1
blank_dict ={}

# OAuth2 procedure
consumer_key = "pnUItdX31QmYpHBFlVcYbocKQ"      # INSERT YOUR KEY HERE
consumer_secret = "gFNX2iztwhfL1tROFCX3UomwRbU8GjUJhHzLQat8DGxvBcyVmw"
     # INSERT YOUR KEY HERE
auth = tweepy.AppAuthHandler(consumer_key, consumer_secret)
api = tweepy.API(auth)

try:
  for page in tweepy.Cursor(api.search, q=search_term, tweet_mode='
   extended', lang='en').pages():
```

```
    for tweet in page:
         for link in tweet.entities["urls"]:
             #resolve a the link gotten from twitter
             count,value, url = resovle_url(link['expanded_url'],count)
             if value != 0 :
                 if(blank_dict.get(url) != None):
                    #reduce count value if link could not be resolved
                    count = count - 1
                 else:
                    #print(count)
                    #print(url)
                    #store link and Ensures distinct url
                    blank_dict[url] = count
             count = count + 1
             time.sleep(15)
    if count > MAX_COUNT:
         #print("\n\n")
         #print(len(blank_dict))
         for x in range(0,len(blank_dict)-MAX_COUNT):
             #Remove excess link added only 1000 distinct links needed
             blank_dict.popitem()
         #used for debugging and print to console to see result values
         #[print(key,':',value) for key,value in blank_dict.items()]
         #print("\n\n")

         # swap values with keys
         blank_dict = {value:key for key, value in blank_dict.items()}

         #store result to a txt file called dict  in folder Q1
         myfile = "Q1/dict.txt"
         with open(myfile, 'w') as f:
             for key, value in blank_dict.items():
                 f.write('%s\n' % (value))

         break
except tweepy.TweepError as e:
  print ("Tweepy Error: %s" % str(e))
```

Listing 2: Collect-link-cursor.py

Discussion

I followed these instruction below:

- Obtain a Twitter Developer Account, create a standalone app in the Developer Portal, and generate consumer keys (API key and secret) and authentication tokens (access token and

secret) – you should have done this in HW0

Answer:
I created this a while back by clicking Apply for Access and followed the instructions.

Here is a picture of the created account

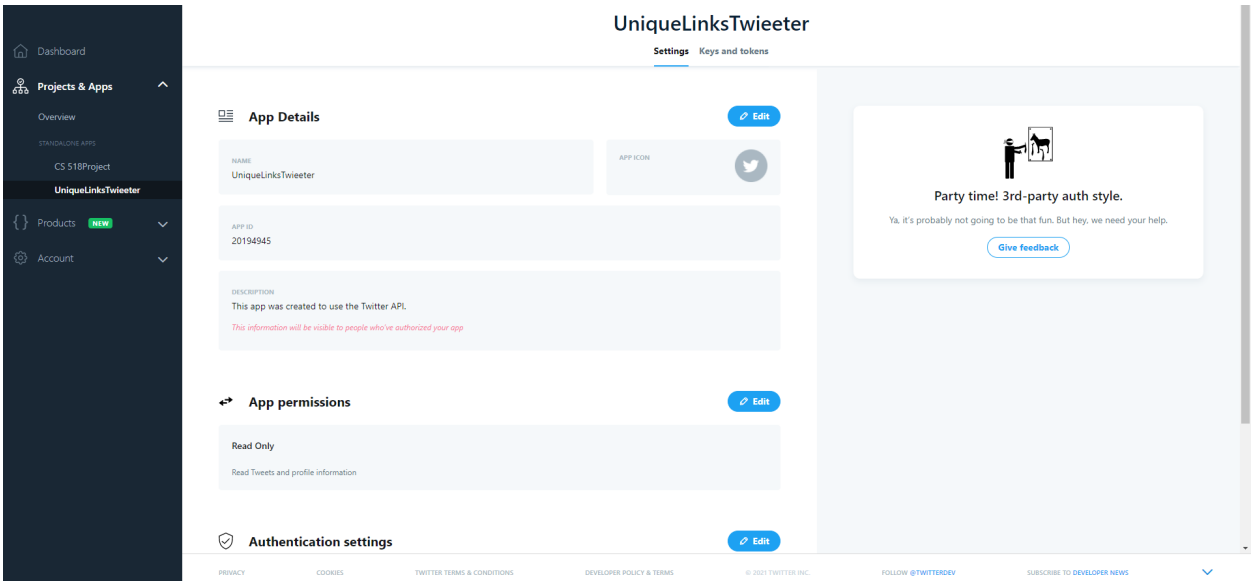

Figure 2: Development Account and Application

- Write a Python program that collects links shared in tweets.
  - exclude links from the Twitter domain (twitter.com)

```
23              #Gets the domain of the URI
24              domain = urlparse(request.url).netloc
25              #Ensures its not a twitter domain
26              if domain.find("twitter.com") ==-1:
27                  url  = request.url
28                  value =1
29                  print("{} {}    :      {}".format( count,base_url,
    request.url))
30                  print("\n")
31              else:
32                  #Reduce the counter because link wasnt resolved bc
    it a twitter domain
33                  count = count -1
```

Listing 3: snapshot of collect-link-cursor.py

Answer:

I performed this by getting the link tweepy gave me and passed it in the resolve_url function on line 72

```
72              count,value, url = resovle_url(link['expanded_url'],
   count)
```

On line 24 I got the domain of the link and checked if it was had a twitter URL.
If it wasnt a twitter URL I saved it. Making the link fully resolved.

- Resolve all URIs to their final target URI (i.e., the one that responds with a 200) – some may be shortened links (dlvr.it, bit.ly, etc.)

Answer:

I first ensured that there was a 200 status code on each request performed. This can be seen in line 20

```
20          if (request.status_code == 200):
```

In the resolve_url function on line 24

```
24              domain = urlparse(request.url).netloc
```

This gets the full final url after following any redirection.

- Verify that the final URIs are unique.
  - if after this step, you don't have 1000 unique URIs, go back and gather more until you are able to get to 1000 unique URIs

Answer:

```
74                  if(blank_dict.get(url) != None):
75                      #reduce count value if link could not be
   resolved
76                      count = count - 1
77                  else:
78                      #print(count)
79                      #print(url)
80                      #store link and Ensures distinct url
81                      blank_dict[url] = count
```

Listing 4: snapshot of collect-link-cursor.py

Once I received the resolved URL in line 72.

```
72              count,value, url = resovle_url(link['expanded_url'],
    count)
```

I used a dictionary data structure declared on line 59 to store url as the key of the dictionary and assigned an integer 1 to indicate that the string has been added to the dictionary blank_dict. Subsequently when the same link is received again on line 72

```
72              count,value, url = resovle_url(link['expanded_url'],
    count)
```

, line 74

```
74                  if(blank_dict.get(url) != None):
```

reduce the count value by one. count variable on declared on line 55

```
55 count = 1
```

tracks the number of links resolved. In line 84

```
84     if count > MAX_COUNT:
```

, ensures count gets to run 1000 times as declared on line 54.

```
54 MAX_COUNT = 1000
```

- Save this collection of links and upload it to your repo in GitHub – we'll use it again in HW3

Answer:

```
98          myfile = "Q1/dict.txt"
99          with open(myfile, 'w') as f:
100             for key, value in blank_dict.items():
101                 f.write('%s\n' % (value))
```

Listing 5: snapshot of collect-link-cursor.py label

First swapped the dictionary keys and values respectively. This can be seen on line 95.

```
95          blank_dict = {value:key for key, value in blank_dict.items
    ()}
```

Then In line 98

```
98          myfile = "Q1/dict.txt"
```

I named the file dict.txt, including the file path where the file should be stored. Finally deposted the result for each values to the text file line by line.

```
101                 f.write('%s\n' % (value))
```

Q2

Download the TimeMaps for each of the unique URIs from Q1 using the ODU Memento Aggregator, MemGator. (Save the TimeMaps and upload them to your GitHub repo – you'll also use these for Q3.)

You may use https://memgator.cs.odu.edu for limited testing, but do not request all of your 1000 TimeMaps from memgator.cs.odu.edu.

There are two options for running MemGator locally:

- Install a stand-alone version of MemGator on your own machine,
  see https://github.com/oduwsdl/MemGator/releases
  - This was described in HW0
- Install Docker Desktop and run MemGator as a Docker Container,
  see notes at https://github.com/oduwsdl/MemGator/blob/master/README.md

Important: Downloading TimeMaps requires contacting several different web archives for each URI-R. This process will take time. Look at MemGator options and figure out how to process the output before running the entire process. You might want to get JSON output, or you might want to limit to the top k archives (especially if there's one that's currently taking a long time to return).

Once you have downloaded and saved all of the TimeMaps, you will use them to analyze how well the URIs you collected are archived.

Create a table showing how many URI-Rs have certain number of mementos. For example

Table 1: A sample long table.

Mementos	URI-Rs
0	750
1	150
2	50
5	47
57	3

If you are using LaTeX, you should create a LaTeX table – don't submit a spreadsheet or image of a table created in something else. If you are using Markdown, view the source of this file for an

example of how to generate a table.

What URI-Rs had the most mementos? Did that surprise you?

Answer

```
#!/bin/bash
# [System.IO.File] class
$count = 1
$Textfie = "C:\Users\adeni\CS532\Week5\hw2-archiving-aaden001\Q1\dict.txt"
foreach($line in [System.IO.File]::ReadLines($Textfie)){

  $stringCount = $count.ToString()
  docker container run -it --rm oduwsdl/memgator --format=json  $line > C:\Users\adeni\CS532\Week5\hw2-archiving-aaden001\Q2\json\$stringCount.json
  $count++

}
```

Listing 6: runShell.ps1

```
# -*- coding: utf-8 -*-
"""
Created on Fri Feb 26 10:44:02 2021

@author: adeni
"""
import matplotlib.pyplot as plt
import pandas as pd
import json #to work with json files
from json.decoder import JSONDecodeError #for error handling json empty/wrong input file
from collections import OrderedDict #for ordering a dictionary
from datetime import datetime #
import seaborn as sns
#from matplotlib.ticker import MultipleLocator
old = 0 # track URI less old than a week
letCount = {}
accessDate = {}
dateAccessed = "2021-02-26T21:30:00Z"
count = 1
previous =0 #track highest mementos
MAX_MOMENTOUS = [] #store URI-Rs of maximum mementos
MAX_DAYS =[]  #store URI-Rs  of maximum duration
```

```
daysMementos1 =[] #to keep track of Oldest duration
daysMementos2 =[] #to keep track of momentos length
datetimeObject = datetime.strptime(dateAccessed,"%Y-%m-%dT%H:%M:%SZ")
for x in range(1000):
    #configure file ordering
    file = "Q2/json/"+ str(count) + ".json"
    try:
        with open(file, "rb") as f:
            #load file in a variable
            data = json.load(f)
            """
            ========================
            VVVVVVVVV=Q3=VVVVVVVVVV
            ========================
            Get the number of days and mementos size
            Get the URL of the highest duration(in days) -URI-R of
    Oldest Momentus
            Get the Number of URL with less than a week old
            """
            if(len(data['mementos']['list']) > 0):
                #convert to default time format
                prevsTime = datetime.strptime(data['mementos']['first']['
    datetime'],"%Y-%m-%dT%H:%M:%SZ")
                #get the duration with when the date accessed
                delta = datetimeObject - prevsTime
                #store the duration
                daysMementos1.append(delta.days)
                #store the mementos length
                daysMementos2.append(len(data['mementos']['list']))
                if(delta.days > 8112):
                    previousDay = delta.days
                    MAX_DAYS.append(data['original_uri'])

                if(delta.days < 7):
                    old = old +1
            """
            ========================
            VVVVVVVVV=Q2=VVVVVVVVV
            ========================
            When a URI has a mementos number
            first entering the dictionary -letCount- we assign the
    mementos
            to key and assign it with a value 1

            When the same memtos number is found again
            An increment is done on the value on the key
            in the dictionary
```

```
            """
            if(letCount.get(len(data['mementos']['list'])) != None):
                #get length mememto and increase count of see same
    number of memento by one
                previous =  len(data['mementos']['list'])
                letCount[len(data['mementos']['list'])]=letCount.get(len
    (data['mementos']['list'])) + 1
            else:
                #get length of memento and store a new number of
    mementos found
                letCount[len(data['mementos']['list'])] = 1
                previous =  len(data['mementos']['list'])
            #Gets URL with mementos > 10000
            if(previous> 40000):
                 #saved the URI-Rs in a list
                 MAX_MOMENTOUS.append(data['original_uri'])
     except JSONDecodeError  as e:
         #Takes care of error file therefore giving it a 0 mememtos
         if(letCount.get(0) != None):
             #increments by one when more errors zero mememtos are found
             letCount[0] = letCount.get(0) +1
         else:
             #store new error as a zero key with 1 value
             letCount[0] = 1
     #change to a new file number
     count = count +1
#sort and store
dict1 = OrderedDict(sorted(letCount.items()))
#convert to pandas object
df =pd.DataFrame(dict1.items(), columns=['Mementos','URI-Rs'])
#store as a csv file
df.to_csv("Q2/mementosURI-Rs.csv",index=False)
#[print(key,':',value) for key,value in dict1.items()]
#print("\n\n")

#convert list of top mementos URL-Rs to pandas object
f = pd.DataFrame(data=[*(MAX_MOMENTOUS)], columns=['Top URI-R With the
    Most Mementos'])
f.to_csv("Q2/topMentos.csv",index=False)


#convert list of top duration URL-Rs to pandas object
l = pd.DataFrame(data=[*(MAX_DAYS)], columns=['Top URI-R With the Most
    duration'])
l.to_csv("Q3/topDuration.csv",index=False)


```

```
109 dM = pd.DataFrame(list(zip(daysMementos1,daysMementos2)), columns=['
    days','mementos'])
110 #sort by days
111 dM =dM.sort_values(by=['days'])
112 #store as a csv file
113 dM.to_csv("Q3/durationMentos.csv",index=False)
114 #console output
115 print("============================")
116 print("============Q2============")
117 print("======Sorted by Mementos====")
118 print("============================")
119 print(df)
120 print("\n\n")
121 print(f)
122 print("\n\n")
123 print("============================")
124 print("============Q3============")
125 print("========Sorted by Days======")
126 print("============================")
127 print(dM)
128 print("\n\n")
129 print(l)
130 print("\n\n")
131 print("Number of URIs less than a week Old")
132 print(old)
133 #use seaborn
134 sns.set_style("whitegrid")
135 plt.figure(figsize=(40,15))
136 plt.title("Q3 Output File")
137 g= sns.scatterplot(x="days",y="mementos",data=dM)
138 # Show the plot
139 plt.show(g)
140 #fig, ax = plt.subplots(figsize=(40,15),sharex=False,sharey=False)
141 #g=sns.scatterplot(data=dM,x="days",y="mementos",size="days",sizes
    =(17,50))
142 print("\n\n")
143 #For console viewing
```

Listing 7: the processing data file using analyze.py

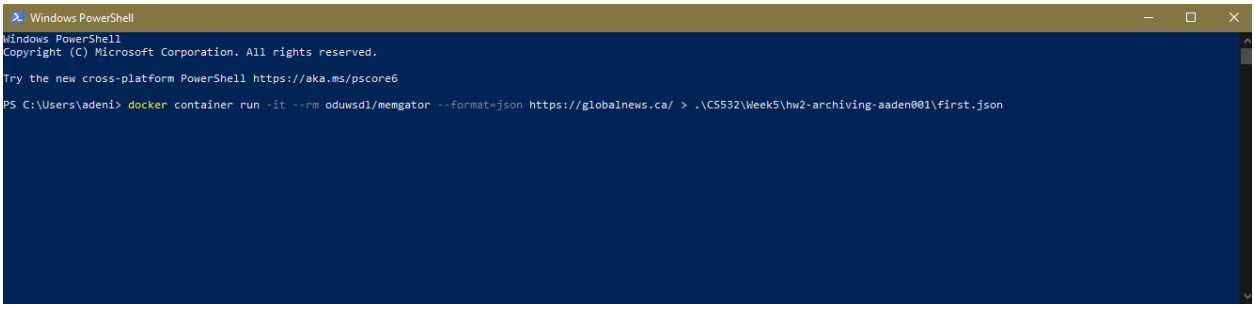

Figure 3: Console output of Question 2 (data has be sorted in ascending order based on the number of Mementos)

Discussion

I followed these instruction below:

- Run MemGator Locally using docker

  Answer

  I followed this step

  - I Installed Docker Desktop and ran MemGator as a Docker Container, see the notes at https://github.com/oduwsdl/MemGator/blob/master/README.md

- Using MemGator to download the time maps of each URL

  Answer

  I followed these steps

  - I tested the command in line 8 of runShell.psa(6) on Windows PowerShell

```
8   docker container run -it --rm oduwsdl/memgator --format=json
      $line  > C:\Users\adeni\CS532\Week5\hw2-archiving-aaden001\
    Q2\json\$stringCount.json
```

Listing 8: the automated command (snapshot runShell.ps1)

  command line. An it produced the desired output.

  I Automated the process using a shell script on windows powershell as seen in Listing 6 above.

  This runs the command through each line in the text file in Q1/dict.txt.
  Finally it deposits each result to a different text file. All text files are stored in Q2/json folder

- Get total URI-Rs For each number of Mementos

Answer

Each of the serialized text files in Q2/json are retrieved. This was done using analyze.py and the information was saved in the dictionary letCount see line 68 -79

```
68            if(letCount.get(len(data['mementos']['list'])) != None):
69                #get length mememto and increase count of see same
   number of memento by one
70                previous =  len(data['mementos']['list'])
71                letCount[len(data['mementos']['list'])]=letCount.get
   (len(data['mementos']['list'])) + 1
72            else:
73                #get length of memento and store a new number of
   mementos found
74                letCount[len(data['mementos']['list'])] = 1
75                previous =  len(data['mementos']['list'])
76            #Gets URL with mementos > 10000
77            if(previous> 40000):
78                 #saved the URI-Rs in a list
79                 MAX_MOMENTOUS.append(data['original_uri'])
```

Listing 9: processes info (snapshot analyze.py)

I sorted letCount dictionary into dict1, I then parsed the sorted dictionary to a pandas DataFrame in line 90 to 95. This result is store in a csv f

```
90 #sort and store
91 dict1 = OrderedDict(sorted(letCount.items()))
92 #convert to pandas object
93 df =pd.DataFrame(dict1.items(), columns=['Mementos','URI-Rs'])
94 #store as a csv file
95 df.to_csv("Q2/mementosURI-Rs.csv",index=False)
```

Listing 10: saving the table as a csv file using pandas (snapshot analyze.py)

I repeated the same process for the list MAX_MOMENTOUS in line 104 to 106. This result is stored in a csv file. MAX_MOMENTOUS contains the the top two URI-Rs with a Highest Momentos

```
104 #convert list of top duration URL-Rs to pandas object
105 l = pd.DataFrame(data=[*(MAX_DAYS)], columns=['Top URI-R With the
    Most duration'])
106 l.to_csv("Q3/topDuration.csv",index=False)
```

Listing 11: saving top two URIs with the highest Momentos (snapshot analyze.py)

These result can be seen on the console(4)

```
============================
============Q2============
======Sorted by Mementos====
============================
      Mementos  URI-Rs
0             0     401
1             1     190
2             2      90
3             3      50
4             4      28
..          ...     ...
120       17359       1
121       17660       1
122       26451       2
123       47180       1
124      100356       1

[125 rows x 2 columns]



   Top URI-R With the Most Mementos
0             https://globalnews.ca/
1       https://www.dailymail.co.uk
```

Figure 4: Console output of Question 2 (data has be sorted in ascending order based on the number of Mementos)

- Create a table showing how many URI-Rs have certain number of mementos

  Answer

Table 2: A table showing Number of URI-Rs with associated to a particular number of momentos

Mementos	URI-Rs
0	401
1	190
2	90
3	50
4	28
5	23
6	13
7	11
8	7
9	14
10	10
11	9
12	5
13	5
14	1
15	2
16	4
17	1
19	1
20	3
21	1
22	2
23	6
24	2
25	3
27	1
28	1
29	3
30	2
31	2
32	1
33	1
34	2
35	1
36	1
39	1
42	1

Continued on next page

Table 2 – continued from previous page

Mementos	URI-Rs
43	2
45	1
46	1
47	1
48	1
49	1
50	1
53	1
54	1
56	1
57	2
59	1
60	2
61	3
62	1
70	1
71	2
73	1
80	1
81	1
82	1
100	3
110	1
113	2
122	1
123	1
127	1
132	1
136	1
142	1
143	1
161	1
169	1
173	1
174	1
176	1
181	1
192	1
195	1
198	1
216	1
223	1

Continued on next page

Table 2 – continued from previous page

Mementos	URI-Rs
225	1
229	1
233	1
235	1
236	1
244	1
245	2
258	1
284	1
318	1
319	1
331	1
334	1
354	1
357	1
390	1
392	1
396	1
412	1
470	1
525	2
530	1
540	1
583	1
635	1
640	1
1335	1
1448	1
1478	1
1536	1
1609	1
1824	1
2129	1
2294	1
2405	1
2843	1
3715	2
3816	1
4866	1
5717	1
10000	1
17359	1

Continued on next page

Table 2 – continued from previous page

Mementos	URI-Rs
17660	1
26451	2
47180	1
100356	1

- *What URI-Rs had the most mementos?*
  - The URI-R with the most mementos came from https://globalnews.ca/ with 100,356 mementos, followed by https://www.dailymail.co.uk with 47180 mementos
- *Did that surprise you?*
  - *The main reason why this link was not a surprise is because they are top news agency network. For instance globalnews.ca according to https://en.wikipedia.org/wiki/Global_News was founded in 1994. That enough explains the reason why the URI-R is very old.*

    Similar case can be said for https://www.dailymail.co.uk according to Wikipedia

Q3

For each of the URI-Rs from Q2 that had ¿ 0 mementos, use the saved TimeMap to determine the datetime of the earliest memento.

Create a scatterplot with the age of each URI-R (days between collection date and earliest memento datetime) on the x-axis and number of mementos for that URI-R on the y-axis. For this graph, the item is the URI-R and the attributes are the estimated age of the URI-R and the number of mementos for that URI-R.

This scatterplot should be created using either R or Python, not Excel.

What can you say about the relationship between the age of a URI-R and the number of its mementos?

What URI-R had the oldest memento? How many URI-Rs had an age of ¡ 1 week, meaning that their first memento was captured the same week you collected the data?

Answer

Refer to List 7 for analyze.py full code

```
==========================================
==============Q3============================
========Sorted by Days=======
==========================================
      days  mementos
117      1         1
118      1         1
119      1         1
120      1         3
332      1         1
..     ...       ...
0     7387     47180
344   8112      1536
227   8112     17660
260   8334    100356
263   8827     17359

[599 rows x 2 columns]
```

Figure 5: Console output of Question 3 links age to mementos amount

Discussion

I followed these instruction below:

- *For each of the URI-Rs from Q2 that had <0 mementos, use the saved TimeMap to determine the datetime of the earliest memento.*

  Each of the serialized text files in Q2/json are retrieved. This was done using analyze.py and the information are saved in two dictionary daysMementos1 and daysMentos2. daysMentos1 stores the days while daysMentos2 stores the number of Mementos the URI-R has. See line 41 -49

```
41              if(len(data['mementos']['list']) > 0):
42                  #convert to default time format
43                  prevsTime = datetime.strptime(data['mementos']['
    first']['datetime'],"%Y-%m-%dT%H:%M:%SZ")
44                  #get the duration with when the date accessed
45                  delta = datetimeObject - prevsTime
46                  #store the duration
47                  daysMementos1.append(delta.days)
48                  #store the mementos length
49                  daysMementos2.append(len(data['mementos']['list']))
```

Listing 12: processes info (snapshot analyze.py)

I parsed these two dictionaries to a pandas DataFrame,sorted the values by the number of days in ascending order and store the result in a csv f in line 109 to 113.

```
109 dM = pd.DataFrame(list(zip(daysMementos1,daysMementos2)), columns=[
    'days','mementos'])
110 #sort by days
111 dM =dM.sort_values(by=['days'])
112 #store as a csv file
113 dM.to_csv("Q3/durationMentos.csv",index=False)
```

Listing 13: saving the table as a csv file using pandas (snapshot analyze.py)

These result can be seen on the console(5)

- *Create a scatterplot with the age of each URI-R (days between collection date and earliest memento datetime) on the x-axis and number of mementos for that URI-R on the y-axis. For this graph, the item is the URI-R and the attributes are the estimated age of the URI-R and the number of mementos for that URI-R.*

  Answer

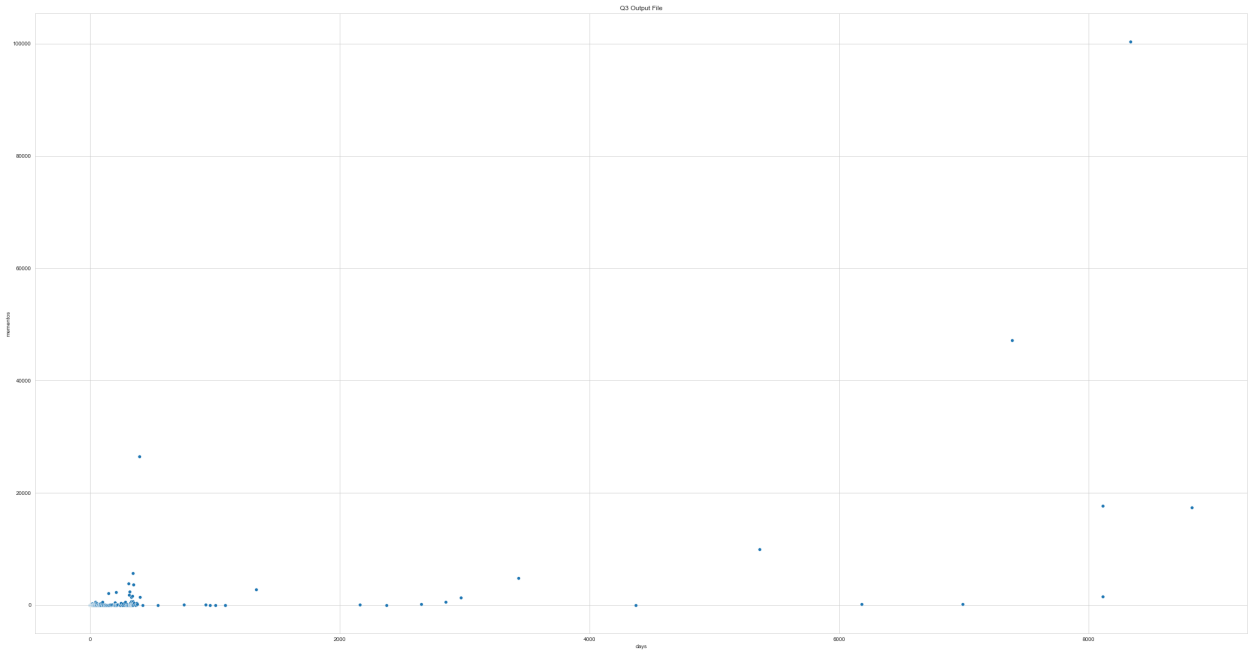

Figure 6: Console output of Question 2 (data has be sorted in ascending order based on the number of Mementos)

- *What can you say about the relationship between the age of a URI-R and the number of its mementos?*

  Answer

  - *URI-Rs that tended towards zero days or less that seven day had a lower number of momentos. This means definitely correlates because these are URI-Rs that are new and therefore do not have no or less history*

- *What URI-R had the oldest memento? How many URI-Rs had an age of ¡ 1 week, meaning that their first memento was captured the same week you collected the data?*

  Answer

  - In collecting the two top URI-Rs with the highes momentos, I compared the age of the current URI-Rs $>$ 8112 to get the top two URI-Rs

```
50                  if(delta.days > 8112):
51                      previousDay = delta.days
52                      MAX_DAYS.append(data['original_uri'])
```

Listing 14: count momento greater than 8112 days

This was done by comparing age of the URI-R(in days) $<$7. Using the variable old declare at zero in line 15. To increment count when this condition is satisfied.

```
54                  if(delta.days < 7):
```

```
55                    old = old +1
```

Listing 15: count momento less than 7 days

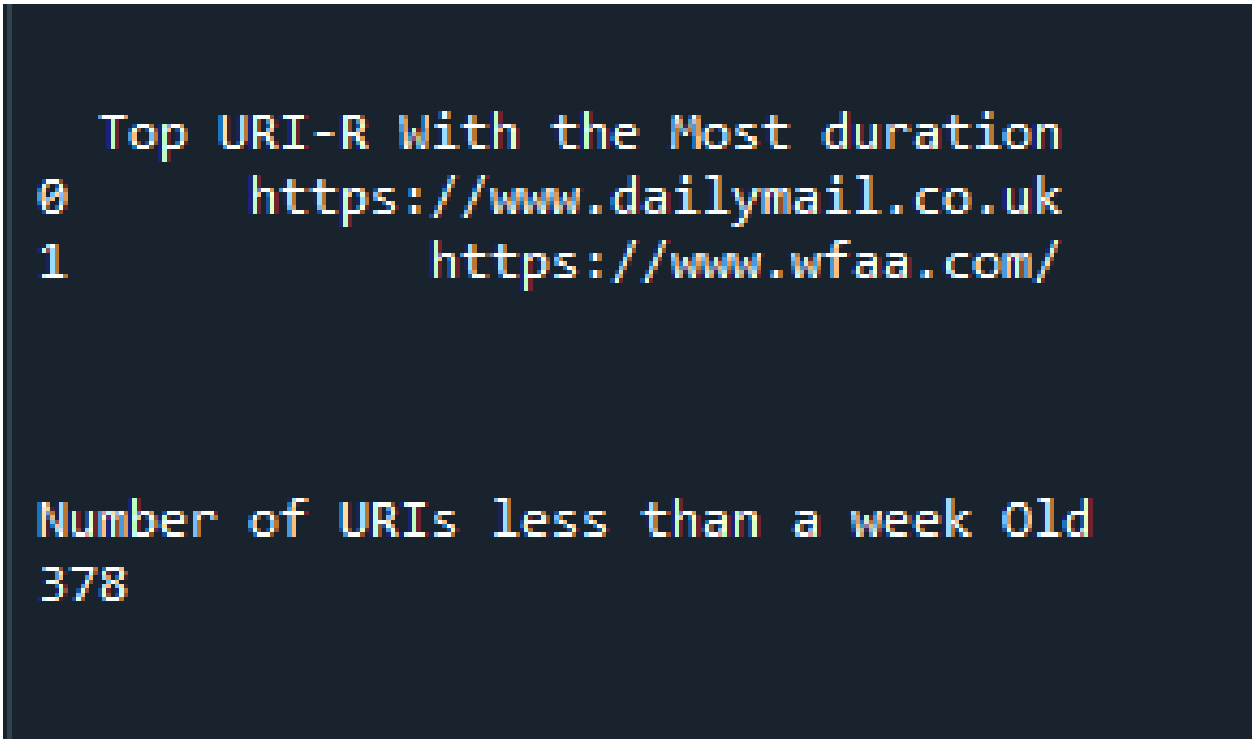

Figure 7: Console output of Question 3 top Oldest URI-R & URI's less that 7 days

Extra Credit

Q4 (2 points)

Create an account at Conifer and create a collection. Archive at least 10 webpages related to a common topic that you find interesting. Make the collection public and include the link to your collection in your report.

Why did you choose this particular topic? Did you have any issues in archiving the webpages? Do the archived webpages look like the original webpages?

After creating your collection at Conifer, download the collection as a WARC file (see Exporting or Downloading Content).

Then load this WARC file into ReplayWeb.page, a tool from the Webrecorder Project (folks who developed Conifer). From https://webrecorder.net/tools:

ReplayWeb.page provides a web archive replay system as a single web site (which also works offline), allowing users to view web archives from anywhere, including local computer or even Google Drive. See the User guide for more info.

Once the WARC file has loaded, click on the "Pages" tab. Take a screenshot that includes the list of pages and the browser address bar (showing replayweb.page/?source=file

Then click on the "URLs" tab and choose "All URLs" from the dropdown menu. How many URLs were archived in the WARC file? How does this compare to the number of Pages?

Create a bar chart showing the number of URLs in the WARC file for each of the file types in the dropdown menu.

Which file type had the most URLs? Were you surprised by this?

Answer

- *Create an account at Conifer and create a collection. Archive at least 10 webpages related toa common topic that you find interesting. Make the collection public and include the link to yourcollection in your report*
  - https://conifer.rhizome.org/aaden001/crytocurrency
- *Why did you choose this particular topic? Did you have any issues in archiving the webpages? Dothe archived webpages look like the original webpages?*
  - Because crytocurrency has be a major topic on every platform today.

    I did not have any issues archiving the web page
  - When I went through the archived webpages the looked very similar
- Once the WARC file has loaded, click on the "Pages" tab. Take a screenshot that includes thelist of pages and the browser address bar (showing replayweb.page/?source=file

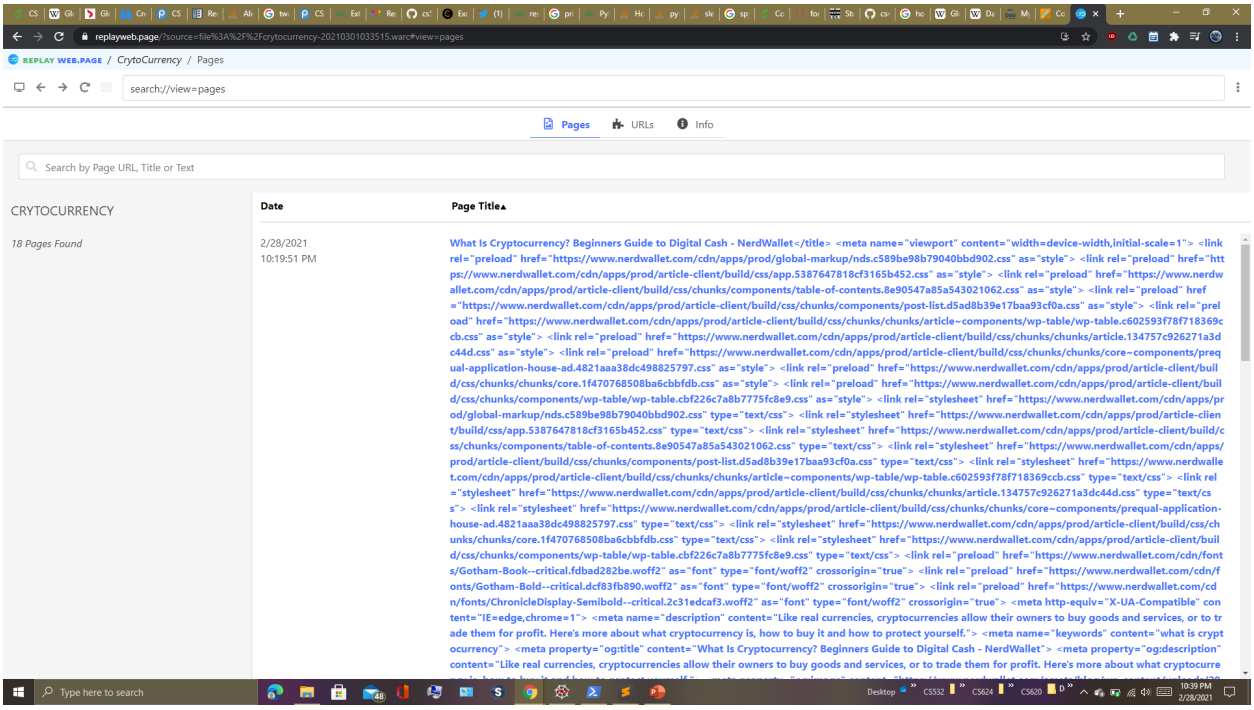

Figure 8: File type and corresponding numbers

- Then click on the ”URLs” tab and choose ”All URLs” from the dropdown menu. How manyURLs were archived in the WARC file? How does this compare to the number of Pages?

- Create a bar chart showing the number of URLs in the WARC file for each of the file types inthe dropdown menu

  - The bar chart was easily created using quickPlot.py using the code

```
# -*- coding: utf-8 -*-
"""
Created on Sun Feb 28 22:46:21 2021

@author: adeni
"""
import matplotlib.pyplot as plt


FileType = ['HTML','Images','Audio/Video','PDF','JavaScript','
    CSS','Fonts','Plain text','Json','Dash/HlS']
Number_Of_Url = [84,183,0,0,250,77,58,8,107,0]
plt.figure(figsize=(20,10))
plt.bar(FileType, Number_Of_Url)
plt.title('FileType Vs Number_Of_Url')
plt.xlabel('FileType')
plt.ylabel('Number_Of_Url')
```

```
17 plt.show()
```

Listing 16: count momento greater than 8112 days

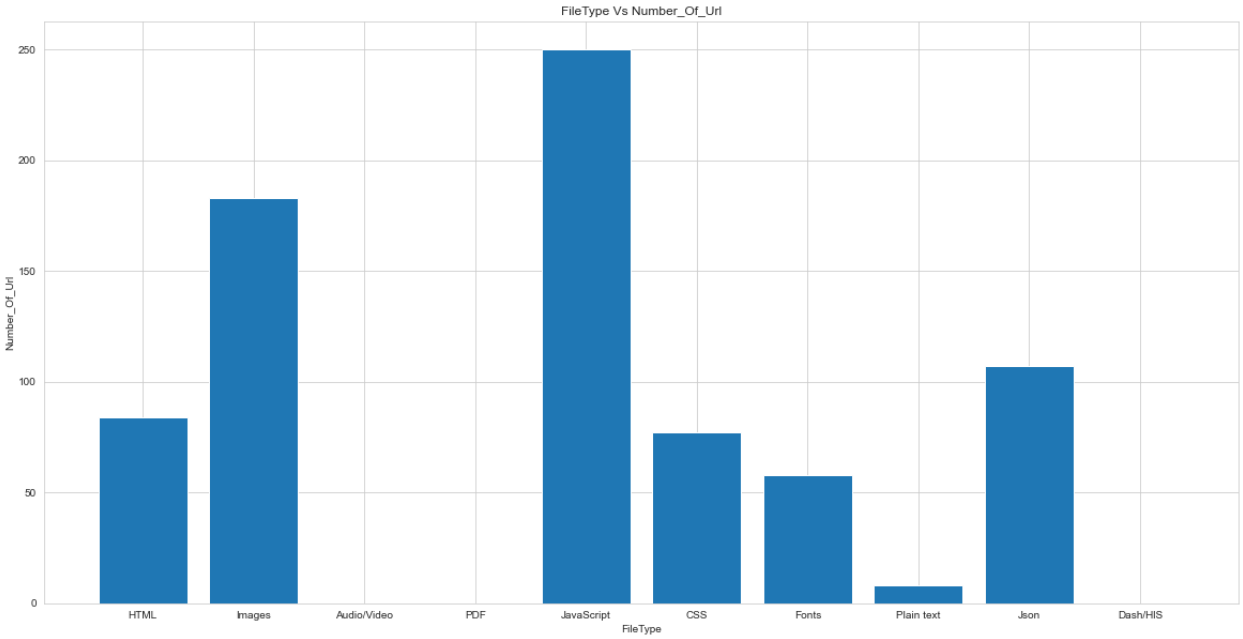

Figure 9: File type and corresponding numbers

- Which file type had the most URLs? Were you surprised by this?
  - Java script had the most file. This is no supprise to me because a vast amount of webpages use javascripts.

References

- https://pythonspot.com/save-a-dictionary-to-a-file/
- https://note.nkmk.me/en/python-function-return-multiple-values/
- https://www.geeksforgeeks.org/g-fact-41-multiple-return-values-in-python/
- https://www.quora.com/How-do-I-write-a-dictionary-to-a-file-in-Python
- https://www.w3schools.com/python/ref_requests_head.asp
- https://stackoverflow.com/questions/14219092/bash-script-and-bin-bashm-bad-interpreter-no-such-file-or-directory

- https://stackoverflow.com/questions/1098786/run-bash-script-from-windows-powershell
- https://www.youtube.com/watch?v=YrZLCDsh5a0
- https://stackoverflow.com/questions/45255361/raise-jsondecodeerrorexpec value-s-err-value
- https://wiki.python.org/moin/ForLoop
- https://stackoverflow.com/questions/53289238/convert-a-dictionary-to-dataframe-with-specified-column-names
- https://www.journaldev.com/23365/python-string-to-datetime-strptime
- https://www.geeksforgeeks.org/create-a-pandas-dataframe-from-lists/
- https://www.tablesgenerator.com/latex_tables
- https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.sort_values.html