Web Scraping Reddit To Find Sources of Conspiracy Theories

Barrett Lattimer

1.0 Introduction

Conspiracy theories and fake news have been sources of huge controversy during recent times. What exactly is a conspiracy theory? A conspiracy theory is defined by dictionary.com as “a theory that rejects the standard explanation for an event and instead credits a covert group or organization with carrying out a secret plot.” Note that a conspiracy theory is not necessarily false; however, they commonly rely on misinformation and speculation to make a point. Recent concerns about the prevalence of conspiracy theories on the internet are growing, but if we can find the sources of conspiracy theories, we could use this information to eradicate misinformation from the source.

We now have the motivation for the first research question, RQ1: Do conspiracy theories originate from certain websites on the internet, and if so, what are these sites? Before we go further, we must define some terms and provide the general approach. *Domains* refer to the domain name of a corresponding URL, so the domain name for https://www.youtube.com/results?search\_query=web+scraping is youtube.com. We define *hits* as the number of times a domain is referenced during web scraping, so every reference to a domain increments its hits by 1. How do we determine the sources of these theories? In this work, we use a method called web scraping to determine the sources of conspiracy theories on the web. The concept of web scraping used is taking an origin URL that usually refers to an article about conspiracy theory and visiting all the referenced links in the origin URL keeping track of the number of times a domain has been encountered and which domains referenced each other. In this work, this is then done recursively for every valid URL that is referenced until there are two degrees of difference between any given URL and the origin URL.

Our definition of web scraping begs three separate questions. RQ2: How do we create a Web Scraper? RQ3: How do you determine if a URL is valid to be searched in the web scraper? RQ4: How do you clearly model web scraping data? The origin URLs are pulled from the subreddit r/conspiracy. r/conspiracy is a community (also referred to as a subreddit) on the website reddit.com that encourages the sharing of personal or public conspiracy theories. For this effort, we have pulled out approximately 300,000 URL’s that are referenced in posts on the subreddit. The URL’s pulled from r/conspiracy act as our origin URL’s. This project uses web scraping to find the frequency of domains referenced by conspiracy theories from r/conspiracy and then analyze the findings to try and find common references.

1. Methods

2.1 Data Collection

The data used as origin URL’s for web scraping were pulled from r/conspiracy. On r/conspiracy some posts are simply text, but others contain links to conspiracy related articles and these are the posts and URL’s that are being scraped. All these posts are stored in a JSON file and there are about 300,000 total URL’s.

* 1. Web Scraper

We begin by declaring a list to keep track of all the Level 1 URL’s, a dictionary to keep track of the number of hits a domain name gets, a list of tuples to record which domain names refer to each other, and a dictionary of domain corresponding to the first level they appear on. In addition, we use the netloc variable generated by urlparse from urllib.parse to extract a domain from a URL. Now we load all the origin URL’s from the JSON file created by data collection. We refer to these as the “Level 1” URL’s. Due to computer limitations and BeautifulSoup timeouts, I was only able to scrape cases with 10, 50, 100, and 150 Level 1 URL’s. Once the number of Level 1 URL’s is determined, they are loaded into the Level 1 list starting with the first URL in the JSON file. As they are loaded, the URL’s are tested to see if they are valid links (described in Section 2.3), and their domain names are extracted and incremented or added in the dictionary storing the amount of hits a domain gets. Each of these Level 1 URL’s is then passed to a recursive function ExploreLinks().

ExploreLinks does the following. If the current level is 3, we return. If not, we take in a parent URL as an argument and extract the domain name and html code. Using BeautifulSoup, we then locate all the URL’s contained within paragraph tags. We refer to these as child URL’s. Each child URL is then validated, continuing to the next child URL if the current one is not valid. Next, the child’s domain is extracted and validated (explained in Section 2.3), the child domain then either creates an entry in the hit tracking dictionary with 1 hit or increments the number of hits the domain has. Lastly, we add a tuple containing the parent and child domain to the domain connection list and update the levels dictionary if the child domain is not already a key. The child URL is then passed to ExploreLinks. Once ExploreLinks is called on every single Level 1 URL, we have all the data for the current set of Level 1 on what domains are referenced most frequently, what domains reference each other, and what level each domain first appears.

* 1. URL and Domain Validation

We validate a URL and domains through a series of tests. A URL is considered valid if it is not null and is alive. URL’s are tested to be alive by sending a single request and if it succeeds the URL is considered alive, otherwise it is considered invalid. Domains are determined valid under three criteria: the domain is not null, if the domain is not empty, and if the domain being considered is not the same as the parent domain. If all three of those conditions are true then the domain is considered valid; however, if one is false the child domain is thrown out and the next domain is analyzed.

* 1. Graphing Results

The graphing is done with the networkx library in conjunction with the bokeh library. Originally all graphs are generated by using the keys from the domain hit counter dictionary as nodes and the tuples from the list of domain connection tuples as edges. Each node is then colored using the seaborn library based off the level the domain was first seen. In this work, graphs included blue for Level 1, yellow for Level 2, and green for Level 3. Some graphs become very crowded if a minimum hit value is not imposed on the nodes, so I have generated multiple graphs, some with no minimum value imposed and some with various minimum values depending on what makes sense for a given graph.

3.0 Results

**Figure 1. Number of Hits per level for different Level 1 URL’s**

Figure 1 shows that regardless of the number of origin URL’s, there is still an exponential increase in the amount of domains found in each level.

**A screenshot of a cell phone

Description automatically generated**

**Table 2. The top 10 most visited domains and their corresponding hit numbers with different number of Level 1 URL’s**

Table 2 shows the dominant domains found when different amounts of Level 1 URL’s are used with their corresponding number of hits. Notice the mix of reliable news sources such as New York Times right along side very unreliable sources such as Twitter, Facebook, and Lermanet.

**A screenshot of a cell phone

Description automatically generated**

Figures 2-9 can also be seen as interactive graphs attached.

**Figure 2. Graph of connected domains with 10 Level 1 URLs and 1 minimum hit**

Starting off we can tell that there are more green (Level 3) hits than anything else on the graph.

**A close up of a map

Description automatically generated**

**Figure 3. Graph of connected domains with 10 Level 1 URLs and 2 minimum hits**

As we can see, most of the current domains are only referenced about 2 times. More Level 1 URL’s will be needed to find more significant numbers.

**A close up of a map

Description automatically generated**

**Figure 4. Graph of connected domains with 50 Level 1 URLs and 1 minimum hit**

Compared to the 10 Level 1 URL’s, notice how many of the green nodes got larger without much of an increase in blue or yellow. This could potentially show that Level 1 and 2 nodes are frequently reaching the same domains.

**A close up of a map

Description automatically generated**

**Figure 5. Graph of connected domains with 50 Level 1 URLs and 5 minimum hits**

This graph shows the small amount of hits domains are still getting at only 50 Level 1 URL’s. With a minimum of 5 imposed there are only 10 nodes.

**A close up of a map

Description automatically generated**

**Figure 6. Graph of connected domains with 100 Level 1 URLs and 1 minimum hit**

There is a sharp increase in connections in this graph and the clear emergence of some commonly referenced domains such as Twitter, Facebook, and Wiki Creative Commons.

**A close up of a map

Description automatically generated**

**Figure 7. Graph of connected domains with 100 Level 1 URLs and 5 minimum hits**

Here we can see a lot of the lesser domains cleared out but some dominating domains such as Twitter emerging.

**A bunch of different types of map

Description automatically generated**

**Figure 8. Graph of connected domains with 150 Level 1 URLs and 1 minimum hit**

We can see the graph now being completely dominated by Twitter and Lermanet.

**A close up of a map

Description automatically generated**

**Figure 9. Graph of connected domains with 150 Level 1 URLs and 10 minimum hits**

With a 10 hit minimum imposed, the number of nodes in the graph drops significantly. We can see many of the Level 1 nodes that were referenced frequently also refer to Twitter greatly.

**A close up of a map

Description automatically generated**

1. Discussion

Table 2 reveals that as the number of Level 1 URL’s increases twitter becomes the dominant domain referenced. Facebook, Wiki Creative Commons, and Lermanet are also very frequently referenced by conspiracy articles along with the New York Times. My hypothesis is that the existence of both reliable news sources such as Washington Post or the New York Times alongside unreliable sources such as Twitter, Facebook, or Lermanet is because conspiracists frequently pull outlandish claims from an unreliable source and then cherry pick facts from respected news outlets to support their claims. This hypothesis cannot be supported; however, it is an interesting interpretation of the data.

Looking at the various graphs generated through bokeh, we can see that most of the domains with high hits such as Twitter, Wiki Creative Commons, and Lermanet are all Level 3 domains meaning the first time they were encountered was at Level 3. This is good data to show that conspiracy theory articles in fact do stem from similar web sites regardless of content. There is no statistical analysis on this claim currently.

My data provides strong support in the direction of RQ1 being yes, conspiracy theories do stem from specific websites on the internet; however, there is no statistical analysis supporting this claim. RQ2 is answered in Section 2.2 in detail where we use a recursive exploration of all the referenced links for three levels of links. RQ3 is answered in Section 2.3 with the main point being that domains are ruled invalid if they are the same as their parent domain. Finally, RQ4 is answered in Section 2.4 where we graph a clear representation of web scraped data through connected graphs with nodes being domains, edges being references between domains, and size of the nodes being the number of hits for the domain.

1. Conclusion

In this paper, URL’s from r/conspiracy were web scraped to find the frequency of domains being referenced. The levels at which domains were first referenced were also recorded to try and find commonly referenced domains by conspiracy theory articles. While many claims were made by my studies, the lack of statistical support leads me to reject my hypothesis that conspiracy theories in r/conspiracy stem from core websites. There are a few improvements I could make on this project in the future. I would like to do an actual statistical analysis on the data that was collected. I feel as though the data could very well be significant; however, I cannot support my claims as they stand so that is the top priority for future work. I would like to add directed edges to the bokeh graphs that were generated so you can see which domains children of other domains were. Also, I recommend adding edge weights to the graphs to see the strength of connection between two domains. BeautifulSoup hanging on its constructor after more than 150 URL’s being visited limited my ability to get a large data set, so I would need to solve that problem should my work continue. I would add a deeper level search as well, instead of stopping at three levels maybe stop at five to see if any data changes. Looking at data given by the web scraper, I would want to mesh similar domains together such as t.co, support.twitter.com, and twitter.com. These are the same website with slightly different domain names and developing some way to detect that would be extremely useful and improve the quality of the results. Lastly, I would experiment with more restrictions on valid domains, possibly imposing a restriction that for every Level 1 URL and its children a domain can only be referenced once. I think this could put an interesting twist on the data and focus more on where level one domains are joining rather that what domain has the most hits.