CS249i Project 2: Web and Content Delivery P1/P2
Mo Akintan (makintan@stanford.edu)
Christina Rogers (rogerscj@stanford.edu)
2/18/2024

Files included:

- starter_code.js: javascript code for gathering meta data of websites (is ran first)
- asn_tag.py: python code that tagged ASN and AS names for metadata entries (is ran after start_code.js)
- output.jsonl: final output data for metadata of domains crawled

Description:

We started off by working with the given javascript code to crawl the website to gather metadata for just a single domain. Once we got this working we tested it out for 25 domains in the csv file to see special cases. We used a csv parser to handle the csv file containing the domain names and limited the number of domains to only the first 1000 and initialized the crawling process. We resolved each domain's IP address, launched a headless browser instance using puppeteer and navigated to the domain's URL where we collected the metadata about each resource loaded on the page. We also did some error handling for when an error occurs during metadata collection so it handles it and moves on to the next domain without issues. Specifically the error was a "Protocol Timeout error" and we bypassed this by using a try and catch and adding a timeout of 1000000ms. A post on Ed was really helpful here about this. Some of the errors we got while running this for all th website were:

1. Error: net::ERR_CONNECTION_TIMED_OUT
2. Error: net::ERR_HTTP2_PROTOCOL_ERROR
3. Error: getaddrinfo ENOTFOUND

We resolved this by filling those entries with "UNKOWN"

We then wanted to tag the ip addresses with their ASN and AS names. To do this we wrote a python script that used pyasn, a python library, that iiterated over the output json file from the previous step. It then tagged these ASN using the latest ipasn database.

The final output we got seems to tag most sites appropriately, but there were still some websites that seemed off or caused errors. For example, we saw some data URLs which seemed obviously off containing spaces, however, this is fine and was not removed as it is a way to embed a small resource (e.g., an image) directly into the URL.

Part 3: Understanding Web Resources

In this section, we will begin by investigating the most common resources that power the modern web. Please share your methodology (a high-level description will suffice) for answering each question.

1. What percent of sites were you able to successfully crawl? For the sites that failed, why did the crawl fail?
    a. We successfully crawled **97.7%** of sites. To identify unsuccessful sites, we checked for entries with an 'UNKNOWN' site IPs, representing failed loads (according to our javascript crawling code). Among the 1000 sites, 23 failed to load successfully, resulting in a success rate of 97.7%.
    b. We utilized a try-catch mechanism to capture errors during crawling, logging them to an error.json file for analysis. Common errors included::
        i. Error: net::ERR_HTTP2_PROTOCOL_ERROR : Indicates there was an issue with HTTP2 protocal
        ii. Error: net::ERR_CONNECTION_TIMED_OUT : Indicates the site did not respond withing the timeout period which we extended to be 1000000 ms
        iii. Error: getaddrinfo ENOTFOUND : Indicates the domain could not be resolved to an IP
        iv. Error: net::ERR_EMPTY_RESPONSE : Indicates no data was sent back from site
        v. Error: net::ERR_CONNECTION_RESET : Indicates the connection was closed

2. How many resources, on average, are loaded per site? What are the top ten sites that load the greatest number of resources? How many resources do those sites load? Why are so many resources loaded for these top sites?
    a. On average, each site loads **110.56** resources. We calculated this by dividing the total resources loaded by successfully loaded sites. We excluding unsuccessful sites because these sites should not count as having zero resources, but rather be an unknown.
    b. To find the top ten sites that loaded the greatest number of resources we created a datastructure that tracked sites against number of resources and then sorted this list. The top ten sites loading the most resources are:

Site	Number of Resources
https://vnexpress.net	12089
https://nypost.com	1997
https://www.ynet.co.il	1980
https://www.merriam-webster.com	1380
https://www.nexusmods.com	1225

https://item.rakuten.co.jp	877
https://letterboxd.com	870
https://nlab.itmedia.co.jp	852
https://www.w3schools.com	836
https://weather.com	821

c. A large number of the leading websites belong to the news category, including all of the top 3. This increased resource load is likely due to news needing to gather information and images from external sources. Further, news sites commonly use ads to generate revene which also would need to be loaded. Some of the other sites contain video games, video hosting, e-commerce, and online courses. These also necessitate many external images, videos, and other resources.

3. What are the most common resource types across all websites in your crawl? You can determine this by investigating the content type of each resource loaded. What fraction of sites use JavaScript?
   a. To identify the most common resource types, we went through all the resources and counted how often each resource type appeared. We noticed some inconsistencies in the content_type values returned by the crawl, such as variations in casing. To standardize, we converted all content_type values to lowercase. Some content types included a charset, while others did not. We chose to remove this information when comparing, prioritizing the content type over its encoding method. Additionally, we observed resource types with different prefixes (e.g., application/javascript vs. text/javascript). Opting for a more inclusive approach, we grouped these variations together as "javascript" to avoid fine granularity. However, for a more detailed analysis, these distinctions could be included. The resulting list highlighted the most common resource types.

Resource type	Number of Resources
javascript	20491
jpeg	12028
gif	11605
html	10410
png	8576
json	6355
webp	5013

css	4425
svg+xml	4298

b. The fraction of sites utilizing Javascript is **89.5%**. This calculation involved counting the number of sites where at least one resource had a content type containing the string 'javascript' and then dividing this count by the total number of successfully loaded sites. We ensured that this calculation was performed on a per-site basis, meaning that even if a site had multiple Javascript resources, it was only counted once.

4. What are the top 10 most commonly included resources across all websites (as defined by file name)? How frequently do they occur in the Top 1000 websites, and what function do they serve?
    a. To identify the most common resources, we iterated through each site's resources, tracking counts for each resource name. Specifically, we extracted the resource name by locating the end of the URL file name string (the final part after the '/') but before any '?'. We opted to count on a per-site basis, ensuring that even if the same resource was included multiple times, we only recorded it once. We made this choice to prevent our results from being influenced by a few sites that include a particular resource many times, providing a broader perspective. Additionally, we excluded resources that appeared unusual to us. For instance, the 8th most occurring resource, 'v84a3a4012de94ce1a686ba8c167c359c1696973893317,' was excluded due to its unusual nature. The table below displays the top ten most common resources we found, along with the functions we believe they serve (determined through online research).

Resource name	Frequency	Function resources serve
analytics.js	259	Enables you to send your data to hundreds of destination tools without having to learn, test, or use a new API every time (link)
gpt.js	133	Google Publisher Tag (GPT) is an ad tagging library for Google Ad Manager (link)
pubads_impl.js	103	Google Publisher Tag (GPT) is an ad tag library that allows publishers to define inventory, initiate and bundle ad requests, and render matching demand (link)
jquery.min.js	102	Compressed version of jqueury - jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation,

		and Ajax much simpler with an easy-to-use API (link)
KFOmCnqEu92Fr1Mu4mxK.woff2	93	Exact purpose is not clear, but WOFF2 are compressed file formats created specifically for web fonts (link)
container.html	91	Likely used to enclose elements like text, images, and videos in an HTML container (link)
api.js	76	Exact purpose unknown, but likely used to interact with an API
favicon.ico	69	A favicon is a small 16×16 pixel icon that serves as branding for your website (link)
logo.png	68	Likely an logo image for a website
ufs_web_display.js	68	Exact purpose unclear, but it likely related to displaying content

5. What are the top domains that serve resources on the most websites? What fraction of websites rely on these domains? What are these domains being used for? Answer this in a table. We recommend using the Mozilla Public Suffix List6 to group URLs by domain, for which there are many libraries.
    a. To discover the domains that serve the most resources, we counted each time a domain appeared within a sites resource list by appending that site to a set for that domain. We used a Python library called tldextract to identify domains. Subsequently, we sorted this domain structure based on the length of the sites array within it which ensured that each site was only counted once, even if it had multiple resources to that domain. After identifying the top domains, we found their purposes by extracting the various content types they provided. The results are presented in the table below:

Domain	Fraction of sites that rely on this domain	What are these domains being used for
googletagmanager	42.89%	javascript, gif, html
google-analytics	35.11%	plain, javascript, gif, html
doubleclick	31.32%	html, gif, plain, javascript, png, x-icon, json, xml
google	30.91%	html, gif, svg+xml, json+protobuf, plain,

		jpeg, javascript, png, binary, css, json, webp, mp4
gstatic	23.03%	html, ttf, woff, gif, svg+xml, woff2, jpeg, javascript, png, webp, css, json, mp4
googleapis	22.01%	x-javascript, html, svg+xml, json+protobuf, javascript, png, css, json
blob	15.35%	png, javascript, mp4
googlesyndication	15.25%	html, none, gif, jpeg, plain, javascript, png, json
cloudflare	13.20%	html, none, plain, octet-stream, png, javascript, css
adnxs	11.77%	javascript, xml, x-javascript, gif, html, jpeg, json

6. For this and the next question, consider a third-party domain to be one that is not equivalent to the domain of the website you visited based on the public suffix list. For example: google.com and blog.google.com are first-party domains, whereas google.com and googledomains.com are thirdparty domains. How many websites depend on third-party content (i.e., a resource loaded from a third party domain)? On average, how many third-party domains do websites rely on? What are the top 10 most included third-party resources (as defined by URL), and what do they do?
   a. We identified the number of websites relying on third-party content by comparing the domain of the site with that of each resource. If the resource and site URL did not match, it was considered a third-party URL. We counted the occurrences of this on both a per-site basis and a total count. We found that **883** sites depend on third-party content.
   b. Next, we calculated the average resources each site relies on by dividing the total count of third-party domains by the per-site count. We found that sites rely on an average of **81.503** external sites
   c. To identify the most included resources, we constructed a mapping of third-party domains to a set of sites they provide for. The results are presented in the table below:

Resource	Number of sites that rely on it	What site does
googletagmanager	419	manage analytics and advertising tools in your apps (link)

google-analytics	343	collects data from your websites and apps to create reports that provide insights into your business (link)
doubleclick	306	let advertisers control how often an ad is shown to a browser (link)
google	302	organize the world's information and make it universally accessible and useful (link)
gstatic	225	store crucial data, including JavaScript code, images, and style sheets (link)
googleapis	215	allows developers to access and manipulate data stored in various Google services (link)
blob	150	storing media, large file backups, and data logs (link)
googlesyndication	149	protects you from trackers and malvertising by enforcing the principles of Ethical Design (link)
cloudfare	129	enables developers to build serverless applications or add to existing applications with its serverless platform, Cloudflare Workers (link)
adnxs	115	provides technology, data and analytics to help companies buy and sell online display advertising (link)

7. What percent of sites load Javascript from third parties? How widely adopted is Subresouce Integrity? What attack can occur when SRI is not used? What are the ten most commonly included Javascript files from third-party domains and what role do those files play in the web ecosystem?
    a. We found that **81.7%** of sites load javascript from third parties. This was found by finding the sites that have at least one third party resource (from the previous step) that has a content type that is javascript.
    b. To find the number of sites that use SRI we used the resource python library and checked for the presence of the integrity attribute in either <script> elements or <link> elements with specific rel values ("style sheet", "preload", or "modulepreload"). We found that aroun **10.03%** of sites use SRI. So, it is not that common.
    c. Without SRI, there's a risk that attackers may inject malicious code into resources, leading to data manipulation (if they gain control of the CDN). SRI

helps mitigate these threats by verifying the integrity of fetched resources through cryptographic hashes.

d. The ten most included javascript files were found similar to the approach for the most common resources in question 6, but now instead tracking Javascript files from third party domains. The results are shown below:

Javascript files from third party domains	Role
analytics.js	Enables you to send your data to hundreds of destination tools without having to learn, test, or use a new API every time (link)
gpt.js	Google Publisher Tag (GPT) is an ad tagging library for Google Ad Manager (link)
pubads_impl.js	Google Publisher Tag (GPT) is an ad tag library that allows publishers to define inventory, initiate and bundle ad requests, and render matching demand (link)
jquery.min.js	Compressed version of jqueury - jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API (link)
api.js	Exact purpose unknown, but likely used to interact with an API
ufs_web_display.js	Exact purpose unclear, but it likely related to displaying content
apstag.js	
aps_csm.js	
fbevents.js	
tag.js	

8. What percentage of websites display ads? What about deploy tracking code? You can use the lists published at https://easylist.to/ to help answer these questions. Anecdotally, what patterns do you see emerge in the types of websites that deploy a large number of ad providers? What about patterns in the sites that display a large number of ads?

   a. To find the number of sites that have display ads we used downloaded the base EasyList file then used a python library named adblockparser to decipher the rules and determine weather url's would be blocked by the rule base. We the checked if sites resource url would be blocked and if any were then the site contains ads. We found that **45.86%** of sites displayed ads.

b. We used a similar methodology for finding sites that deploy tracking code, but we used the easyprivacy file rather then the easylist file since this one “removes all forms of tracking from the internet.” We found that **63.97%** of sites track code.

c. Predictably,sites that have a large number of resources are more likely to have both ads and tracking code. Further, many .orgs had neither ads or tracking code.

Part 4: Uncovering Web Centralization

In this section, you will investigate the networks that the web is most reliant on (versus the domains most relied on in the last question). Please share your methodology (a high-level description will suffice) for answering each question.

1. What are the top 10 ASes that are responsible for hosting the most websites in your crawl? How many websites does each top AS host? (hint: what is the IP address of the root page of the website itself?)
    a. To find the top 10 ASes that are responsible for hosting the most websites in our crawl and how many websites each top AS hosts, we wrote a script that tracks the website counts per ASN and stores encountered sites. It parses each JSONL line, extracts site URLs, checks uniqueness, and aggregates ASN hosting data. After sorting ASNs by hosted websites, it outputs the top 10 ASNs with their counts and names.

S/N	ASes	ASN NAME	No of Websites each AS Hosts
1	13335	CLOUDFLARENET	373
2	16509	AMAZON-02	95
3	16625	AKAMAI-AS	54
4	15169	GOOGLE	45
5	20940	AKAMAI-ASN1	41
6	54113	FASTLY	37
7	396982	GOOGLE-PRIVATE-CLOUD	19
8	39572	ADVANCEDHPSTERS-AS	18
9	14907	WIKIMEDIA	16
10	29789	REFLECTED	15

2. How many websites are hosted by ASN 13335? What AS is this, and what service do they provide?
    a. There are 373 websites hosted by ASN 13335. To find this We wrote a script to identify ASN 13335 and gather details from the provided JSONL file, we adapted the Python script to process the data systematically. It checks if ASN 13335 exists in the data, retrieves its website count and name, and prints this information.
    b. ASN 13335 IS CLOUDFLARENET and the service it provides is content delivery network services.
3. On average, how many unique ASes host resources (both first party and third party) per website in your crawl? What ten websites rely on the largest number of unique ASes, and what websites rely on the least unique ASes?

    High-Level Description of our code: We wrote a python script that iterates through each line of the JSONL file, extracting website URLs and ASNs from the resources. It checks if resources are present and not empty for each entry. If resources exist, it extracts ASNs from the site's resources, excluding AS: None. For each site, it updates a dictionary (**unique_as_counts**) to store the count of unique ASes hosting resources. Additionally, it maintains a set (**unique_as_set**) to store unique ASes across all sites, ensuring each AS is counted only once. After processing all entries, the script calculates the number of unique ASes that host resources (both first and third party) and the average number of unique ASes hosting resources per site. It sorts the sites based on the number of unique ASes they rely on, both in descending order (for sites relying on the largest number of unique ASes) and ascending order (for sites relying on the least number of unique ASes). Finally, the script prints out the top 10 sites relying on the largest and least number of unique ASes, excluding AS: None. It also prints the total number of unique ASes hosting resources and the average number of unique ASes per site.

    a. The number of unique Ases that host resources (both first and third party) is 358. The average number of unique ASes hosting resources per site is 0.37.
    b. The ten websites that rely on the largest number of Unique ASes are:

S/N	Website	No of Unique Ases they rely
1	https://vnexpress.net	66
2	https://www.merriam-webster.com	57
3	https://www.ynet.co.il	52
4	https://www.goal.com	51

5	https://www.nexusmods.com	49
6	https://www.dailymail.co.uk	48
7	https://timesofindia.indiatimes.com	48
8	https://weather.com	47
9	https://www.geny.com	47
10	https://www.w3schools.com	46

c. The ten websites that rely on the least Unique ASes are:

S/N	Website	No of Unique ASes they rely
1	https://schools.kundelik.kz	1
2	https://www.starfall.com	1
3	https://scholar.google.com	1
4	https://support.google.com	1
5	https://kim.ibomma.lin	1
6	https://chromewebstore.google.com	1
7	https://www.latamairlines.com	1
8	https://www.gov.uk	1
9	https://translate.google.com	1
10	https://play.google.com	1

4. What are the top 10 third-party ASes that websites rely on to serve resources, and what fraction of websites do they appear on? Consider a third-party AS to be one that is not equivalent to the AS that hosts the website.
    a. For our Python script, we implemented to solve this question, after extracting the necessary information from the resource in each line of the JSONL file and processing all entries, it calculates the fraction of appearances of each third-party AS by dividing the count of appearances by the total number of sites. It sorts the third-party ASes based on their appearance fraction in descending order and

prints out the top 10 third-party ASes along with their appearance fraction and corresponding AS names making sure to exclude AS: None. The top 10 third party ASes that websites rely on to serve resources and the fraction of websites they appear on are:

S/N	ASN	AS NAME	Fraction of websites they appear on
1	15169	GOOGLE	51.40%
2	16509	AMAZON-02	24.30%
3	13335	CLOUDFLARENET	21.70%
4	54113	FASTLY	21.20%
5	396982	GOOGLE-PRIVATE-CLOUD	17.20%
6	16625	AKAMAI-AS	15.90%
7	20940	AKAMAI-ASN1	14.90%
8	14618	AMAZON-AES	13.90%
9	60068	CDN77	12.90%
10	16276	OVH	12.00%

5. What are the potential implications of having websites increasingly rely on a small number of ASes to serve and host content?
    a. Depending heavily on just a few Autonomous Systems (ASes) for hosting and serving websites can lead to various consequences. The security implications of relying on a small number of ASes to host websites could be catastrophic. These ASes would become prime targets for malicious actors aiming to disrupt internet services or compromise sensitive data. Imagine if someone pulled off a hack on one of these big hosting companies, like Cloudflare, that hosts a ton of websites, say 373 of them. It'd be chaos! All those websites could go down, or worse, get their data stolen which makes the big hosting companies a single point of failure. Also if something goes wrong with one of these big hosting services, it's like a domino effect, messing up everything else connected to it.