Historical Robots.txt Data Collection

Vajinder Kaur

April 1, 2025

1 Introduction

Objective

This research project analyzes how online publishers have modified their robots.txt files over time in response to the rise of generative AI-based search engines. Your task is to systematically collect and analyze historical robots.txt data from selected publishers using the Internet Archive's Wayback Machine.

Scope

Timeframe: June 2023 – March 2025.
Publishers: We have two groups of target domains as follows:

Group A

- time.com
- spiegel.de
- fortune.com
- entrepreneur.com
- latimes.com
- independent.co.uk
- adweek.com
- blavity.com
- prisa.com
- rtl.de

Group B

- nytimes.com
- wsj.com
- washingtonpost.com
- bbc.com
- theguardian.com
- bloomberg.com
- businessinsider.com
- vox.com
- wired.com
- forbes.com

2 Methodology

2.1 Data Collection

Source: Data was collected from the Internet Archive's Wayback Machine using the Wayback CDX Server API, which enables complex querying, filtering, and analysis of archived snapshots.

Tool Used: As a reference for this project, we utilized the open-source GitHub repository: Historical Robots.txt Parser. However, our final implementation deviates significantly from

the original version. Using this base code, we optimized and extended the functionality with additional features, including:

- Custom timeframes for retrieval,
- Extraction of robots.txt file contents and associated domains,
- Concurrent request handling for efficiency.

Thus, the final code used in this project is entirely custom-built.

2.2 Data Cleaning and Preprocessing

- **Merging Datasets:** Data was collected in three timeframes—June 2023 to December 2023, January 2024 to July 2024, and August 2024 to March 2025—using `main.py` to optimize retrieval time. The datasets were later merged in `Preprocessing.ipynb` based on timestamps and domains.
- **Collapsing the User-Agent Column:** The `main.py` script initially collected all user-agents from each robots.txt file for a given timestamp. However, since individual user-agents were not the focus, we collapsed all user-agents into a single row per timestamp using `Preprocessing.ipynb`.
- **Timestamp Conversion:** Since most exploratory data analysis (EDA) was performed in R, timestamps were converted to the proper POSIXct format for compatibility.
- **Creating the `Blocked_All_Bots` Binary Variable:** Analyzing the data revealed that 97% of domains in Group A and 91% in Group B blocked all bots. To capture this, we introduced a new binary variable, `Blocked_All_Bots`.

Since the script only collected successful snapshots (i.e., those containing robots.txt files), there was no need to handle missing values.

3 Code

- **main.py :** Collects Historical Robots.txt data for the given domains and timeframe. Provides a CSV with Timestamp, Robots.Txt (Content), Domains
- **blocked.py:** Extracts information from the `robots.txt` content column in the CSV produced by **main.py**. Identifies and lists crawlers that are explicitly disallowed, creating a new column `Blocked_Crawlers`.
- **Preprocessing.py:** Performs data preprocessing, including:
  - Merging datasets with the same domains but different timeframes.
  - Combining datasets with the same timeframe but different domains.
  - Collapsing multiple user-agent entries into a single row per timestamp.
- **EDA.rmd:** Conducts exploratory data analysis (EDA), including:
  - Generating frequency charts and heatmaps to visualize blocked crawlers.
  - Creating a binary variable `Blocked_All_Bots` to indicate whether all bots were blocked.
  - Handling timestamps by converting them into a proper POSIXct format for R-based analysis.

4 Results

4.1 Exploratory Data Analysis

From the frequency graphs, we can observe that GptBot is blocked most frequently by publishers in both groups. Google-Extended is the second most blocked crawler in Group A, but the difference between GptBot and Google-Extended is more than twice.

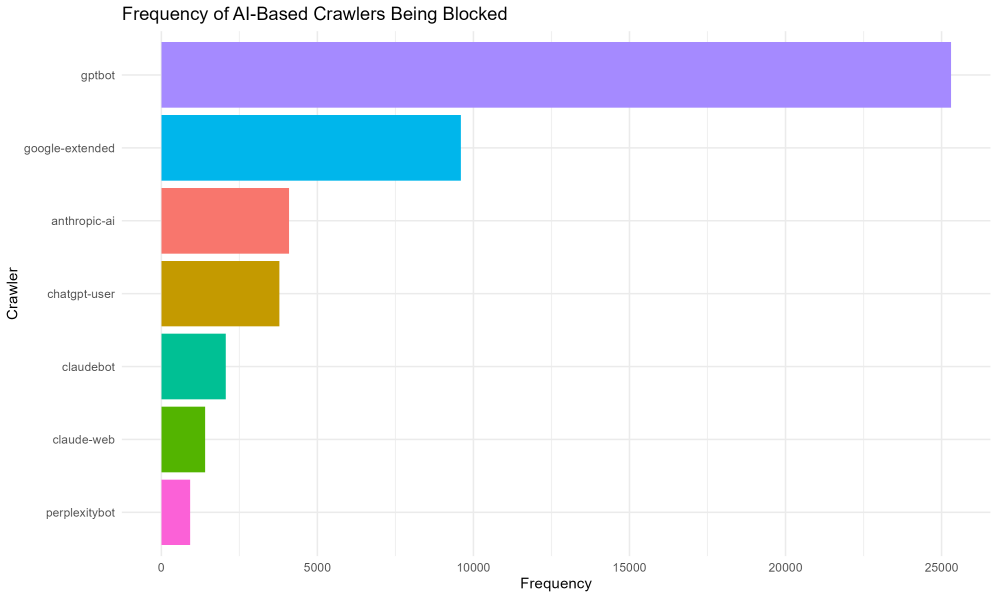

Figure 1: Frequency_GroupA

Group B Publishers, in numbers, blocks AI Crawlers more than Group A (more than double). In fact, it blocks all the Crawlers by a good margin as compared to the Group A publishers (cumulative).

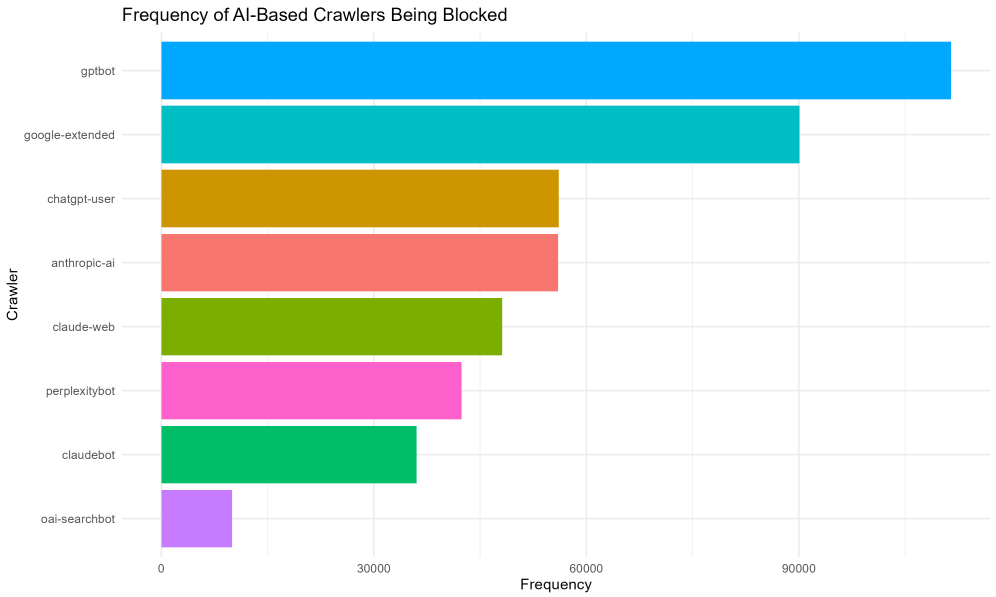

Figure 2: Frequency_GroupB

This Heatmap (Figure 3) accounts for the timeframe as well. Out of the 10 Publishers in Group A, we have only 7 here, and even in these, a couple of them started blocking the AI crawlers quite late (adweek.com, rtl.de,time.com). Around the end of 2024, we can see that there's been less blockage toward AI crawlers, which is surprising.

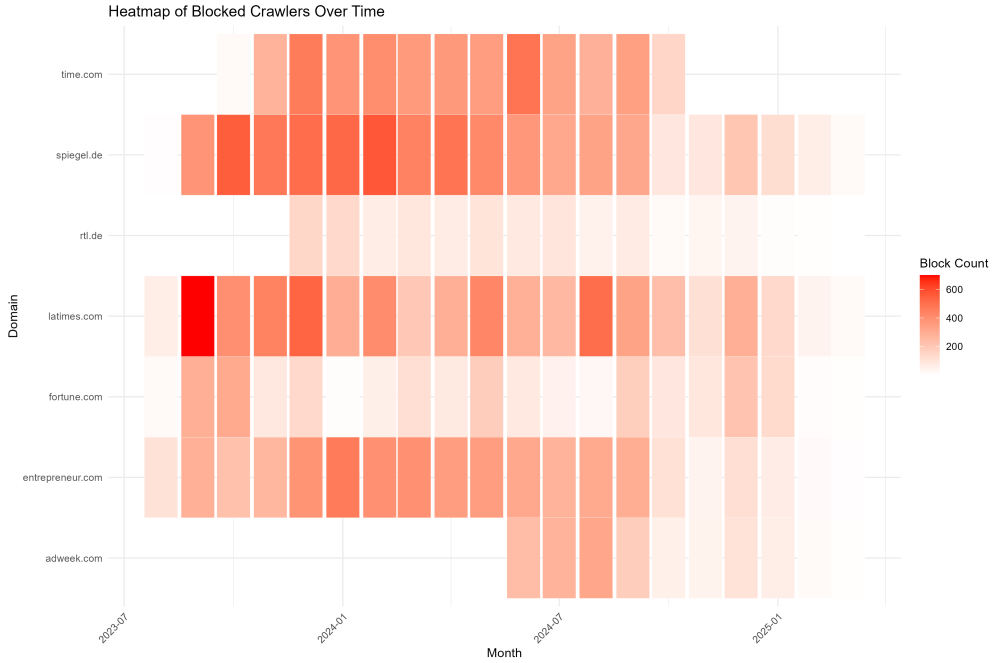

Figure 3: GroupA_Heatmap_DomainTime

This heatmap (Figure 4) not only accounts for the time and domains but also individual Crawlers. Again, we can see GptBot is blocked by almost all of the domains at some point, whereas other bots are more or less blocked by just 2 or 3 domains.

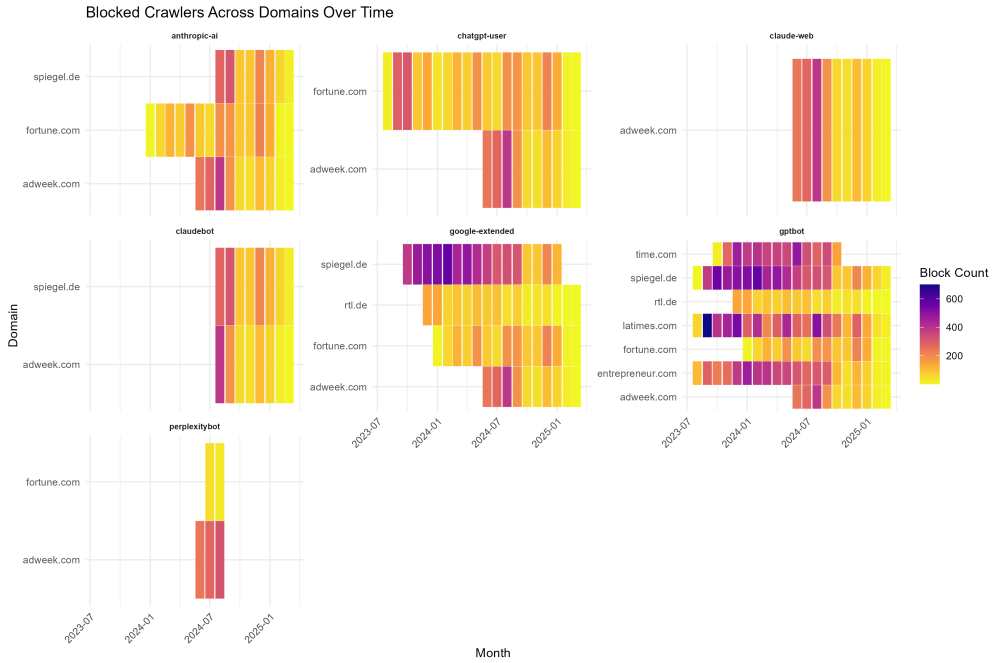

Figure 4: GroupA_Heatmap_DomainTimeBots

This shows Group A publishers aren't that vigilant towards bots other than GptBot

This heatmap (Figure 5) appears fainter in color compared to Group A's, but consistency and numbers remain key. All domains in Group B block AI crawlers, and they have been doing so from an early stage. However, similar to Group A, we observe a decline in blocked instances by the end of 2024. We specifically examined *washingtonpost.com* as it stands out for having significantly fewer AI crawler blocks after the initial period.

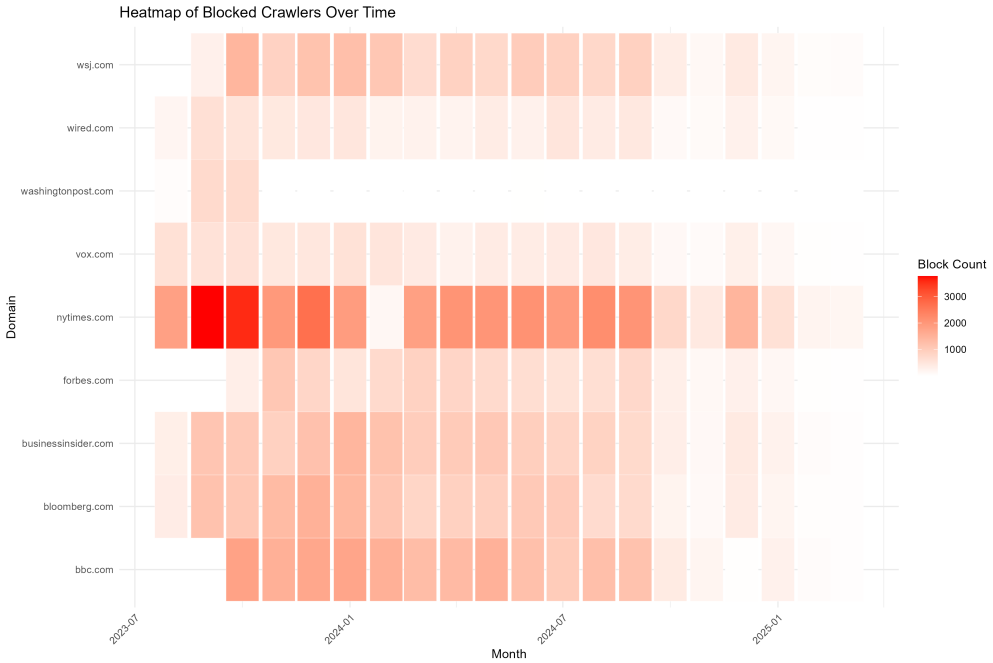

Figure 5: GroupB_Heatmap_DomainTime

The heatmap below confirms that all domains in Group B block AI crawlers.

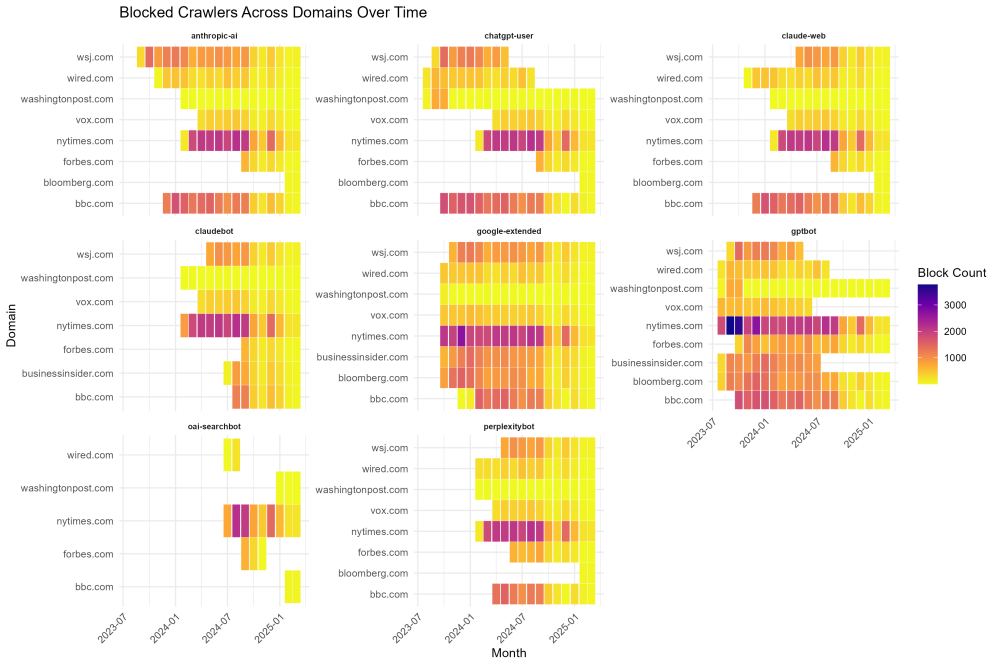

Figure 6: GroupB_Heatmap_DomainTimeBots

While some started blocking later, we observed new AI crawlers being restricted in 2025. For instance, *Bloomberg.com* blocks *anthropic-ai, chatgpt-user, claude-web,* and *perplexitybot*, while *bbc.com* and *washingtonpost.com* block *oai-searchbot*.

5 Challenges

Group A had significantly less data compared to Group B. I attempted to remove the filtering threshold of 200 to retrieve more data, but this resulted in more NAs than the current script. Many snapshots of `robots.txt` files are missing—I can verify their existence manually, but I am unable to retrieve them using the CDX API.

Another challenge was the slow response time of the Wayback Machine. On some evenings, the server was down, making data retrieval difficult. Additionally, sending too many requests or attempting to speed up the process led to temporary blocks. As a result, the only viable approach was to collect the data patiently.

Data for theguardian.com couldn't be collected with the current script, as it is frequently redirected to /us or /international for snapshots and returns to the original URL path for the robots.txt. This makes it difficult for the current script to handle. However, work is ongoing to address these kinds of cases.