The True Drivers of Web Carbon Footprints: A Data-Driven Study of Page Size, JavaScript, Cacheability, and Client-Side Processing

Shailesh Bolduc
shaileshbolduc@uchicago.edu
University of Chicago
Chicago, Illinois, USA

Jake Soublet
jpsoublet@uchicago.edu
University of Chicago
Chicago, Illinois, USA

Abstract

As the scale of global web traffic continues to grow, even small inefficiencies in webpage design can result in significant carbon emissions. It is therefore essential to understand which structural or performance related characteristics meaningfully influence per-page emissions in order to promote and guide sustainable web development. This study analyzed a dataset of 50 highly visited websites using Google Lighthouse performance metrics, network logs, and WebsiteCarbon emissions estimates in order to examine how page size, Total Blocking Time (TBT), JavaScript, caching potential, and network request quantity contribute to estimated operational carbon.

Supported by three ordinary least squares regression models, a performance model for TBT, a direct emissions model, and a mediation model, it was confirmed that the total bytes transferred is the strongest and most consistent predictor of estimated carbon emissions. JavaScript load strongly predicts TBT, showing that large scripts increase client side computational effort. However, TBT does not independently predict emissions once total page weight is accounted for. This indicates that the dominant mechanism that JavaScript affects emissions through is increased data transfer rather than CPU load. Cacheability ratio showed no statistically significant relationship with emissions.

These findings suggest that sustainable web design practices should prioritize reducing total page weight above optimizing performance metrics. The paper concludes by discussing practical design implications and recommending future work focused on measuring real-world caching behavior and repeated-visit scenarios.

Keywords

Web Design, Sustainability, Data Transfer, Caching, Operational Carbon, JavaScript, Blocking Time, HTTP Requests, Networks, Carbon Emissions, Modeling, Ordinary Least Squares, Significance Testing

ACM Reference Format:
Shailesh Bolduc and Jake Soublet. 2026. The True Drivers of Web Carbon Footprints: A Data-Driven Study of Page Size, JavaScript, Cacheability, and Client-Side Processing. , 9 pages.

1 Introduction

The environmental impact of the web is largely shaped by the scale at which digital services operate. Popular websites may receive millions or billions of monthly visitors, meaning that seemingly minuscule inefficiencies can result in a large increase in carbon emissions. As the use and scale of the internet continues to rise, understanding how specific webpage design choices and technical components influence the per-page energy use and carbon emission has become essential for creating a more sustainable web. Despite growing awareness on digital sustainability, empirical studies on which web design choices meaningfully affect operational carbon remain limited. This study aims to address this knowledge gap by taking a more holistic approach at analyzing performance audits, network logs, and modeled emission estimates of widely-visited websites.

1.1 Background and Motivation

The energy cost of loading a webpage is driven by two primary factors: the physical movement of data over our global network infrastructure and the resulting computational work that must be done to render and execute the webpage on user devices. Prior research on this topic seems to suggest that data transfer in particular plays a huge role in operational carbon emissions due to the energy demands of data centers and network hops. While infrastructural operational emissions seem unavoidable, modern websites often heavily incorporate JavaScript, third-party trackers, ads, and client side rendering. The heavy use of these items in modern webpages raises questions about how structural design decisions can influence both performance and emissions.

At the same time these design decisions directly affect performance metrics. Take the metric Total Blocking Time (TBT) which serves as a proxy for computational effort. If a JavaScript heavy design leads to increased TBT and in turn CPU workload, it could in turn provide another pathway through which webpages could generate higher emissions through the increased energy needed for the device. Taking into account these factors, it is still unclear whether this dimension of performance independently contributes to carbon emissions once page weight is accounted for.

Motivated by these uncertainties, this study seeks to examine how JavaScript load, caching potential, network requests, and page weight potentially contribute to the operational emissions of real world webpages. By grounding our analysis in our dataset of some of the top visited websites, this study aims to provide relevant insights that could help to inform sustainable design practices at scale.

1.2 Research Questions and Hypotheses

To guide the analysis, the research focuses on three central research questions:

(1) **Which webpage characteristics most strongly predict per-page operational carbon emissions?**
(2) **Does heavy use of JavaScript increase emissions primarily through increased CPU workload (Total Blocking Time), increased data transfer, or both?**
(3) **Does a higher cacheability ratio — representing the share of bytes eligible for browser caching — meaningfully reduce estimated emissions?**
Based on existing knowledge of the web and the current literature, the following hypotheses were formed:

- **H1:** Total bytes transferred will be the strongest and most statistically significant predictor of carbon emissions.
- **H2:** JavaScript-heavy pages will exhibit higher TBT, and if TBT mediates emissions, then TBT should significantly predict carbon emissions when included in a regression model.
- **H3:** Websites with a higher cacheability ratio will show reduced emissions, since more content is eligible for reuse rather than retransmission.

These hypotheses reflect intuitive understandings and expectations of the net (e.g., large pages emit more carbon) and some unresolved or unclear questions on current webpage design practices (e.g the sustainability impact of caching strategies and performance metrics).

1.3 Contributions of This Work

This study aims to make three primary contributions:

(1) **Empirical evaluation of key emission drivers.** Use Lighthouse audits, HAR network logs, and WebsiteCarbon estimates to identify which webpage features meaningfully affect operational carbon and which do not.
(2) **A mediation analysis of JavaScript and Total Blocking Time.** By testing the relationship between JavaScript load, TBT, and emissions, it is shown that JavaScript-driven CPU work does not independently predict carbon emissions once data transfer is considered—providing an evidence-based clarification to a commonly held assumption.
(3) **Practical implications for sustainable web design.** This provides actionable guidance showing that reducing total page weight is far more influential than reducing request count, improving CPU performance metrics, or altering resource composition. The limitations of cacheability were also identified as a sustainability proxy and outline directions for more accurate future measurement.

Combined, these three contributions aim to provide a much clearer understanding of how specific web-page design choices translate into real world environmental impacts and to provide a practical foundation to help build lower-emission websites at scale.

2 Methodology

2.1 Dataset Description

The data set consists of site performance and structural=related measurements collected from a sample of 50 publicly accessible websites. Each record represents a single website's primary landing page captured in Google Chrome Incognito Mode. For each site, the variables were obtained from publicly available tools that provided network activity, breakdown of a website's resources, and performance evaluations.

The data include both raw and derived variables related to:

- Size of resource transfer (e.g., total bytes transferred)
- Types of resources and their composition of the page size (e.g., JavaScript bytes and fraction of page attributed to JavaScript)
- Number of requests (e.g., number of HTTP requests)
- Performance characteristics (e.g., Total Blocking Time)
- Cacheability characteristics (e.g. number of bytes that can be cached)
- Estimated carbon emissions (in grams and on log(grams) scale)

The final data set used for the analysis contained 45 observations after cleaning and filtering. This size is consistent with a general exploratory data set for a typical data-focused study where manual auditing is the primary collection method. This data set is composed of a single snapshot per website and does not include multiple observations per website or time-series data.

Since websites naturally vary in how they function, the way they are structured, what they are used for, and how often they are used, the data set represents a wide variety of web design patterns which allowed for investigation of general trends rather than any domain-specific conclusions.

2.2 Data Collection

Data were collected via Chrome DevTools Lighthouse, Chrome DevTools Network logs, and WebsiteCarbon (**websitecarbon**). Lighthouse performs audits of a webpage's performance, accessibility, SEO, best practices, and progressive web app features and generates a report with scores, numerical outputs, and categorical identifications of various features. These reports, exported as JSON files, were used in this study to obtain the performance and structural characteristics of the web pages.

The Network logs monitor the data coming into and being emitted from a webpage. These logs, which were exported as HAR files for analysis, were used to obtain information about cache headers. This allowed for derivation of the cacheability ratios (CR) of the websites.

The website SimilarWeb (**similarweb**) was used to obtain a list of the top 50 most visited websites which made up the majority of the dataset. However, if accurate data could not be obtained from one of these websites, a random other website was chosen to substitute it.

2.2.1 *Lighthouse Report Generation.*

(1) Open Google Chrome in Incognito Mode to ensure Chrome runs in a clean slate free of browser extension or user profile interference.
(2) Visit the primary landing page for each site in the aforementioned list from Similar Web.

(3) Navigate to the Lighthouse Tab in the Chrome DevTools interface and select all default settings except for device type, which should be set to "Desktop."
(4) Generate a report.
(5) Save report as JSON.
(6) Enter the same URL as was used for the Lighthouse report into Website Carbon.
    - If WebsiteCarbon cannot generate a value for the site, select a random website not on the SimilarWeb list and begin from step 2.
(7) Enter a new entry into a separate table with the website name being the same as the associated Lighthouse report's "requestedUrl" tag and the $co2\_g$ value being the output of Website Carbon.

2.2.2 Network Log Collection.

(1) Open Google Chrome in Incognito Mode to ensure Chrome runs in a clean slate free of browser extension or user profile interference.
(2) Visit the primary landing page for each site in the aforementioned list from Similar Web.
(3) Navigate to the Network tab in the Chrome DevTools interface and select "Preserve Logs" and "Disable Cache."
    - Cache is disabled to get an accurate reading of all resources on a page that have the potential to be cached without worrying about differing behavior between visits or networks the data is collected on.
(4) Select the download button and download the HAR file of the network logs.

2.2.3 Extracting then Combining the Data.

Using Python scripts that relied on the Pandas and JSON libraries, the appropriate fields corresponding to variables in the data set were extracted from the JSON and HAR files. The scripts can be made available upon request. Data manipulation, specifically unit standardization and fractional variables, was performed in the scripts.

For cacheability ratio, any item that had a Cache-Control, ETag, Expires, or Last-Modified header was considered cacheable. Otherwise, the element was not considered cacheable.

The data were then combined on a union of the two resulting lists on the website name.

2.3 Variable Definitions

2.3.1 Dependent Variable: Estimated CO_2.

In the data set, $co2\_g$ is the estimated carbon emissions associated with loading a website in grams while log_co2 is simply the natural logarithm of this field. The estimate is derived from Version 4 of the Sustainable Web Design Model which models approximate emissions based on data movement through a network and supporting infrastructure. The log transformation was applied to reduce skewness and stabilize variance which improves suitability for linear modeling and makes results in log-log modeling interpretable as elasticities.

2.3.2 Primary Predictors.

- $total\_bytes$ (log-transformed as $np.log1p(total\_bytes)$): The total number of bytes transferred when loading the webpage, including all components like HTML, CSS, images, JavaScript, fonts, and third-party scripts. Since the distribution of bytes was highly right-skewed, a log1p transformation was used to normalize the distribution while allowing for values that were initially zero with a plus-one change.
- $requests$ (log-transformed as $np.log1p(requests)$): The total number of requests sent over a network by the website. This variable captures natural complexity of a site as well as structural characteristics. Similar to the other variables mentioned, this variable was log-transformed to reduce skew between records.
- js_frac The total fraction of the page size attributed to JavaScript files. This variable was often kept separate from js_bytes which measured the total number of bytes in a page attributed to JavaScript due to obvious collinearity.
- img_frac: The total fraction of the page size attributed to image files. This variable was often kept separate from img_bytes which measured the total number of bytes in a page attributed to images due to obvious collinearity. This captures visual-heavy designs in websites.

2.3.3 Caching Proxy: Cacheability Ratio.

$cacheability\_ratio$ is a metric that represents the proportion of resources on a page than have the potential to be cached in persistent browser caches. The potential for caching was obtained by identifying elements of a webpage that included cache-friendly HTTP headers such as Cache-Control, E-Tag, or Expires.

This variable was used as a proxy for caching behavior under the assumption that elements with valid caching instructions are more likely to be reused and stored across sessions which would reduce redundant data transfer. Furthermore, given measurement constraints of this study and protection of privacy in Chrome, identifying the true contents of persistent caches was limited. So, $cacheability\_ratio$ plays a role in this study as a scalable and reproducible alternative.

2.3.4 Performance Proxy: Total Blocking Time (TBT).

tbt_ms is a metric that represents the amount of time the main thread for the webpage was blocked while loading content, while log_tbt was the log-transformed version of this field. TBT is a commonly used proxy for CPU effort in browser environments since it is influenced primarily by JavaScript execution time. It captures how long a user's browser sits idle due to tasks that add to the computational workload of loading and running the site on a client browser. Once again, the log transformation corrects for skew.

2.4 Data Cleaning and Transformations

Several steps were taken to ensure the already-limited data were reliable prior to analysis:

(1) **Remove incomplete data**: Any websites that were missing core metrics such as carbon emissions, total page size, or whatnot were completely removed from the data set.

(2) **Handling of skewed distributions**: Key variables that exhibited high-variance right-skewed distributions were transformed under NumPy's $log1p()$ function to reduce extreme values, improve linearity in regression, normalize scale, and meet assumptions of normally distributed residuals.
(3) **Standardization of units**: All size-related variables had versions of the fields in both larger and smaller units (e.g., megabytes and bytes) to ensure consistency across observations and regressions.
(4) **Validation of Ratios**: Fractional variables (such as $frac\_js$ were created to normalize the scale at which structural components were studied to understand the relationship between proportional composition and dependent variables.

2.5 Modeling Framework

The main analytical goal of this study was to identify website characteristics that are most strongly associated with operational carbon emissions. Therefore, three linear regression models were constructed to analyze both direct and indirect relationships between content type, performance indicators, content size, and carbon emissions.

All models were built using ordinary least squares (OLS) regression with the Statsmodels Python library. (**seabold2010statsmodels**) The dependent variable varied between models, but was the logarithm of estimated CO_2 emissions for the primary models. Independent variables were selected based on the initial hypothesis of the study. They represented three categories: content size, content structure, and loading behavior.

2.5.1 *Performance Model (TBT).*

The first model fit was created with Total Blocking Time (log_tbt) as the dependent variable. The aim of this regression was the test whether JavaScript or other types of assets had a significant impact on the processing demand, as predicted by the initial research and hypothesis, by increasing the work done by the CPU for the browser. The motivation for this model came from existing literature as well as the correlation between different JavaScript, emissions, and total blocking time seen during initial exploration:

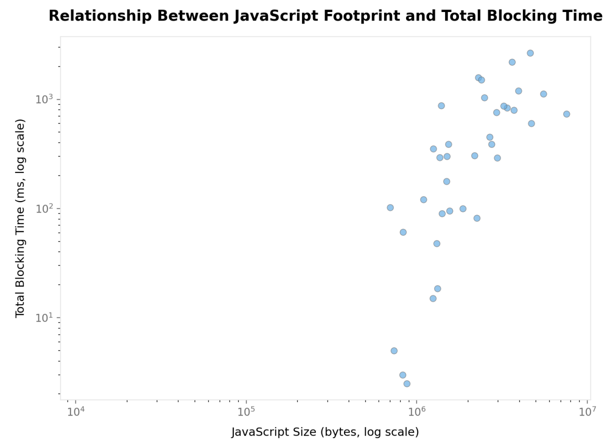

Figure 1: A scatterplot showing the relationship between bytes of JavaScript and total blocking time on a log-log scale. Shows a highly positive correlation. $R^2 = .72$**.**

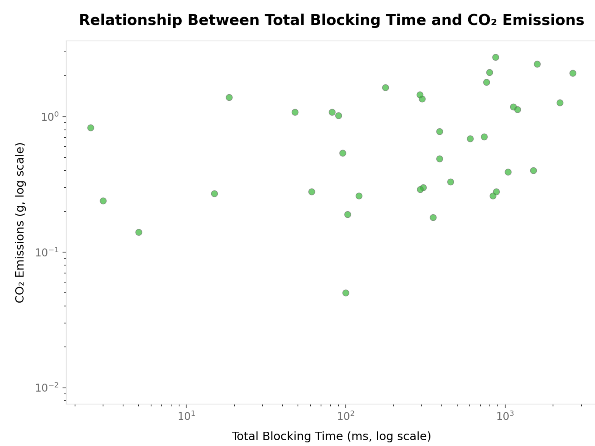

Figure 2: A scatterplot showing the relationship between total blocking time and CO_2 **emissions on a log-log scale. Shows a somewhat positive correlation.** $R^2 = .52$**.**

The independent variables included:

- Log-transformed JavaScript bytes
- Log-transformed image bytes
- Log-transformed CSS bytes
- Log-transformed total bytes
- Cacheability ratio
- JavaScript file count

This model was used to assess how much certain structural features of a webpage explain the variation in browser workload. Strong effects here would support the conceptual chain of "JavaScript increases CPU work increases energy use."

2.5.2 *Main Emissions Model.*

This model was created to directly measure the effects of chosen webpage characteristics on estimated total emissions rather than through the mediating variable of TBT. The model is as follows:

$$
\begin{aligned}
log(CO_2) = &\beta_0 + \beta_1 log(1 + total\_bytes) \\
&+ \beta_2 cacheability\_ratio + \beta_3 js\_frac \\
&+ \beta_4 img\_frac + \beta_5 log(1 + requests)
\end{aligned}
$$

This design captures:

- Data transfer volume via $total\_bytes$
- Caching capability via $cacheability\_ratio$
- Content type via js_frac amd img_frac
- Structural complexity via $requests$

This model mirrors other findings in sustainable web design literature that data transfer is a large contributor to energy use and allows for the types of data being transferred to explain variation in emissions directly rather than through blocking time. Log-transformations were applied to stabilize variance and make coefficients interpretable as elasticities.

2.5.3 Mediation Model.
To examine whether performance (proxied with TBT) mediates the relationship between content and emissions, this third model was built:

$$
\begin{aligned}
log(CO_2) = &\beta_0 + \beta_1 log(1 + total\_bytes) \\
&+ \beta_2 cacheability\_ratio + \beta_3 js\_frac \\
&+ \beta_4 log\_tbt
\end{aligned}
$$

This model tested whether TBT actually explains a significant portion of the relationship between size attributable to JavaScript and emissions or if there is just covariance. The interpretation of results from this model was mainly based on the following two criteria:

- Does log_tbt significantly predict log_co2?
- Does including log_tbt in the model significantly affect the coefficient of js_frac?

Finding answers to these questions aligns with typical systems where multiple variables could be interacting to contribute to overall environmental impact. However, the limitations of this mediation model are that it is an exploratory decomposition of overlapping variation and explanatory power, not a formal causal mediation analysis which would often include a few other regressions or steps to make formal conclusions about causation.

3 Results

3.1 Descriptive Overview

The final dataset consisted of 45 websites, each with information on page size, size of assets, performance metrics, and fractional variables. The main outcome variable analyzed was the log-transformed estimates for carbon dioxide emissions (log_co2). The primary predictors were page size (log total bytes), compositional features (JavaScript and image bytes/fractional components), request count, cacheability ratio, and total blocking time.

Across the entire dataset, there was not a wide spread of cacheability ratios. Most values exceeded .9. This indicates that most web assets were eligible for caching based on HTTP headers; however, this does not guarantee reuse or cache storage in practice. There were some extreme cases, like YouTube with a .3 CR since large video files do not get cached, but most were above .9 and .8.

3.2 Regression Results: TBT Model

This model regressed log-transformed Total Blocking Time onto various structural and asset-detailing predictors and explained a substantial proportion of variation: $R^2 = .713$, Adjusted $R^2 = .668, p < .001$.

The key findings were that bytes of JavaScript used in the page are a strong and statistically significant predictor of TBT with a coefficient of 1.346 and a p-value less than .001. This provides actionable evidence that larger JavaScript loads increase browser blocking and execution time.

The variables for image bytes, CSS bytes, total bytes, cacheability ratio, and JS file count were all not statistically significant in this model.

This supports the conceptual hypothesis of this study that more JavaScript on a website drives higher browser processing load. However, to complete the chain, it was then tested whether total blocking time was really what caused higher emissions.

3.3 Regression Results: Emissions Model

This model regressed log-transformed carbon dioxide emissions on similar features to the previous model, but, most importantly, does not include TBT as a predictor. This model explained a moderate proportion of variance: $R^2 = .406$, Adjusted $R^2 = .330, p < .001$.

The only statistically significant predictor was $log(total\_bytes)$ with a coefficient of 0.2168 and a p-value of .01. This indicates that overall page size is the dominant driver of estimated emissions.

All other predictors were not significant at any standard significance level.

This suggests that emissions are largely driven by total data transfer. This is consistent with prior research on the topic and remains aligned with the initial hypothesis of the study.

3.4 Mediation Analysis Findings

To explore whether performance, specifically TBT, explains part of the relationship between content and emissions, a third model was estimated which added log(TBT) to the carbon model. The model fit was similar here, but slightly improved: $R^2 = .419$, Adjusted $R^2 = .361, p < .001$.

The most interesting outcomes of this model were that log_tbt was not statistically significant (coefficient 0.0382, p=.202), the effect of $log(total\_bytes)$ remained relatively significant (coefficient .1299, p=0.059), and the coefficient on $frac\_js$ became much smaller and changed direction slightly from the main model (coefficient $.2477 \rightarrow -0.073$, $p = .318 \rightarrow .755$).

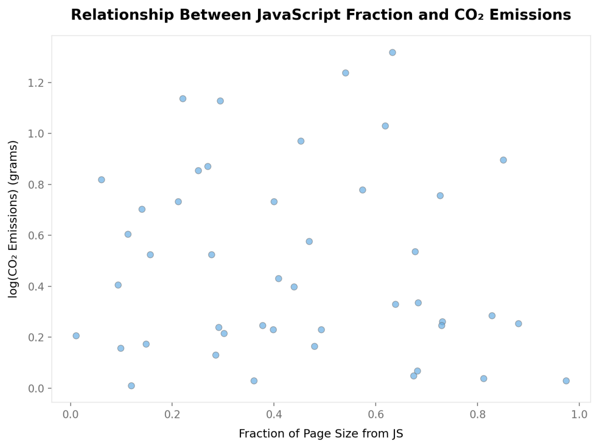

Figure 3: A scatterplot showing the relationship between the log-emissions values and the fraction of page size attributable to JavaScript. The relationship is clearly extremely weak. This follows from the analysis because it is the size of the page, not merely the composition details that drive emissions.

This suggests that while JavaScript strongly predicts TBT, TBT does not independently predict CO_2 once total page size is present. So, while JavaScript increases both blocking time (and therefore CPU workload) and data transfer, the data transfer portion of this is the dominant path of increasing emissions. Performance does co-vary with emissions, but does not independently drive them.

4 Discussion

4.1 Interpretation of Key Drivers

Across all three models, total bytes transferred emerged as the strongest and most consistent predictor of estimated emissions. The main emissions model shows that page weight alone significantly explains variation in log-emissions while other features such as request count and cacheability ratio did not show any meaningful explanatory power. These findings suggest that the main determinant of operational carbon emissions for webpages is the sheer volume of data that must be moved through the network.

This finding aligns with the current literature on web sustainability which emphasizes that data transfer through network infrastructure is extremely energy intensive and thus is constitutes the dominant share of operational carbon. The fact that this relationship remained true even after controlling for other performance and structural variables further reinforces the importance of reducing total page weight in order to lower emissions.

4.2 Role of JavaScript and Mediation

The results of the TBT model confirm that JavaScript meaningfully increases client-side CPU workload. This is evidenced by its large, statistically significant coefficient ($p < .001$, $\beta = 1.346$). Pages that heavily rely on JavaScript impose more work on the CPU's main thread leading to longer TBT.

While the above is true, the mediation model demonstrated that this increased CPU workload does not independently predict carbon emissions once total bytes are included in the model. The coefficient on log(TBT) was statistically insignificant which suggests that although JavaScript does affect performance, its contribution to overall emissions comes overwhelmingly from increased data transfer rather than CPU power draw.

Overall the mediation analysis sheds light on the mistaken commonly held assumption that JavaScript's increased CPU workload is a major driver of emissions. In practice, the analysis on this dataset appears to show that the browser's energy use is a minor factor compared to the energy required to transmit additional bytes of data across network infrastructure.

4.3 Implications for Sustainable Design

Taken together, these findings suggest that sustainable web design strategies should mainly focus on reducing total page weight, rather than solely optimizing performance metrics such as TBT. Such strategy approaches such as minimizing heavy JavaScript use, compressing assets, lowering the use of scripts and third party trackers are likely to delivery the most substantial reductions in emissions. In contrast, metrics associated with performance like Total Blocking Time did not idependently explain variation in emissions once total page size was accounted for. The lack of significant for request count also suggests that the amount or number of assets that must be loaded is less important than the size of assets highlighting that total byte volume is the more reliable variable to optimize.

This has several implications for how web designers approach creating their websites in an emissions-aware manner:

(1) **Reduce total byte volume as the primary strategy.** The strong effect of total bytes implies that the most impactful sustainability approaches are the ones that decrease the amount of data transmitted on an initial page load. Compressing images, bundling resources, removing unused code, and removing unnecessary assets are likely to produce the most meaningful reductions in emissions. This is especially true for JavaScript bundles as supported by the findings of this analysis.

(2) **Limit heavy JavaScript, not because of CPU load, but because it increases page size.** While JS bytes strongly predicted TBT, TBT was not what ended up significantly driving emissions. This suggested that the sustainability cost of JS arises not from CPU work, but because JS bundles increase the overall data transferred. So, design choices that reduce these payloads of JS will likely yield emissions benefits even if CPU runtime only modestly improves.

(3) **Request count matters less than asset size.** The absence of a significant effect for request count show that the number of files being loaded and number of network requests they send is far less important than how large those files are. Modern HTTP protocols tend to mitigate these overheads which aligns with the finding that emissions are driven by byte volume rather than count of resources or request count. Now, operational carbon does not account for embodied carbon, but effects are roughly equal at a holistic glance.

(4) **Prioritize structural changes over micro-optimizations.** Small efficiency changes could have emissions savings, but are unlikely to do so from the results of this study. If micro-optimizations are a concern, then changing stacks entirely, such as to React and TypeScript, might be more desirable then figuring out how to optimize JS files and increase the file size for these small payoffs. Especially when sustainability is a priority. Selecting lighter libraries, designing efficient media pipelines, and critically evaluating third-party tracks and scripts will help a lot, too.

Furthermore, the lack of a strong effect from cacheability ratio indicates that simply enabling cache friendly headers does not create a meaningful reduction in emissions for single visit scenarios. Webpage designers may still benefit from setting long lived cache headers and optimizing caching on repeat visit but these benefits were not captured in the one time page load and analysis used in this study.

4.4 Summary of Findings

Three core findings emerged from this analysis:

(1) **Total bytes transferred is the dominant predictor** of estimated operational emissions for webpages.
(2) **JavaScript size strongly predicts Total Blocking Time** but TBT does not independently predict emissions once total bytes are included.
(3) **Cacheability ratio and request count show limited explanatory value** for single-visit emissions.

Together these three findings emphasizes that a reduction in emissions is best achieved through reducing total page weight especially for large payloads such as JavaScript or images, while optimizing performance metrics alone are unlikely to produce any real impact in reducing the environmental impact of emissions. The mediation analysis also demonstrates that JavaScript's sustainability impact is primarily one of data transfer not CPU workload.

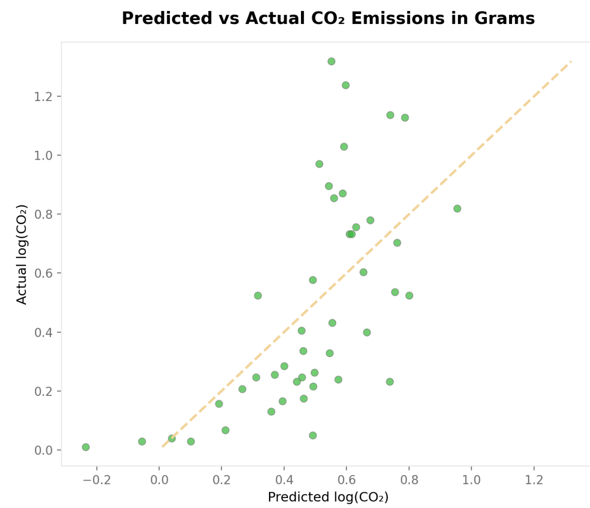

Figure 4: A scatterplot showing the relationship between the log-emissions values predicted by the main model and the actual values. Includes a dotted line that represents a perfect prediction.

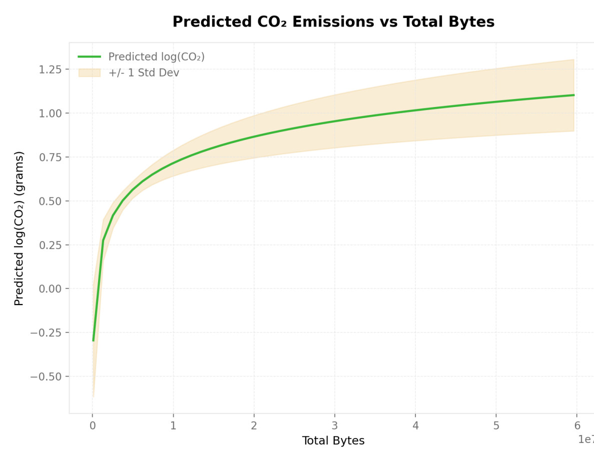

Figure 5: A line plot showing the relationship between total bytes and carbon emissions with a plus/minus 1 standard deviation range surrounding the line. This line shows roughly how much in operational carbon dioxide emissions is expected from a given page size.

5 Limitations and Future Work

5.1 Sample Size and Scope

This study is based on a dataset that takes 50 of the most visited websites and captures a single load snapshot on the sites landing page. While the sample does include many high traffic websites in order to reflect a broad range of design patterns the size of the data set limits the statistically power of regression models and restricts the ability to generalize results across the entire web. Additional because only one page per website was collected (the landing page) the analysis does not account for variability with a domain. Larger studies with a larger more diverse data set with many different types of pages would provide a stronger foundation to draw generalizable conclusions from.

While this study did not measure hardware metrics or HTLM/CSS practices, other studies attempt to address the heavy role how certain web standards such as the use of HTML, CSS, contribute to the carbon cost of rendering webpages. Instead of looking at the effect of total page weight like this study had done, one study measured physical hardware metrics such as CPU/GPU/RAM usage for different HTML and CSS features to gauge their energy footprint (Dawson, 2023). Their results showed that certain HTML or CSS constructs release more carbon than others suggesting that certain design choices, not only page size matters for a sites sustainability.

5.2 Browser and Environment Constraints

All data was collected using Google Chrome in Incognito mode on two hardware devices under the same network conditions. This somewhat restricts the validity of the measurements since real world users experience inconsistent device performance, cached resources, different browsers, different devices (such as mobile devices), different network performance, or varying CPU power profiles. Total blocking time itself is a browser-dependent performance metric and different devices may show different performance when it comes to executing JavaScript tasks. As a result the magnitude of performance effects may differ under different testing and data collection conditions. While there is some research into the effectiveness of caching in helping reduce energy costs and total bytes loaded, it still remains limited in scope. One empirical study (Malatova et al, Mobilesoft 2020) investigated whether caching for web apps on Android devices running Firefox browser would affect energy usage and performance. They found that there was no significant difference in energy consumption between cached and non cached loads but that cached pages loaded faster. This aligns with our findings that the increase of user device energy needed to load uncached pages did not significantly increase emissions.

5.3 Caching Measurement

Cacheability ratio served as a proxy for caching behavior but it does not reflect true real world cache performance (such as cache hits and misses) in repeated visit scenarios. A resource marked as cacheable with a cache header may not actually be stored in a users cache due to eviction, storage limits, user settings or device constraints. Similarly a resource without a cache header may still be cached by some browsers. Because WebsiteCarbon models only first-load emissions of a page, potential reductions from cache reuse on subsequent visits cannot be captured. Future research could employ a controlled multi-load experiment or use browser tools capable of analyzing persistent cache contents to quantify caching real world caching behavior and the effect it has on emissions.

5.4 Role of Third-party Bytes

While this study found that third-party bytes was a significant contributor to total page weight it was unable to directly quantify the effect on emissions said bytes have beyond increasing the amount of data that needed to be transferred. While the scope of this study fails to meaningfully analyze these bytes, the current literature has a multitude of studies that do so. One study crawled over 2 million websites in order to quantify the extra carbon cost of these third-party trackers. Upon this the study found that" web tracking increases data transmission up to 21 percent annually" while also releasing 10.76 metric tons of carbon making it comparable to the meat, aviation, or transportation industries (Pachilakis et al, 2023). This is supported by previous older studies that seeked to understand the environmental impact of online advertising. One study found that in 2016 online advertising alone consumed between 20.38 to 282.75 TWh of energy and 11.53-159.93 million tons of emissions in order to produce said energy (Parssinen et al, 2018) While outdated it still provides a valuable look at how the web and its complexity has grown especially as online services demand more and more energy.

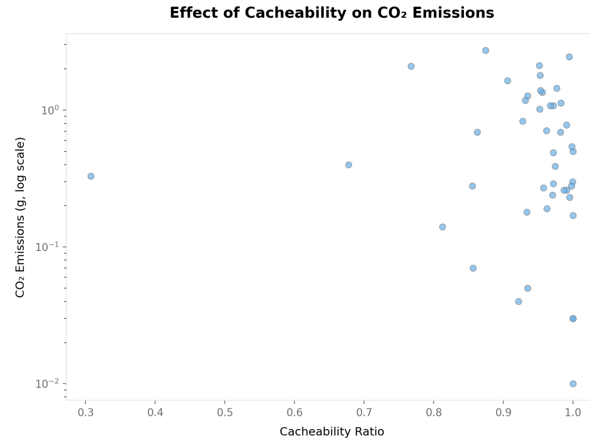

Figure 6: A scatter plot showing the relationship between the log-transformed carbon emissions and cacheability ratios. Values of the CRs are highly skewed to .8 and above, showing how the statistical significance of this variable was likely affected by the lack of variability in the data, among other factors.

Although the results of these studies use different methodologies, together they reinforce the conclusion that third party bytes are a major driver of webpage energy use. These findings are consistent with this study's observation that total page weight, which by virtue includes third party bytes, strong influences and increases emissions. Reducing the amount of such bytes would in turn lower the per page emissions of webpages by decreasing the amount of data that must be transferred.

While lowering the amount of third party bytes would surely demonstrate clear environmental benefits, it poses practical challenges. Many websites and companies heavily rely on advertising and analytics to generate revenue. This monetary incentive would make it extremely difficult to convince companies to voluntarily reduce third party bytes. Due to this it is an area of future research could be how to lower the bytes needed for advertisements and analytics without reducing revenue and worsening user experiences.

6 References

References

websitecarbon

[1] Website Carbon. (2024). *Website Carbon Calculator: The tool to help decarbonise the Internet.* https://www.websitecarbon.com/
similarweb
[2] SimilarWeb. (2025). *Top websites ranking: Most visited websites in the world.* https://www.similarweb.com/top-websites/
pachilakis2023quantifying
[3] Pachilakis, M., Dambra, S., Sanchez-Rola, I., Bilge, L. (2023). *Quantifying carbon emissions due to online third-party tracking.* arXiv. https://arxiv.org/pdf/2304.00927
malavolta2020evaluating
[4] Malavolta, I., Chinnappan, K., Jasmontas, L., Gupta, S., Soltany, K. A. K. (2020). Evaluating the impact of caching on the energy consumption and performance of progressive web apps. In *Proceedings of the IEEE/ACM 7th International Conference on Mobile Software Engineering and Systems (MOBILESoft '20)* (pp. 109–119). https://doi.org/10.1145/3387905.3388593
parssinen2018environmental
[5] Pärssinen, M., Kotila, M., Cuevas, R., Phansalkar, A., Manner, J. (2018). Environmental impact assessment of online advertising. *Environmental Impact Assessment Review, 73*, 177–200. https://doi.org/10.1016/j.eiar.2018.08.004
websitesustainability2023cache
[6] Website Sustainability. (2023). *Research on the environmental impact of web caching and sustainability.* https://websitesustainability.com/cache/files/research23.pdf

7 Appendix

7.1 Full Model Outputs

7.1.1 *Total Blocking Time Model.*

Dep. Variable:	log_tbt	**R-squared:**	0.713
Model:	OLS	**Adj. R-squared:**	0.668
Method:	Least Squares	**F-statistic:**	15.72
Date:	Thu, 11 Dec 2025	**Prob (F-statistic):**	5.61e-09
Time:	22:13:01	**Log-Likelihood:**	-79.865
No. Observations:	45	**AIC:**	173.7
Df Residuals:	38	**BIC:**	186.4
Df Model:	6		
Covariance Type:	nonrobust		

	coef	**std err**	**P> |t|**	**[0.025**	**0.975]**
Intercept	-18.0655	4.566	0.000	-27.309	-8.822
np.log1p(js_bytes)	1.3460	0.305	0.000	0.728	1.964
np.log1p(img_bytes)	0.0150	0.108	0.890	-0.204	0.234
np.log1p(css_bytes)	-0.0702	0.088	0.429	-0.248	0.107
np.log1p(total_bytes)	0.3556	0.427	0.411	-0.510	1.221
cacheability_ratio	-1.5032	2.316	0.520	-6.193	3.186
js_count	0.0033	0.005	0.470	-0.006	0.012

Omnibus:	1.613	**Durbin-Watson:**	2.400
Prob(Omnibus):	0.446	**Jarque-Bera (JB):**	1.576
Skew:	-0.395	**Prob(JB):**	0.455
Kurtosis:	2.536	**Cond. No.**	1.83e+03

7.1.2 *Main CO_2 Emissions Model.*

Dep. Variable:	log_co2	**R-squared:**	0.406
Model:	OLS	**Adj. R-squared:**	0.330
Method:	Least Squares	**F-statistic:**	5.332
Date:	Thu, 11 Dec 2025	**Prob (F-statistic):**	0.000792
Time:	22:14:40	**Log-Likelihood:**	-6.2448
No. Observations:	45	**AIC:**	24.49
Df Residuals:	39	**BIC:**	35.33
Df Model:	5		
Covariance Type:	nonrobust		

	coef	**std err**	**P> |t|**	**[0.025**	**0.975]**
Intercept	-3.0085	1.002	0.005	-5.035	-0.982
np.log1p(total_bytes)	0.2168	0.080	0.010	0.055	0.379
cacheability_ratio	0.1792	0.403	0.659	-0.636	0.994
js_frac	0.2477	0.245	0.318	-0.248	0.743
img_frac	0.2177	0.249	0.388	-0.287	0.722
np.log1p(requests)	-0.0208	0.091	0.821	-0.206	0.164

Omnibus:	4.627	**Durbin-Watson:**	1.645
Prob(Omnibus):	0.099	**Jarque-Bera (JB):**	3.919
Skew:	0.721	**Prob(JB):**	0.141
Kurtosis:	3.099	**Cond. No.**	370.

7.1.3 *Mediation Model.*

Dep. Variable:	log_co2	**R-squared:**	0.419
Model:	OLS	**Adj. R-squared:**	0.361
Method:	Least Squares	**F-statistic:**	7.209
Date:	Thu, 11 Dec 2025	**Prob (F-statistic):**	0.000181
Time:	22:15:25	**Log-Likelihood:**	-5.7514
No. Observations:	45	**AIC:**	21.50
Df Residuals:	40	**BIC:**	30.54
Df Model:	4		
Covariance Type:	nonrobust		

	coef	**std err**	**P> |t|**	**[0.025**	**0.975]**
Intercept	-1.8004	1.066	0.099	-3.955	0.354
log_tbt	0.0382	0.029	0.202	-0.021	0.098
np.log1p(total_bytes)	0.1299	0.067	0.059	-0.005	0.265
cacheability_ratio	0.2173	0.392	0.582	-0.574	1.009
js_frac	-0.0736	0.234	0.755	-0.546	0.399

Omnibus:	2.636	**Durbin-Watson:**	1.639
Prob(Omnibus):	0.268	**Jarque-Bera (JB):**	2.413
Skew:	0.551	**Prob(JB):**	0.299
Kurtosis:	2.729	**Cond. No.**	397.