# MiningWeb Evolution: A Dataset of DOM Changes Across Websites

Hilal Taha
hilal.taha@uni.lu
University of Luxembourg
Luxembourg, Luxembourg

Jeongju Sohn
jeongju.sohn@knu.ac.kr
Kyungpook National University
Daegu, Republic of Korea

Mike Papadakis
michail.papadakis@uni.lu
University of Luxembourg
Luxembourg, Luxembourg

## Abstract

Continuous evolution in web applications leads to frequent structural modifications, presenting significant challenges for software maintenance and empirical analysis. Yet large-scale datasets that capture real-world, structural web evolution remain limited. This paper introduce a longitudinal, archive-derived structural dataset of the landing pages for 223 popular websites, collected from the Internet Archive's Wayback Machine. For each site, up to 1,000 time-ordered versions were collected at a minimum interval of one day. Each version's HTML was parsed into a structural representation that records, for every node, its absolute XPath position (from the document root) together with the exact capture timestamp; this yields a consistent, content-agnostic view of layout and hierarchy across time. In total the dataset contains over 246 million DOM node positions, providing a robust foundation for studies on change patterns, predictive modelling of web evolution, and benchmarking maintenance tools. By releasing this resource to the research community, we facilitate reproducible experimentation and enable novel, data-driven investigations into the long-term evolution of web applications.

## CCS Concepts

• **Software and its engineering → Maintaining software**; *Software evolution.*

## Keywords

Web evolution; Web archives; DOM structure; Open datasets; Reproducibility; Software evolution; Web testing

**ACM Reference Format:**
Hilal Taha, Jeongju Sohn, and Mike Papadakis. 2026. MiningWeb Evolution: A Dataset of DOM Changes Across Websites. In *Companion Proceedings of the ACM Web Conference 2026 (WWW Companion '26), April 13–17, 2026, Dubai, United Arab Emirates.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3774905.3795444

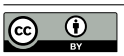
## 1 Introduction

Web applications evolve continuously, with developers frequently updating their structure, content, and design to accommodate new features, security patches, and user experience improvements. This continuous evolution poses significant challenges for automated web testing. Despite the critical importance of web evolution data for advancing test maintenance research, publicly available datasets remain scarce and limited in scope. Existing studies have primarily relied on small-scale datasets, often comprising fewer than six web applications [3, 16], or focused on narrow aspects of web testing such as end-to-end test suites from open-source projects [19]. A comprehensive survey on web testing found that research papers typically evaluated their techniques on an average of only 5.3 open-source applications, highlighting the difficulty of collecting suitable web applications for experimentation [11]. Furthermore, maintenance studies focus on events where they analyse the evolution through events (i.e. commits, deployment and test failures) rather than continuous monitoring. Studies tracking Document Object Model (DOM) changes over extended periods are rare, with most empirical work examining snapshots at discrete time points rather than continuous evolution [1, 9, 12]. This pattern is also extended to evaluating locator robustness in web testing, where studies choose two versions at two different time points for evaluating the performance rather than continuous survivability [15, 17, 20, 23].

Despite mounting evidence that structural changes in web pages are a primary cause of test fragility. Studies have shown that 67% of absolute XPath locators and 50% of id-based relative locators break after structural modifications between releases [14], with locator breakage responsible for up to 74% of GUI test failures [23]. The absence of large-scale datasets capturing real-world web evolution has hindered progress in validating robust locator-generation and test-repair techniques at scale [8]. As a result, most empirical evaluations remain limited to point-in-time analyses or small-scale benchmarks, thereby failing to reflect the challenges encountered in continuously evolving production environments. While web archives such as the Internet Archive's Wayback Machine [10] preserve billions of historical web pages spanning decades [24], these resources have been underutilised in web testing research.

To bridge this gap, we present a dataset of web evolution, specifically designed for training and evaluating locator-stability prediction models and robust locator-generation techniques. Our dataset leverages the Wayback Machine to systematically collect historical snapshots of websites. Of 1 million candidate domains, 223 remained after exclusions for non-standard markup or rendering failures and ethical exclusions (e.g., adult content). For each website, we retrieved up to 1,000 recent versions, subject to temporal constraints, resulting in a comprehensive record of DOM evolution patterns. We build a publicly available dataset that captures the historical structural changes of 223 websites[1].

[1] https://doi.org/10.5281/zenodo.17562074




*WWW Companion '26, Dubai, United Arab Emirates*

ACM ISBN 979-8-4007-2308-7/2026/04
https://doi.org/10.1145/3774905.3795444

## 2 Background

Longitudinal studies of the Web rely on snapshots—historical captures that let us observe how a page's structure changes over time. In this paper, a snapshot (or version) is the archived, server-rendered HTML of a site's landing page as preserved by the Internet Archive's Wayback Machine [10] (Wayback for short) and discoverable via its standard interfaces. Wayback is a non-profit digital archive that preserves past versions of public web pages. We analyse the DOM as archived and do not execute client-side code during interpretation.

Within a snapshot, we distinguish between DOM nodes and DOM positions. A DOM node is any object in the HTML tree; here it is either an element (such as div, a, or li) or an attribute attached to an element. Because we are interested in anchors commonly used for web-testing locators and automation, we restrict attribute nodes to id. Treating id explicitly as an attribute of interest aligns with widespread practice in robust locator construction and avoids conflating structural anchors with stylistic or non-unique attributes. A DOM position is the absolute XPath that resolves to a specific node within a particular snapshot. Positions let us describe where something sits in the document's structure at that time without assuming that an element's identity persists across redesigns. Throughout the paper, references to "positions" mean these snapshot-relative, absolute paths computed from the root using the document hierarchy and sibling indices. The dataset exposes a single binary ground-truth label, changed, defined for adjacent snapshots of the same site. For elements, a position is labeled changed if the element's absolute XPath differs between the two snapshots, which also covers the case where the element disappears. For id attributes, a position is labeled changed if the id value differs between the two snapshots, regardless of whether the element's XPath moves. This asymmetric definition separates structural movement (captured for elements) from identifier stability (captured for id) and avoids double counting the same event at both the element and its attribute. Because subsequent analyses and models rely on structure rather than content, each position is also accompanied by structural features that characterize its position within the tree [18, 21]. These features are defined to be content agnostic and informative for position stability prediction.

Finally, to ensure the dataset reflects prominent, actively maintained sites and remains reproducible. We use the Tranco popularity ranking [13] as our source for finding the top ranked websites for the study. For context, many earlier web measurement studies relied on Alexa ranking lists, but Alexa has since been retired, so Tranco is the natural replacement for reproducible, citable top site sampling [2, 22].

### 2.1 Related Work

Large-scale datasets for studying Web change typically originate from repeated crawls, but they rarely provide fine-grained, longitudinal **structural** ground truth at the level needed to evaluate DOM-anchored maintenance techniques [24].

Callan et al. created the dataset ClueWeb09 [4] providing a billion-page snapshot crawl (with smaller subsets for experimentation), enabling reproducible IR and web mining evaluations on a static collection. Its successor ClueWeb12 [5] similarly supports large-scale experimentation on a later crawl, and its collection process explicitly considers filtering undesirable content to improve research utility. While invaluable for scale, the datasets focus on the content of the websites for the purpose of information retrieval and not for structural analysis.

web-scale datasets provide repeated snapshots of the open web over long time spans. Common Crawl [6] is explicitly leveraged as a longitudinal dataset, and recent work proposes methods to make multi-snapshot longitudinal studies more practical by exploiting its indices [24]. However, such data primarily expose page-level captures and metadata, and researchers typically must perform additional extraction and alignment to recover structural representations suitable for DOM-centric benchmarking. Complementary datasets exist in software engineering that capture **web testing artifacts** rather than page evolution; for instance, E2EGit [19] curates thousands of real-world end-to-end web tests mined from open-source repositories. These collections enable maintenance studies over tests, but do not provide longitudinal, archive-derived DOM structure histories of production websites [19].

## 3 Research Methodology

In this section, we describe our approach to constructing a representative dataset of structural DOM changes. We detail the data mining process using Wayback and explain how we construct ground truth labels for changes in the DOM structure.

### 3.1 Data Collection

From the Tranco top 1,000 domains (of the 1 million domain list), we drew a simple random sample of 300 domains as our candidate cohort. To build a representative dataset of real-world web evolution, we mined websites from the Wayback Application Programming Interface (API) to collect historical snapshots. Using its API, we retrieved the DOMs of the landing pages for 300 candidates. Since some archived snapshots were spaced by only seconds, we enforced a *minimum interval of one day* between collected versions. For each website, we retrieved up to 1,000 recent versions, subject to this time constraint. This approach ensures that each snapshot captures meaningful structural changes rather than trivial updates, while providing sufficient temporal coverage to observe long-term evolution patterns. By focusing on high-traffic websites from Tranco, we ensure that our dataset represents real-world web applications that are actively maintained and frequently updated, reflecting the challenges faced by practitioners in production environments. During processing, some websites were excluded because their structure or implementation (e.g., non-standard markup, or language-specific constructs) caused failures in the automated pipeline. Additionally, we omitted domains that raised ethical concerns, such as adult content.

Landing pages were selected for archival consistency and relevance to locator robustness, as deeper pages often require authentication and are less uniformly archived. While our approach uses periodic snapshots, the dataset reflects continuous evolution in a longitudinal sense, spanning hundreds of versions per site. We acknowledge that dynamic content (e.g., news articles or advertisements) may contribute to observed changes; however, our analysis focuses on structural patterns rather than transient content. While elements such as images, text, or videos can vary, the

underlying DOM structure that hosts these components typically remains stable and is captured as part of the structural layer of the website.

*3.1.1 Mining Pipeline.* Our data collection and processing pipeline consists of five sequential stages, as depicted in Figure 1.

**Stage 1: Web Archive Retrieval.** The pipeline begins by querying Wayback to retrieve available snapshots for each of the 223 selected websites. Wayback maintains an index of all archived versions, organised by URL and collection timestamp. We query this index to obtain a complete list of available snapshots for each website's main page.

**Stage 2: Temporal Filtering.** In the second stage, we filter the retrieved snapshots based on temporal constraints. Since archived snapshots may be captured at very short intervals (sometimes only seconds apart) consecutive snapshots often contain negligible differences. To ensure that our dataset captures meaningful changes, we enforce a *minimum interval of one day* between consecutive collected versions, to do so, we pick we start with one version, then pick the next version with at least one day apart. This filtering step removes redundant snapshots while preserving sufficient temporal density to observe evolutionary patterns. Through this process, we transform an unfiltered set of snapshots into a time-curated sequence suitable for analysis.

**Stage 3: DOM Structure Extraction.** For each filtered snapshot, we download and parse the HTML content to extract the complete DOM tree structure. This stage reconstructs the Document Object Model for each webpage version, capturing the hierarchical relationships between elements, their attributes, and textual content. The extraction is done by retrieving the web page as a string, we then parse the string using Beautifulsoup python library, the reason for choosing the library is that it stores the relationship of each node with other nodes (e.g. children, siblings, parent). The extracted DOM is stored in a structured format that preserves all information necessary for subsequent analysis.

**Stage 4: Node Extraction and Timestamping.** In this stage, we extract all nodes (both element and attribute nodes) from each DOM structure and assign timestamps corresponding to the snapshot's collection date. For each element node, we compute its absolute XPath expression from the document root. For attribute nodes, we specifically extract and record `id` attributes along with their absolute XPath expressions. For each node, we also extract structural features, that can be used to train models on structural change, the features represent the relationship a node has within the structure of the DOM.

**Stage 5: Change History Computation.** The final stage computes the change history by comparing nodes across consecutive versions. For each node (element or `id` attribute), we determine whether its absolute XPath expression persists between consecutive versions. Positions are labelled **TRUE** if their XPath remains identical, and **FALSE** if the XPath changes. In the case of `id` nodes, if the value of the attribute has to matches between two versions regardless of the XPath, it is labeled **TRUE** otherwise, it is considered **FALSE**. We make this distinction to avoid two different nodes( the element node and the XPath node residing in it) having the same label increasing the accuracy of the data.

## 3.2 Ground Truth Construction

We construct two types of ground truth: (1) DOM position changes, used to train and evaluate the stability prediction model, and (2) locator breakage, used for the overall evaluation of robust locator generation. These ground truth labels enable systematic evaluation of locator stability prediction and test repair techniques.

## 3.3 DOM Position Changes

A DOM position resolves to either an element node or an attribute node. We distinguish between these two cases and describe how the ground-truth labels are collected for the nodes in each case.

*Positions Resolving to Element Nodes.* We define the structural persistence of a DOM position based on its presence across versions, using exact XPath matching. If an absolute XPath expression resolving to a given DOM position appears in two subsequent versions, we label the position as *False* (i.e. unchanged); otherwise, it is *True* (i.e. changed).

*Positions Resolving to Attribute Nodes.* For attribute nodes, we restrict ground-truth labelling to `id` attributes only. While any attribute can reside at a DOM position (such as `class`, `style`, or `data-*`). Attributes such as class or style are not unique within a document, making reliable change detection difficult. In contrast, `id` attributes are defined to be unique within the DOM and often serve as structural anchors in XPath locators (e.g., `//*[@id="foo"]`). Their consistent structural role across versions makes them a reliable signal of persistence [17]. By labelling only positions that resolve to `id` attributes, we reduce noise from semantically unstable attributes and preserve the position-centric modelling focus. To increase the validity of the data, we label `id` attribute position as stable if its absolute XPath expression appears in both consecutive versions, and unstable otherwise. This ensures consistency in our ground truth construction across both element and attribute nodes.

Figure 2 shows that the majority of websites exhibit only a small number of DOM changes between adjacent archived versions. Most observations lie in the lowest bins (i.e. 0–10 changes), while a long right tail captures a minority of websites with substantially higher changes. The median $\approx$ 14.32 lies in the lower range, indicating that at least half of the websites change only modestly from one version to the next. This distribution is consistent with our data collection protocol snapshots are spaced by at least one day so the low change mass reflects typical structural stability rather than trivial instantaneous edits. However, the tail reveals outliers undergoing frequent and extensive redesigns, such as Canva (68.43%), Repubblica (56.92%), and MSN (53.41%). These sites likely represent highly dynamic environments—Canva as a design platform with frequent UI updates, Repubblica as a news portal with continuous content and layout changes, and MSN as a media aggregator with evolving presentation strategies. Such behaviour suggests that certain domains (e.g., design tools, news, and media) prioritize rapid iteration and experimentation, leading to structural volatility. This heterogeneity underscores the importance of adaptive test maintenance strategies: while most websites evolve gradually, robust approaches must account for extreme cases where structural churn is the norm.

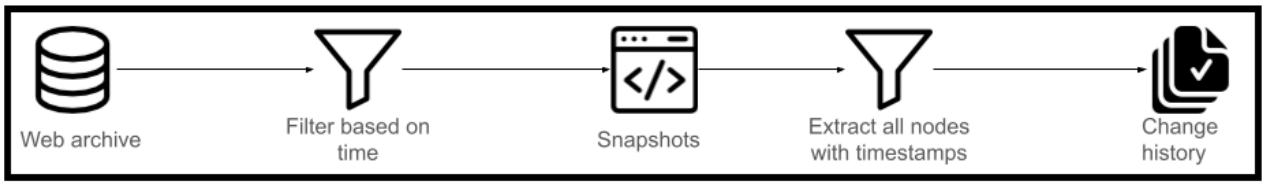



**Figure 1: Overview of the mining pipeline. (1) retrieving snapshots from the web archive, (2) filtering snapshots based on temporal constraints, (3) extracting DOM structures from HTML, (4) extracting timestamped nodes and computing their XPath expressions and positioning information, and (5) computing change history by comparing consecutive versions to establish ground truth labels.**

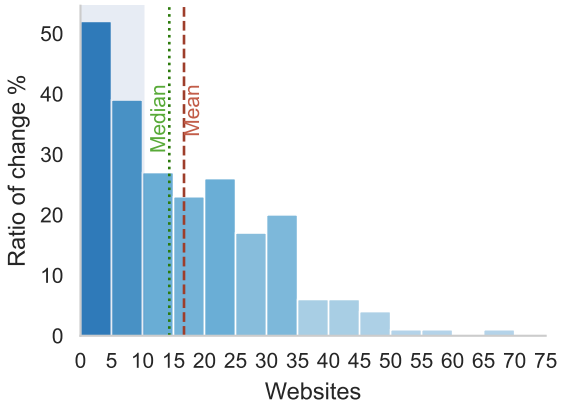



**Figure 2: Distribution of per-website DOM change ratio between consecutive versions across the 223 websites in our dataset. Most websites exhibit few changes (left-skewed with a long right tail), and the vertical lines indicate the *median* (green, dotted) and *mean* (brown, dashed). The gradient encodes bin frequency (darker indicates higher counts).**

## 4 Position Change Dataset

The dataset comprises of the following:

- Number of websites: 223.
- Total instances (rows) across websites: 246,407,044. This reflects the cumulative number of DOM nodes observed across all versions for all websites.
- Per-website size: The median website contains 855,530 rows (mean $\approx$ 1, 104, 964.32; highly skewed distribution driven by a few very large sites).

These figures indicate that the dataset is large-scale, with billions of DOM attributes and paths implicitly represented through XPath and node metadata, and nearly a quarter-billion concrete node observations used in the analyses below. Each website contains many versions (unique timestamp values), enabling change analyses. Unique timestamps per website: min 90, median 993, mean 906, max 999. In practice, most websites approach the upper bound (i.e. 1,000 versions). The maximum value of 999 reflects the fact that, although 1,000 versions were crawled, the initial version is excluded from change computations because no prior state exists for comparison. The binary flag changed marks whether a node's value changed relative to the previous version.

### 4.1 DOM Element Distribution

Figure 3 presents the distribution of the top 15 most frequent node types across all websites in our dataset, expressed as the average percentage of total nodes per website. The data reveal clear patterns in modern web development practices.

**Structural Container Elements** dominate the distribution, with `div` elements comprising the largest share at 26.34% of all nodes on average. This prevalence reflects the widespread adoption of `div` as the primary layout and grouping element in modern HTML. The `a` (anchor/link) element follows at 20.14%, indicating the heavily hyperlinked nature of web content. The `li` (list item) element accounts for 10.38%, and `span` for 9.05%, both serving as common structural and inline formatting containers. **Interactive and Form Elements** are represented by `input` (1.01%), `option` (0.85%), suggesting that while present, form-related elements constitute a relatively small proportion of the overall DOM structure on main pages. This aligns with expectations, as landing pages typically emphasise content presentation over user input mechanisms. **Media and Graphics Elements** include `img` at 3.76%, `svg` at 1.06%, and `path` at 1.68%. The presence of `svg` and `path` elements reflects the growing use of scalable vector graphics for icons, logos, and data visualisations in modern web design. **Heading and Typography Elements** are represented by `h2` (0.91%), `h3` (1.17%), `p` (2.69%), and `i` (0.97%), indicating moderate use of semantic heading hierarchy and paragraph structures. **Attribute Nodes** represent 6.80%, corresponding to the `id` attributes we specifically extracted for ground truth construction. This percentage indicates that approximately one in fifteen nodes possesses an `id` attribute, confirming that `id` attributes are selectively applied rather than ubiquitously assigned.

### 4.2 DOM Position Stability

Figure 4 presents the average distribution of *changed* versus *unchanged* DOM positions across all websites in the dataset. On average, **16.6% of DOM positions change** and **83.4% remain stable** between consecutive snapshots (median change 14.0%). This striking imbalance demonstrates the highly dynamic nature of modern web applications. Even with a one-day minimum interval between

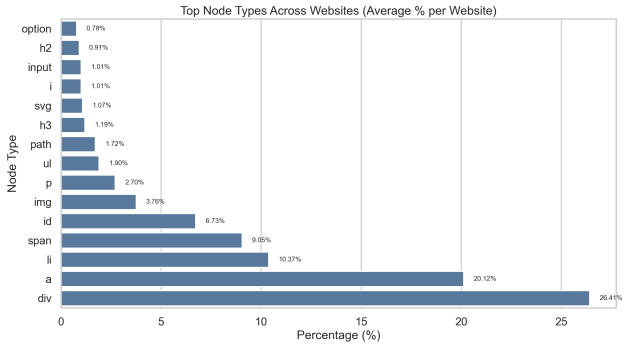



**Figure 3: Distribution of the top 15 most frequent node types across 223 websites, shown as average percentage per website. Structural container elements (`div`, `a`, `li`, `span`) dominate, collectively accounting for approximately 65% of DOM nodes. The `id` attribute appears in 6.80% of nodes.**

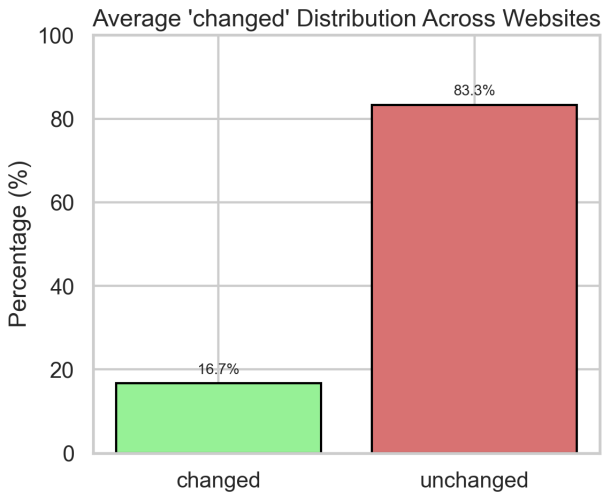



**Figure 4: Average distribution of changed versus unchanged DOM positions across 223 websites. 83.4% of positions remain stable between consecutive versions, while 16.6% undergo structural changes.**

collected versions, the a relatively high number of DOM positions exhibit structural changes across versions.

The average share of changed nodes (TRUE): 16.6% (median 13.99%), implying that roughly 5–6 out of every 6 nodes stay the same between adjacent versions on average. Average share of unchanged nodes (FALSE): 83.4%. Range across websites: from 0.01% (almost static) up to 68.43% (high change). Only 3 websites exceed 50% changed nodes on average between consecutive versions. Examples at the high end include canva (68.43%), repubblica (56.92%), and msn (53.41%); extremely low-change cases include ctrip (0.01%) and douyu (0.54%). This high rate of instability has implications for automated web testing. Individual websites exhibit substantial variation in stability rates. Some websites undergo frequent, comprehensive restructuring, while others maintain more stable DOM structures over time. This heterogeneity suggests that test maintenance strategies may need to adapt to the specific evolution patterns of the application under test.

## 5 Potential Uses of the Position Change Dataset

Our dataset enables diverse research and practical applications in web testing and maintenance. In this section, we outline several key use cases and expected benefits for different stakeholder groups.

*Data-Driven Model Training.* The primary intended use of the dataset is to train machine learning models that predict DOM position stability across versions. Researchers can leverage the temporal sequences of DOM structures and ground truth stability labels to develop predictive models using, for example, recurrent neural networks. Furthermore, the dataset's DOM-structured sequences can be leveraged to fine-tune or prompt LLMs for tasks such as forecasting DOM property changes, semantic classification, and automated navigation or repair [7].

*Benchmarking Test Repair Tools.* Test repair and self-healing frameworks can be evaluated using the dataset's locator breakage ground truth. By simulating test failures at each version transition, researchers can assess how well different repair strategies (WATER, WATERFALL, UITestFix, etc.) recover broken locators. The 223 websites provide diverse evolutionary patterns, enabling evaluation across different Domains and development practices. The dataset's public availability facilitates reproducible research and enables fair comparison of tool effectiveness.

*Studying Web Evolution Patterns.* Beyond test automation, the dataset provides rich data for analysing web evolution at scale. Researchers can study:

- Frequency and magnitude of DOM structure changes across different website categories
- Temporal patterns of evolution (e.g., whether changes are gradual or episodic)
- Correlation between website characteristics (traffic rank, domain type, industry) and evolution rates
- Stability of specific DOM patterns and their resilience to change

Such analyses can inform best practices for web development and help practitioners understand typical evolution patterns in their domain.

## 6 Threats to Validity

**Internal Validity** The enforced one-day minimum interval between collected versions may miss rapid structural changes that occur within shorter timeframes. Modern web development practices, including continuous deployment and A/B testing, can introduce multiple versions per day. Future work could explore collecting finer-grained temporal data for websites employing continuous deployment strategies.

Our ground truth definition relies on exact matching of absolute XPath expressions between consecutive versions. This approach may be overly strict, as minor, non-structural changes (such as

reordering of attributes or addition of non-semantic whitespace) could cause XPath strings to differ even though the element remains semantically identical. Conversely, some structural changes might not affect the absolute XPath if elements maintain their relative positions.

**External Validity** Our dataset captures DOM snapshots of static HTML as preserved by Wayback. Modern web applications increasingly rely on client-side rendering with JavaScript frameworks (React, Vue, Angular), which may dynamically generate or modify DOM structures not fully captured by archived snapshots. Wayback indexes the server-rendered HTML, potentially missing runtime DOM modifications introduced by JavaScript execution and dynamic content. This is by design, as we ignore client-side rendering to focus on the archived, server-rendered DOM, because framework-generated dynamic content populates the existing structure instead of redefining it.

## 7 Conclusion

We have presented a large-scale dataset of web evolution compiled from 223 high-traffic websites using the Wayback API. Our dataset comprises up to 1,000 DOM structure snapshots per website, temporally filtered to enforce a one-day minimum interval between versions. We construct fine-grained ground truth labels for large-scale data-driven solutions addressing web applications.

The dataset enables multiple research and practical directions: training and validating stability prediction models, benchmarking test repair and self-healing frameworks, analysing web evolution patterns at scale, improving locator robustness guidelines, training machine learning-based testing tools, and conducting software engineering research on web application maintenance. We make the dataset publicly available to the research community, with the goal of facilitating reproducible research and accelerating progress toward more robust, maintainable web test automation.

Future work can extend this dataset by incorporating additional websites across different categories, collecting multi-page snapshots to better represent complex applications, analysing client-side rendering frameworks, and exploring alternative locator strategies beyond XPath. We also plan to develop reference implementations of stability prediction models trained on this dataset, providing baseline approaches for the community to build upon. By continuing to expand and refine web evolution datasets, we can establish a foundation for more principled, data-driven approaches to web test maintenance.

## References


[1] Eytan Adar, Jaime Teevan, Susan Dumais, and Jonathan L. Elsas. 2009. The Web Changes Everything: Understanding the Dynamics of Web Content. In *Proceedings of Web Search and Data Mining (WSDM) 2009* (proceedings of web search and data mining (wsdm) 2009 ed.). Association for Computing Machinery, Inc. https://www.microsoft.com/en-us/research/publication/the-web-changes-everything-understanding-the-dynamics-of-web-content/

[2] Alexa Support. 2021. We will be retiring Alexa.com on May 1, 2022. https://web.archive.org/web/20220119171322/https://support.alexa.com/hc/en-us/articles/360061329348-We-will-be-retiring-Alexa-com-on-May-1-2022. Archived copy (Wayback Machine), accessed 2026-01-14.

[3] Sebastian Balsam and Deepti Mishra. 2025. Web application testing—Challenges and opportunities. *Journal of Systems and Software* 219 (2025), 112186. doi:10.1016/j.jss.2024.112186

[4] Jamie Callan, Michael Hoy, Choonsik Yoo, and Le Zhao. 2009. ClueWeb09 Dataset. http://www.lemurproject.org/clueweb09/. Carnegie Mellon University, Language Technologies Institute.

[5] Jamie Callan, Michael Hoy, Choonsik Yoo, and Le Zhao. 2012. ClueWeb12 Dataset. http://www.lemurproject.org/clueweb12/. Carnegie Mellon University, Language Technologies Institute.

[6] Common Crawl Foundation. 2026. Common Crawl. https://commoncrawl.org/. Accessed: 2026-02-02.

[7] Xiang Deng, Prashant Shiralkar, Colin Lockard, Binxuan Huang, and Huan Sun. 2022. DOM-LM: Learning Generalizable Representations for HTML Documents. *ArXiv* abs/2201.10608 (2022). https://api.semanticscholar.org/CorpusID:246285527

[8] Sergio Di Meglio and Luigi Libero Lucio Starace. 2024. Towards Predicting Fragility in End-to-End Web Tests. In *Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering* (Salerno, Italy) *(EASE '24)*. Association for Computing Machinery, New York, NY, USA, 387–392. doi:10.1145/3661167.3661179

[9] Dennis Fetterly, Mark Manasse, Marc Najork, and Janet Wiener. 2003. A large-scale study of the evolution of web pages. In *Proceedings of the 12th International Conference on World Wide Web* (Budapest, Hungary) *(WWW '03)*. Association for Computing Machinery, New York, NY, USA, 669–678. doi:10.1145/775152.775246

[10] Internet Archive. 2025. Wayback Machine. https://web.archive.org/. Accessed: 2025-06-26.

[11] Iva Kertusha, Gebremariem Assress, Onur Duman, and Andrea Arcuri. 2025. A Survey on Web Testing: On the Rise of AI and Applications in Industry. arXiv:2503.05378 [cs.SE] https://arxiv.org/abs/2503.05378

[12] Wallace Koehler. 2003. A longitudinal study of Web pages continued: A consideration of document persistence. *Inf. Res.* 9 (01 2003).

[13] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. 2019. Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS 2019)*. doi:10.14722/ndss.2019.23386

[14] Maurizio Leotta, Diego Clerissi, Filippo Ricca, and Cristiano Spadaro. 2013. Comparing the maintainability of selenium WebDriver test suites employing different locators: a case study. In *Proceedings of the 2013 International Workshop on Joining AcadeMiA and Industry Contributions to Testing Automation* (Lugano, Switzerland) *(JAMAICA 2013)*. Association for Computing Machinery, New York, NY, USA, 53–58. doi:10.1145/2489280.2489284

[15] Maurizio Leotta, Filippo Ricca, and Paolo Tonella. 2021. Sidereal: Statistical adaptive generation of robust locators for web testing. *Software Testing, Verification and Reliability* 31 (5 2021), e1767. Issue 3. doi:10.1002/STVR.1767

[16] Maurizio Leotta, Andrea Stocco, Filippo Ricca, and Paolo Tonella. 2014. Reducing Web Test Cases Aging by Means of Robust XPath Locators. In *2014 IEEE International Symposium on Software Reliability Engineering Workshops*. 449–454. doi:10.1109/ISSREW.2014.17

[17] Maurizio Leotta, Andrea Stocco, Filippo Ricca, and Paolo Tonella. 2016. Robula+: An Algorithm for Generating Robust XPath Locators for Web Testing. *J. Softw. Evol. Process* 28, 3 (mar 2016), 177–204. doi:10.1002/smr.1771

[18] Sergio Di Meglio and Luigi Libero Lucio Starace. 2024. Towards Predicting Fragility in End-to-End Web Tests. In *ACM EASE*. doi:10.1145/3661167.3661179

[19] Sergio Di Meglio, Luigi Libero Lucio Starace, Valeria Pontillo, Ruben Opdebeeck, Coen De Roover, and Sergio Di Martino. 2025. E2EGit: A Dataset of End-to-End Web Tests in Open Source Projects. In *2025 IEEE/ACM 22nd International Conference on Mining Software Repositories (MSR)*. 836–840. doi:10.1109/MSR66628.2025.00121

[20] Michel Nass, Emil Alégroth, Robert Feldt, Maurizio Leotta, and Filippo Ricca. 2023. Similarity-Based Web Element Localization for Robust Test Automation. *ACM Trans. Softw. Eng. Methodol.* 32 (4 2023). Issue 3. doi:10.1145/3571855

[21] Yu Pei, Jeongju Sohn, and Mike Papadakis. 2025. An Empirical Study of Web Flaky Tests: Understanding and Unveiling DOM Event Interaction Challenges. In *IEEE International Conference on Software Testing, Verification and Validation (ICST)*. Findings: element-level interactions across multiple DOMs increase flakiness; DOM-event categories and fixes.

[22] Quirin Scheitle, Oliver Hohlfeld, Julien Gamba, Jonas Jelten, Torsten Zimmermann, Stephen Strowes, and Narseo Vallina-Rodriguez. 2018. A Long Way to the Top: Significance, Structure, and Stability of Internet Top Lists. 478–493. doi:10.1145/3278532.3278574

[23] Andrea Stocco, Rahulkrishna Yandrapally, and Ali Mesbah. 2018. Visual web test repair. *ESEC/FSE 2018 - Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 503–514. doi:10.1145/3236024.3236063

[24] Henry S Thompson. 2024. Improved methodology for longitudinal Web analytics using Common Crawl. In *Proceedings of the 16th ACM Web Science Conference* (Stuttgart, Germany) *(WEBSCI '24)*. Association for Computing Machinery, New York, NY, USA, 59–69. doi:10.1145/3614419.3644018