Enabling Correct and Efficient Web Archiving in the Era of JavaScript

Two key goals of any web archive are to serve archived pages with high fidelity to users and to minimize operational costs. However, the widespread use of JavaScript (JS) on the web is currently a significant hindrance in satisfying either objective. Today, web archives crawl and save large amounts of JS and, yet, the snapshots of many JS-heavy pages fail to accurately mimic how those pages originally looked and functioned. To address these seemingly contradictory problems, University of Southern California requests $428,043 in funding for a three-year Applied Research project which aligns with the NLG-L program's Objective 3.2 under Goal 3. We seek to develop methods which will 1) ensure that archived pages load without any errors for users, and 2) enable budget-constrained libraries and museums to archive many more pages than they can today. Our work will be guided by an advisory board comprising representatives from a variety of organizations that build and use services that enable web archiving, and we seek to inform the development of next-generation archival software and services. The end result of our work will be to enable high-fidelity preservation of web content at low cost.

1 Project Justification

Over time, a web page may cease to exist at the URL where it was originally available [29,38] or the content available at that URL might change due to the page being modified [24,31]. Therefore, web archives play a key role in the web ecosystem, enabling users to lookup the content that existed at any particular URL at various times in the past.

Any web archive, such as the Internet Archive (IA) and the Library of Congress, repeatedly crawls web pages over time and saves many snapshots of every page. For every snapshot of a page, an archive first downloads all resources (e.g., HTMLs, CSS stylesheets, JavaScripts, images) on the page, and then stores these resources after rewriting all URL references to point to the copy hosted by the archive. When a user wants to later view any stored snapshot of a page, the user's web browser loads the snapshot from the archive in the same manner as it would load any page on the live web.

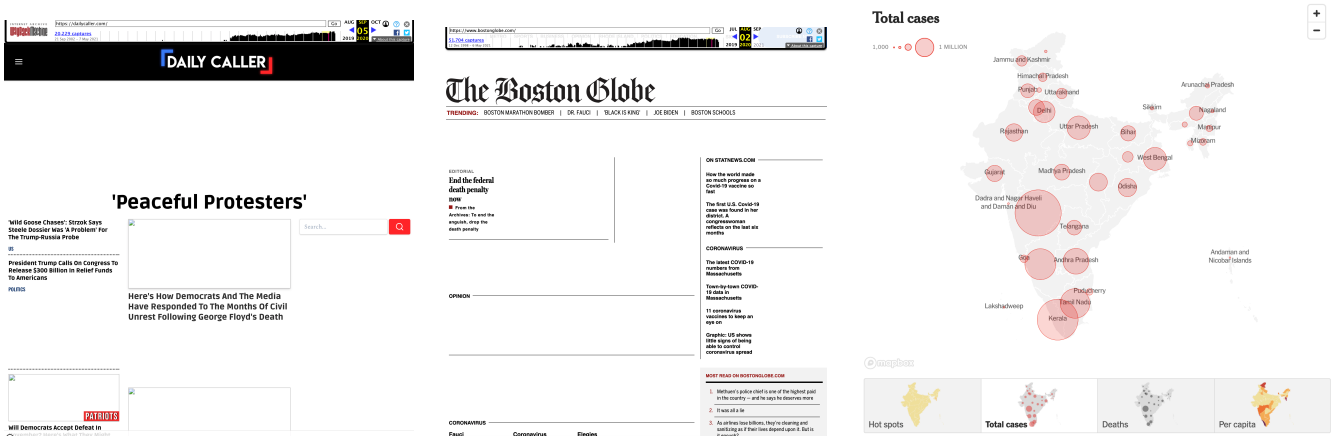
Context for our proposed work. If one compares the web of today with that from a decade ago, one dramatic change that is apparent is the increased use of JavaScript, up from around 100 KB on the median web page in 2010 to over 500 KB today [11]. To cope with this increase, browser-based crawlers such as Brozzler and Browsertrix have been developed to enable web archives to save the resources on any page that are fetched by JS [18]. In addition, when a user interacts with a page (e.g., clicks a button or hovers over an image), systems such as Conifer enable archival of the content fetched by the JS which gets executed to handle the user's interaction.

But, existing solutions have two significant shortcomings, both of which stem from properties of modern JavaScript which web archives currently do not account for.

Non-deterministic JS execution breaks archived pages. First, across multiple loads of a page, the manner in which JS code on the page gets executed can vary based on: 1) characteristics of the client device, e.g., Chrome or Safari, 3G or WiFi, or even the resolution of the display, and 2) values returned by the web browser in response to requests for random numbers, the current time, etc. As a result, when a user loads an archived page snapshot, the browser's execution of JS on the page might cause it to request resources which were never crawled. Due to the complex dependencies between the resources on a page, one failed resource fetch often has a cascading effect on the rest of the page load, and many other resources go unfetched.

Non-functional JS hampers archival efficiency. Second, a significant fraction of the JS on a typical page relies on the user's device interacting with the page's servers, e.g., JS that enables a page's provider to push real-time notifications to the user and JS that enables users to post comments. However, such functionality cannot work on an archived page. Therefore, crawling and storing the JS that powers these capabilities consumes network, compute, and storage, without any resultant benefits to preserving page fidelity.

Example of adverse impact in practice. Based on our review of the literature and our conversations over the last two years with various stakeholders involved in web archiving, there is not much empirical



(a) dailycaller.com [16] (b) bostonglobe.com [15] (c)

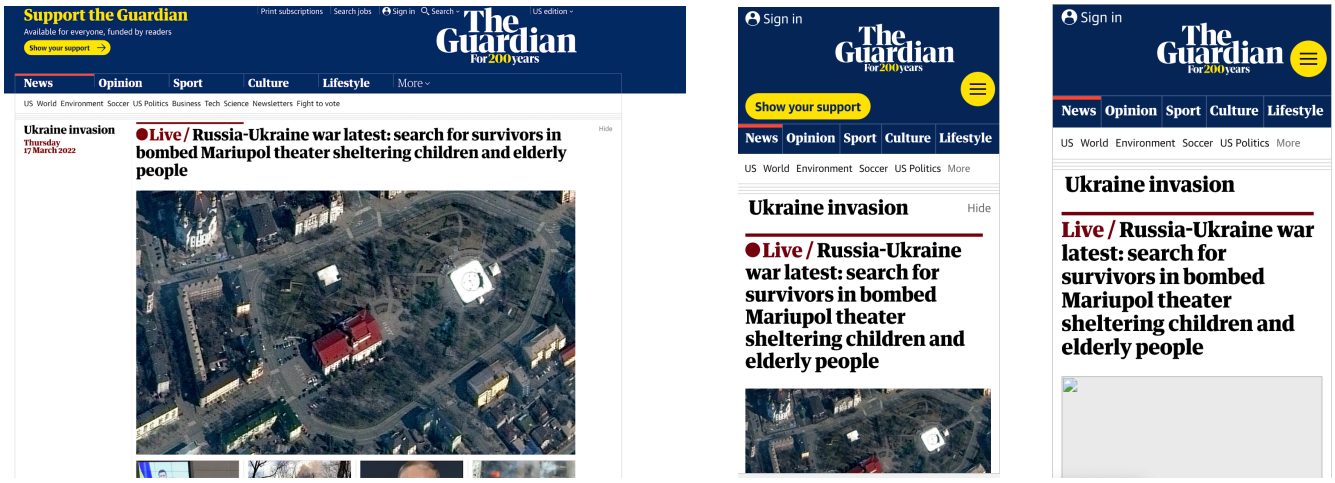
Figure 1: **(a and b) Examples of page snapshots loaded from the Internet Archive with missing images. (c) Example of interactions on a page that are enabled by JavaScript; the user can see different types of statistics by clicking on the appropriate icons.**

data or widespread awareness about the aforementioned problems. Therefore, drawing upon our prior work in programmatically instrumenting [26] and analyzing [30] JS code, we showed in a recent paper [27] that the Internet Archive's snapshots for many JS-heavy pages render with missing page resources and runtime errors reported by the browser; Figures 1(a) and 1(b) show a couple of examples. We showed that web archives cannot sidestep these problems by archiving pages as screenshots because modern web pages often include interactive features (such as menu bars and image carousels) that are powered by JS; Figure 1(c) shows an example from a NYTimes article on coronavirus in India, wherein the user can see different types of statistics by clicking on different insets of the included graphic. Moreover, in a corpus of 1 million page snapshots (roughly 3500 page snapshots each for 300 websites) that we downloaded from the Internet Archive, we were able to safely discard 36% of the JavaScript bytes without any apparent loss of fidelity. One of the two PhD students who led this effort, Ayush Goel, will present our findings at the IIPC's Web Archiving Conference (WAC) later this year in May.

Target group. In addressing the above-described adverse impacts of JavaScripts on web archives, our primary goal is to help inform changes to how web archives operate. In particular, we will work closely with the teams at the Internet Archive and at the Webrecorder project who are responsible for the development of their browser-based web crawlers, namely Brozzler [6] and Browsertrix [5]; Jefferson Bailey from the Internet Archive and Ilya Kreymer, the lead developer of Browsertrix, have agreed to serve as advisors for this project. Both of these crawlers form the basis of widely used web archiving services [2, 4] offered by these institutions. So, any improvements we enable will benefit a large population of users. As an example of similar impact, our recent work [32] has helped inform improvements in the tools used by the Internet Archive to patch broken external references in Wikipedia articles.

More broadly, a number of organizations—cultural heritage institutions, national libraries, and public museums—operate web archives to ensure long-term preservation of content on the web; a recent survey estimates that there are 119 web archives in the United States alone [23]. The principles for high-fidelity low-cost web archiving that we develop as part of our work will, thus, have the potential for widely applicable long-term impact.

In addition, our work will help improve the quality of two important collections of archived pages: 1) the End of Term (EOT) dataset [10], which captures a snapshot of every publicly available federal website once every four years at the time of presidential transition, and 2) the Collaborative ART Archive (CARTA) [8], which includes at-risk web-based art materials captured collaboratively by the Internet Archive and the New York Arts Resources Consortium (NYARC). By identifying the ways in which JavaScript is affecting the fidelity with which page snapshots in these collections can be replayed and by developing methods to address these negative effects of JavaScript, we will improve the long-term sustainability of these vital datasets. We will work closely with Mark Phillips (Associate Dean for Digital Libraries at the University



(a) (b) (c)

Figure 2: **Screenshots of guardian.com's homepage when we (a) load it on a laptop, (b) load it on a smartphone, and (c) crawl it on a laptop, and then load this copy on a smartphone.**

of North Texas, which is one of the partner institutions involved in the collection of the EOT dataset) and Sumitra Duncan (Head of the Web Archiving Program at NYARC), two of our advisors on this project.

Beneficiaries. In the long term, our work will have a positive impact on all users of web archives. For example, a common use of web archives is to look up a copy of the linked page when a user encounters a broken link on the web. In such cases, our work will help ensure that the copy will load without errors and any functionality on a page that can work on archived copies indeed works. This can have significant widespread impact given that the Internet Archive serves millions of people each day and is one of the top 300 web sites in the world [12]. By reducing the amount of JS that needs to be saved per page and thereby enabling more pages to be archived for the same operational cost, our work will also increase the fraction of broken links on the web that can be patched by linking to the archived copies of these links.

On the other hand, our work will enable services like Archive-It to give all of their customers the option to filter JS code that will never be executed. This will greatly help libraries with limited budgets [13] by enabling them to safely discard JS files and significantly reduce their costs.

2 Research Questions and Proposed Work

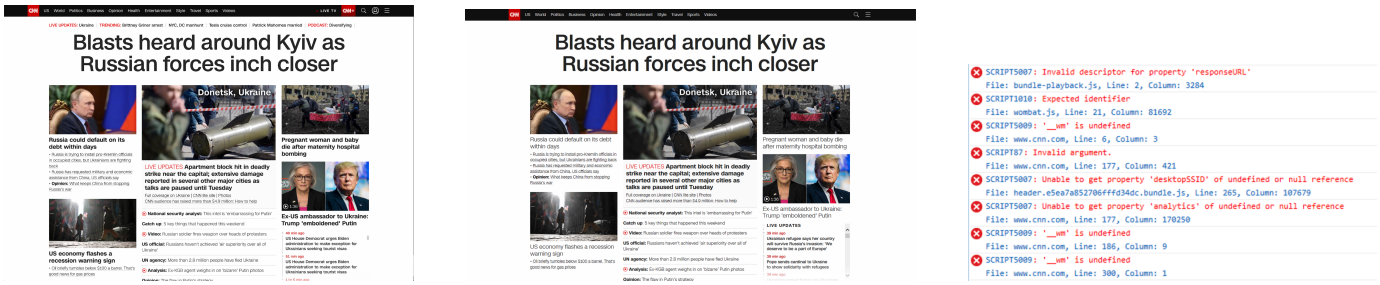
To address the impact of JavaScript on web archives, we propose to tackle three research questions.

- *RQ1:* Given a set of archived resources (i.e., HTML, CSS, JS, images, etc.) for a web page, how can we programmatically assess whether page fidelity might be compromised in *some* future load of the page?
- *RQ2:* How should archival systems be modified so that, in *all* loads of the archived copy of a page, it is guaranteed to mimic the page as it existed at the time of archival?
- *RQ3:* How can an archival crawler identify which subset of the JavaScript code on a page will *never* be used in *any* future load of the page's archived copy and is, therefore, optional for it to fetch and save?

In this section, we describe the challenges involved in answering these questions and the methods that we plan to use to address these challenges.

2.1 Assessing fidelity of archived pages

Due to the use of JavaScript, a modern web page is essentially a computer program. This program can behave differently based on the environment in which it is executed. For example, all of us are familiar with the fact that many pages look different when we load them on our smartphones, compared to when we visit these pages on a laptop or desktop. Figure 2 shows an example of such a page: the homepage of guardian.com. We load this page once in the Chrome web browser on a Mac laptop, and again in Chrome but on a smartphone. As seen in Figures 2(a) and 2(b), the image associated with the top story



(a) (b) (c)

Figure 3: **Screenshots of cnn.com's homepage when we load it (a) with Chrome on Mac OS, and (b) with IE on Windows. (c) If we crawl the page using Chrome on Mac OS and then load this copy using IE on Windows, JS code on the page exits prematurely with runtime errors.**

looks identical (only a portion of the image is visible on the smartphone without scrolling). However, the browser is not fetching the same image file in both loads. **Instead, JS code on the page fetches images of higher/lower quality depending on the height and width of the client device's display.** Consequently, when we crawl the page in Chrome on the laptop, and then load this locally stored copy of the page from a smartphone, the browser's attempt to fetch lower quality versions of images on the page fails. As we see in Figure 2(c), the image associated with the top story is therefore missing.

The property that JavaScript on a page can execute differently in different loads of that page can also have more serious effects. As an example, Figure 3 shows loads of cnn.com's homepage in three cases: (a) in Chrome on Mac OS, (b) in IE on Windows, and (c) crawled in Chrome on Mac OS, and this local copy is then loaded in IE on Windows. Figures 3(a) and 3(b) show that the page looks largely unchanged whether we load it on a Mac or a Windows machine. However, when we crawl this page on a Mac and then load this copy on Windows, we see a blank page (not shown). **The reason for this is that, as shown in Figure 3(c), when JS code on this page is executed on Windows using a copy crawled on Mac OS, the code exits prematurely with runtime errors and the page load gets aborted.**

The implications of these observations is that it is challenging, if not impossible, for a human to reason about all feasible loads of an archived page. After archiving a page, an archivist can load the archived copy using a few different types of client types. But, it is impractical to comprehensively test all possible combinations of all the factors that can execute JS execution.

Therefore, we instead propose to use *concolic execution* [25, 36], a well-known program analysis technique which can be used to reason about all possible executions of a computer program. We have applied this technique in our prior work for enabling the web browser to safely parallelize the execution of the scripts on a page when the browser is loading that page [30]. While our goal there was to speed up the loads of pages on the live web, we plan to adapt the same technique to instead reason about the potential for fidelity to be compromised in *some* future load of an archived page snapshot.

A key question that we will address is: **when we load an archived page snapshot in a browser, how can we tell if the page's fidelity has been compromised in some form?** Simply taking a screenshot of the page after the browser has finished loading it and comparing the screenshots across loads with different client types will not suffice. On the one hand, declaring every difference in screenshots as a fidelity violation will result in many false positive alerts, e.g., as seen in Figure 2, a page may legitimately be rendered differently on different clients. On the other hand, perfectly matching page screenshots is not an assurance that all functionality on the page will work fine; the user's attempt to click on a button and play a video may not work, or when the user hovers over the sidebar, the dropdown menu may not appear. Therefore, we need a careful assessment of a page's fidelity from both a visual and functional perspective.

2.2 Archiving with guarantees on fidelity

Second, for *RQ2*, we will study how browser-based crawlers such as Brozzler and Browsertrix need to analyze and instrument the JS code that they fetch so that they can make strong guarantees about page

fidelity. Today, web archives already rewrite crawled JS to ensure that any resources fetched via JS are requested from the archive and not from the page's original servers. We will extend this practice so that, when archived JS is executed in a user's browser, it will only request resources that were saved by the web archive when the page was crawled.

Towards this end, we will study how to separately handle the two different sources of non-determinism that can affect the execution of JavaScripts on archived page snapshots.

1. First, as discussed above, page loads can be affected by the characteristics of the client used, e.g., user-agent, screen dimensions, and even, network conditions. To prevent these factors from impacting page fidelity, we plan to investigate how browser-based crawlers should be modified so that, in addition to storing the resources fetched as WARC records, they also capture and store the values returned from all browser APIs that are invoked when crawling a page; an API (application programming interface) is the interface to a function implemented within the browser, which the code on any page can invoke in order to get a specific piece of information (e.g., dimensions of the local display). We will also study how the stored HTML and scripts on a page should be rewritten so that, when a user loads the archived page, the browser uses the API return values that were saved at crawl time, rather than the values applicable to the client device that this user is using to load the page.

2. In addition to client device characteristics, the execution of JavaScripts can also vary across loads of an archived page due to browser APIs that provide random numbers, the current date/time, etc. In the corpus of 3,000 pages that we examined in our recent study [27], we found that the values from these browser APIs typically do not influence the URLs of resources fetched. However, in this project, we will study more comprehensively as to how to preserve any non-determinism which is important for the functionality on the page and does not impact the set of resources fetched, e.g., on an archived copy of the NY Times' Spelling Bee, users should still be able to shuffle the letters.

2.3 Improving efficiency without compromising fidelity

Lastly, for *RQ3*, we will develop methods which can enable browser-based crawlers to identify JS files that they could skip fetching because the presence of these scripts has no impact on the fidelity with which archived pages can be served to users. For example, as shown in Figure 4, the homepage of forbes.com includes the capability to push notifications to users who visit the page. While it takes 4 MB to save all resources on this page, 2.4 MB is accounted for by JavaScripts, of which 0.9 MB corresponds to the JS code that enables push notifications. **Fetching and saving this JS code will require a web archive to devote CPU, network bandwidth, and storage. However, all of this effort makes no contribution to preserving the page's fidelity, since push notifications will not work on any archived copy of the page.**

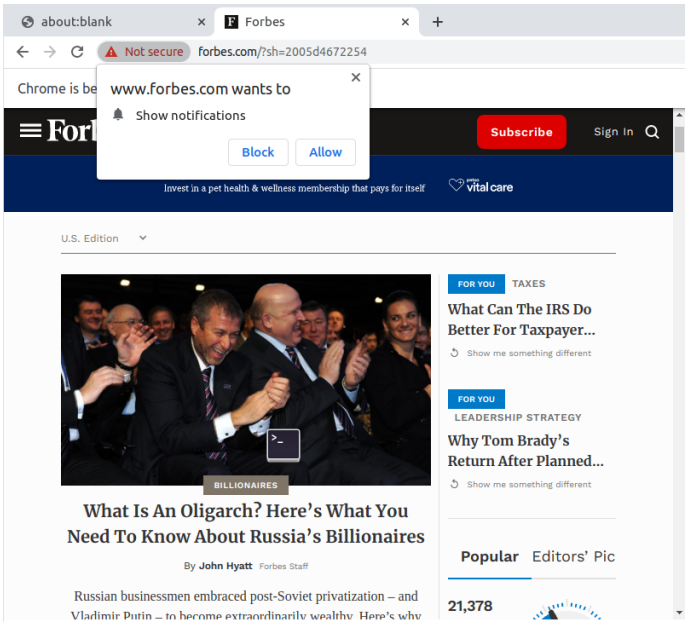


Figure 4: **The homepage of forbes.com includes 900 KB of JavaScript to enable push notifications. Saving all of this code in a web archive makes no contribution to page fidelity, since this functionality will not work on archived copies of this page.**

In general, any JavaScript code which requires communication with the page's original servers will not work on archived pages. When a user loads an archived copy of a page, the web archive can only serve those resources on the page which it saved when it crawled the page.

The question then is: how to identify which portion of the JS code on any page need not be saved, or even

fetched and processed, when the page is archived? Our preliminary findings indicate that, on a typical page, most of the code on the page which implements functionality that will not work on archived copies of the page is compartmentalized into a few third-party scripts (i.e., JS files which are served from a domain other than the page's provider). For example, on the forbes.com's homepage, all the code that implements push notifications is in scripts served from the domain pushly.com.

Therefore, to identify most of the non-functional JavaScript code in archived pages, it is unnecessary to perform any complex code analysis. Instead, it suffices to assemble and use a "filter list" which captures the features distinctive to the URLs of scripts containing non-functional code. **When crawling pages, a web archive would simply have to discard, and not even fetch, any script whose URL matches the filter list.** This filter list will comprise rules akin to those used by browser add-ons to block advertisements and trackers on the web, except that the rules we aim to identify will filter out page resources which are important on the live web but serve no functional purpose on archived page copies, e.g., on news sites, code for tracking the number of articles a user has read. In contrast to our recent work [27], where we manually compiled such a filter list for the previously mentioned corpus of 1 million page snapshots, we seek to automatically generate the filtering rules for any particular site.

3 Project Work Plan

3.1 Personnel, Expertise, and Responsibilities

All work on this project will be conducted by the PI and his graduate students. PI Madhyastha's group has been working on projects related to the web for more than a decade, and the outcomes from our work have been used by many popular web providers such as Google, Facebook, and 1-800-Flowers. His group's work in this area has resulted in several papers at top conferences [19–21,30,35,37], two of which have been recognized twice with the IETF's Applied Networking Research Prize [1] for their likely impact on Internet standards and technologies.

Of particular relevance to the proposed project is our expertise in understanding how web browsers execute JavaScripts and how the scripts on a page interact with each other. In previous work, we have used this understanding to speed up page loads by enabling web servers to prioritize the delivery of JavaScripts [21, 35] and by enabling web browsers to reuse or parallelize JavaScript executions [26,30].

Leveraging the expertise gained from our prior work, the proposed effort is one of multiple projects that we have begun in the last two years to ensure long-term preservation of web content. On the one hand, we have studied broken links in Wikipedia articles which the Internet Archive was unable to patch due to the absence of archived copies for them [32]. The findings from our study have helped inform changes to the bots that the Internet Archive runs on Wikipedia to deal with link rot. On the other hand, we have studied a large corpus of page snapshots on the Internet Archive to highlight the prevalence of fidelity violations caused due to JavaScript and to draw attention to the feasibility of improving crawling efficiency [27]. This study forms the basis of our work in this project.

One of the PhD students who co-led our above-mentioned study on the impact of JavaScript [27] – Jingyuan Zhu – will initially lead our work on this project too. In each of the three years of this project, the student will devote 20 hours per week to this project throughout the year. Jingyuan is currently a PhD student at the University of Michigan, where PI Madhyastha was a faculty member until December 2023. Following PI Madhyastha's recent move to the University of Southern California (USC), Jingyuan is currently in the process of transferring to USC. Like how Jingyuan learned the intricacies of JavaScript from PI Madhyastha's previous students, he will help train new students that PI Madhyastha will recruit at USC. One of these students will take over as the lead graduate student on the project if Jingyuan completes his PhD and graduates prior to the completion of this project.

The PI will supervise the students' work on this project and be responsible for overall management of the project. This includes reaching out to and communicating with members of the advisory board, helping disseminate our work via publications and talks, and ensuring sustainability of our work beyond the completion of the project.

3.2 Datasets

For most of our analysis in this project, we will primarily focus on two datasets. Given the importance of these datasets, it is critical to understand the impact of JavaScript on the fidelity with which the pages in these datasets can be served to users. Moreover, since both datasets are added to over time, helping address sources of fidelity violations will have significant benefits for data collected in the future.

Our first dataset will be the End of Term (EOT) dataset [10], which captures a snapshot of every publicly available federal website once every four years. The purpose of this dataset is to document the change in the web presence of the Executive Branch of the US government at the time of presidential transition. This dataset comprises data captured in 2008, 2012, 2016, and 2020, totalling over 600 TB in all. We will analyze a random sample of the pages from this dataset, focusing particularly on snapshots from 2016 and 2020, since the presence of JavaScript in them is likely to be greater.

Our second dataset will be the Collaborative ART Archive (CARTA) [8], which includes at-risk web-based art materials captured collaboratively by the Internet Archive and the New York Arts Resources Consortium (NYARC). The CARTA project utilizes the resources and professional expertise of a collaborative entity of arts and cultural heritage institutions to build collections of archived web-based content related to art history and contemporary art practice. We will initially study a random sample of the pages from CARTA's publicly available collections on Archive-It [3]. Later, we will work with our advisors from the Internet Archive and NYARC to gain access to and analyze the WARC records of these collections.

3.3 Advisory board

To ensure that we take practical considerations into account in our research project and to get feedback on our findings on a continual basis, we have enlisted four advisors, all of whom have significant expertise in the field of web archiving.

1. Sumitra Duncan is Head of the Web Archiving Program at NYARC. We will lean on her expertise in studying the CARTA dataset.
2. Ilya Kreymer, the lead developer of the Browsertrix browser-based crawler will provide us feedback on the practicality of incorporating our proposed recommendations.
3. Similarly, Jefferson Bailey, the Director of Archiving and Data Services at the Internet Archive (IA), will serve as our liaison to the team at IA which works on Brozzler, their browser-based crawler.
4. Mark Phillips is the Associate Dean for Digital Libraries at the University of North Texas, which is one of the partner institutions involved in the collection of the EOT dataset.

3.4 Proposed Work Timeline

Throughout the three year duration of this project, will meet with each of our advisors once every four to six months. The goals of these conversations will be three-fold: 1) to apprise them about our findings and our progress since we last met, 2) to solicit their feedback on whether the methods that we develop and our recommended changes to web archives are practical, and 3) to get their recommendations on the avenues that we should explore to maximize the impact of our work. Since all of our advisors have deep-rooted expertise in the field of web archiving, we will heavily lean on their input to help us disseminate our results.

Year 1: Assess page fidelity. Analyze target datasets. Raise awareness. In the first year of the project, we will focus on developing methods to automate the assessment of whether fidelity can be compromised in any load of an archived page snapshot. As described earlier, we plan to use concolic execution for this analysis. We will also study how to comprehensively assess all facets of fidelity, i.e., when a user loads an archived page, will the page render without errors and will all the functionality that can work on an archived copy indeed work?

We will then apply our methods to study pages from both the EOT and CARTA datasets. This analysis will help us identify which pages in these datasets are prone to fidelity violations when users load them, and in

what ways. Beyond sharing our findings with our advisors, we will seek to raise awareness more broadly about the adverse impacts of JavaScript on archival fidelity. For this, we will advertise some of the notable examples we find in a variety of ways, such as emails to mailing lists used by the web archival community, talks at IIPC's general assembly, and blog posts advertised on social media.

Year 2: First iteration of revising crawlers. Automate generation of filter list. Submit first set of papers. In the second year, we will begin by studying how browser-based crawlers like Brozzler and Browsertrix should be modified to prevent fidelity violations of the kind we unearthed in the first year. First, when a page is crawled, we will investigate how to save the relevant client characteristics which are queried by JavaScripts on the page. Second, we will develop custom JavaScript which can be inserted into the HTML of any archived page so that, when a user later loads this page, scripts on the page will use the saved client characteristics in place of the corresponding information provided by the user's browser.

Thereafter, we will shift focus from fidelity to efficiency. To enable web crawlers to identify JS code that will go unused in loads of archived pages, we will develop techniques to automate the identification of rules that can be used to match against the URLs of JavaScript files. We will then study the feasibility of incorporating such a filter list into Brozzler and Browsertrix. As part of this work, we will also reach out to budget-constrained libraries and museums, and seek to understand how our work might benefit them.

During this year, we expect to be able to write up our findings from the first two years and submit a research paper summarizing our work. We will also submit proposals to give talks at venues like Code4Lib [7], WADL [17], the annual meeting of the Society of American Archivists [14], and the Digital Library Federation (DLF) Forum [9].

Year 3: Refine implementation. Incorporate feedback. Focus on dissemination and sustainability. In the final year, we will focus on incorporating the feedback received from our advisors as well as in response to our papers and talks. The experience we develop over the first two years of the project will also help us identify undesired consequences of our changes, e.g., our attempts to improve fidelity might hurt efficiency and vice-versa. We will design and implement solutions to such problems in this final year of the project. We will also spend a significant fraction of our time in this year on validating our proposed changes across a corpus of several thousand pages that we crawl from a wide variety of sites on the live web.

To ensure broad and sustained applicability of our work, we will publish an open technical document that summarizes our recommendations for how web archives should assess and filter JavaScripts. Such a document will be more accessible to archivists than formal research publications. The principles that we advocate in this document can be incorporated into any browser-based crawler, not just Brozzler and Browsertrix. We will post a preliminary version of this document at the end of the second year, and the final version will address the feedback that we receive from the community during the third year.

3.5 Testing and Validation

We will evaluate the benefits of our work along three dimensions: 1) improvement in page fidelity, 2) reduction in storage, and 3) impact on crawling throughput. On both the EOT and CARTA datasets, we will periodically redo our evaluation along these different dimensions. Doing so will help us determine when improvements along one dimension has made things worse along a different dimension. For example, discarding some JavaScripts will reduce storage, but might worsen page fidelity. Or, the overheads imposed by our techniques to identify which JavaScripts can be safely pruned might degrade crawling throughput. Based on any anomalies observed in testing, we can then revise our approach appropriately.

3.6 Dissemination

We will pursue a multi-pronged approach to ensure wide dissemination of the project's outcomes.

Conversations with developers of Brozzler and Browsertrix. First, throughout the duration of this project, we will have periodic conversations with our advisors from the Internet Archive and the Webrecorder project. Via these conversations, we will strive to help facilitate the incorporation into Brozzler and Browsertrix of the methods that we develop in this project.

Recommendations for other archivists. Second, to ensure broader and more sustained applicability of our work, we will publish an open technical document that summarizes our recommendations for how web archives should assess and filter JavaScripts. Developers and administrators of other web archives, beyond the two mentioned above, will be able to consult this document for how they should modify their tools and operational practices. We will post this document on a dedicated website for this project that we will host on GitHub. To ensure that this document reflects feedback from a broad sample of the web archiving community, we will advertise preliminary versions of it on mailing lists such as the listserv of the National Digital Stewardship Alliance (NDSA).

Research publications. Third, we will seek to publish papers describing our work at venues like the ACM/IEEE Joint Conference on Digital Libraries and the International Journal on Digital Libraries. We will also aim to publish papers summarizing our work at conferences in the computer systems and networking research areas, such as USENIX Symposium on Networked Systems Design and Implementation and ACM Symposium on Operating System Principles. PI Madhyastha's group has a long history of publishing at these venues. By advertising our project at these venues, we hope that others with deep-rooted expertise in JavaScript will become aware of the problems we are tackling in this project, and some of them might consider building on our efforts.

Talks. Lastly, we also hope to give presentations about our work at venues that are at the intersection of technology and web archival. For example, we will strive to present our work at the Code4Lib conference [7], the annual meeting of the Society of American Archivisits [14], the Web Archiving and Digital Libraries (WADL) workshop [17], and the Digital Library Federation (DLF) Forum [9]. Presenting our work to the librarians and archivists who attend these venues will be key to ensure broad dissemination of the project's outcomes.

4 Diversity Plan

All members of our advisory board are from institutions which have ample resources to archive the web at scale. However, not all libraries and museums have this luxury [13], and they face an uphill battle to keep up with the growing presence of born-digital information [34]. We believe that our efforts towards making web archiving more efficient will be a significant step towards addressing this problem. By advertising our work in all the ways described earlier in the proposal, we plan to connect with cash-strapped institutions and understand how our methods will need to be folded into services like Archive-It and Browsertrix Cloud in order to benefit them.

In addition, we will strive to ensure that we study pages from a diverse corpus of sites, so that our findings and recommendations are not applicable only to a narrowly defined category of websites. Our choice of studying the EOT and CARTA datasets is influenced by this goal, as these two collections have snapshots of pages from very different site types; the EOT data has pages from federal websites, while the CARTA data includes at-risk web-based art materials. Accounting for a diverse collection of pages in our analyses is vital to ensure that our recommended changes to web archives do not myopically apply to only a fraction of the web, but are broadly applicable.

Lastly, we plan to leverage this project to expose undergraduate and graduate students in computer science at USC to opportunities for applying their technical knowledge to the field of archiving. Today, many computer science graduates are highly motivated to apply their education for social good and are disillusioned of the traditional career as a software developer at a big tech company. We anticipate that the proposed project will be of great interest to these students looking for careers that contribute to the world at large. In particular, we will invite our advisors and others from their organizations to talk about the threats that society faces in preserving human knowledge stored digitally. We envision that these talks will inspire the students to work on problems like the one we tackle in this project, and some of them will develop creative solutions that were previously unforeseen.

5 Project Results

Deliverables. The primary outcome from our work will be our proposed modifications to the browser-based web crawlers used by the Internet Archive and the Webrecorder project, which respectively oversee the Archive-It and Browsertrix Cloud services. We will work closely with the teams overseen by our advisors from these organizations – Jefferson Bailey and Ilya Kreymer – to ensure that our proposals are practical. Once the takeaways from our work are folded into their crawlers, the large number of institutions which use Archive-It and Browsertrix Cloud will benefit from having their archived pages preserved with greater fidelity. Budget-constrained libraries and museums which use these services will also either see a reduction in their archival costs or be able to afford to archive more pages than they can today.

In addition, a critical component of our work will be to validate that our changes do not cause unintended impacts, e.g., worsen page fidelity in new ways. Therefore, as described earlier, we will develop a principled methodology for testing the fidelity of pages crawled. This testing framework will be a key outcome of our work, and will likely prove useful in other similar efforts too.

Expected impact. Our work will help web archives improve in two significant ways.

First, our work will help ensure that archived pages more closely approximate their original versions found on the web. This will include both how the page is rendered and the ways in which users can interact with page after it has been rendered.

Second, due to the widespread usage of JavaScript on modern web pages, needing to crawl, execute, save, and serve JavaScripts accounts for a sizeable fraction of a web archive's costs. By providing web archives with the capability whereby they can choose whether or not to fetch JavaScript that makes no contribution to preserving the fidelity of archived pages, we will enable them to significantly reduce their operational costs. Consequently, users of web archives can crawl and save many more pages than they can today under their budgetary constraints.

Improvements in web archives in both of these respects is significant because a key use of archived web pages is to help combat link rot. Prior studies [22, 28, 33, 38, 39] have found that many of the links included on a page often do not work even a few years after the page's creation. Therefore, when a user encounters a broken web link, their only recourse is to lookup archived copies of the pages that previously existed at this link. In such cases, our work will help increase the chances that an archived copy does exist and that copy is indeed usable. Else, the care that authors of web pages put into including a carefully curated set of links will go to waste over time, thereby robbing visitors of a page of the context that the page's author meant to provide them.

Ideas. A broader impact of our work will be to draw awareness towards the various ways in which archived pages fundamentally differ from pages on the live web. Today, archiving a page is considered to constitute loading the page in a browser and saving all resources fetched by the browser during the page load. Users can then load this copy of the page in their browser as they would any page on the web.

The principles put forth by our work will call for a rethink of both of these conventional practices when it comes to JavaScript-heavy pages. On the one hand, our work will make a case for not saving *all* the JavaScript that is fetched when crawling a page, but only that subset which serves any utility on an archived copy of the page. On the other hand, we will show that JavaScripts on an archived page cannot be executed by web browsers in a manner akin to how they are executed on live pages; to ensure that non-determinism does not have unintended impact on page fidelity, the browser must instead reuse some of the characteristics of the environment in which the page had been originally crawled. In general, we will draw awareness to the fact that archivists need to carefully think about which kinds of user agents and client environments are they archiving pages for.

Schedule of Completion

Year 1

Activity	Months			
	1–3	4–6	7–9	10–12
Conduct conversations with advisory board members to get feedback		■		■
Download random sample of pages from End Of Term and CARTA collections	■			
Apply concolic execution to understand impact of client characteristics on fidelity	■	■		
Develop methodology for checking visual and functional fidelity		■		
Run methods developed on EOT and CARTA datasets; revise methods accordingly			■	
Revise implementations to fix any issues detected during testing				■
Disseminate work done over the first year, particularly notable examples where JavaScript results in violations of page fidelity				■

Year 2

Activity	Months			
	1–3	4–6	7–9	10–12
Conduct conversations with advisory board members to get feedback		■		■
Invite two advisors to give talks at USC	■			
Study what it takes to modify Brozzler and Browsertrix so that, when pages are archived, client characteristics captured when crawling are reused when archived pages are later loaded	■			
Automate identification of filter list which can be used to prune out JS code which will not function on archived pages		■		
Reach out to libraries who have limited budget for web archiving and understand their constraints		■		
Modify crawlers to not fetch JavaScripts files which match filter list			■	
Write and submit paper summarizing work done over the first two years				■
Give presentations at venues focused on intersection of technology and archiving			■	■

Year 3

Activity	**Months**			
	1–3	4–6	7–9	10–12
Conduct conversations with advisory board members to get feedback		■		■
Invite two advisors to give talks at USC	■			
Publish open technical document summarizing our recommendations for archiving JavaScript-heavy pages	■			
Incorporate feedback from advisors, and address issues detected during testing	■			
Run test harness to detect any unintended impact on page fidelity and fix any problems		■		
Push changes to crawlers to open-source repositories			■	
Present our work to researchers at other institutions as well as to many web archives				■

Digital Products Plan

Types of Digital Products

This project will result in the following digital products.

First, our work will result in modified versions of two browser-based crawlers: Brozzler and Browsertrix. We do not anticipate that our modified versions of these crawlers will be ready for use in production. However, our changes will serve as a reference for the primary developers of these crawlers, with whom we will be in contact via our advisors from the respective organizations.

Second, as described in the project work plan, we will develop a principled framework for evaluating the fidelity of a collection of archived pages. This testing framework will be a key outcome of our work, and will likely prove useful in other similar efforts too.

Third, we will publish a catalogue of pages in which we have observed that JavaScript's non-deterministic execution results in violations of fidelity. We envision that most of these pages will be from the End Of Term and CARTA datasets. For such pages, we will simply include a reference to the appropriate page snapshot in those collections. For other pages, we will host our crawled copy of the page in the WARC format.

Finally, we will publish an open technical document which summarizes our recommendations for how web archives should deal with JavaScript. The final version of this document will reflect all the feedback we receive from archivists on preliminary versions.

Availability and Access

We will make all of our data and code publicly available on a website specifically hosted for this project on GitHub. We will use a permissive license such as the Apache License 2.0 (https://www.apache.org/licenses/LICENSE-2.0), which will enable others to build on our work. Since our work does not involve any privacy-sensitive data, we will not need to anonymize any data prior to sharing it.

Sustainability

We will use standard formats for data (e.g., WARC) and metadata (e.g., CSV and JSON) to ensure that they will be interpretable in the long-term. Since our project website will be hosted on GitHub, all the data and code that we will post there will continue to remain accessible beyond the duration of the project. When applicable, we will also contribute our modifications to Brozzler and Browsertrix back to the public repositories on GitHub for these crawlers. We will work with our advisory board members to get our open technical document published at a site such as https://github.com/iipc/awesome-web-archiving or https://specs.webrecorder.net/.

Data Management Plan

In this document, we describe the types of data that we will collect or generate as part of this project, as well as how we will manage, share, preserve, document, and enable reuse of this data.

Types of Data

Most of the data that we will study in this project will come from existing collections of archived page snapshots, namely the End of Term and CARTA datasets. In addition, we will crawl thousands of pages from a wide variety of sites on the live web; unlike our analysis of existing collections, we can augment the crawling process to gather additional information for these pages. We will use existing browser-based crawlers such as Brozzler and Browsertrix, as well as using our modified versions of them. We will recrawl pages roughly once a year, and we expect the size of the crawled data over the lifetime of the project to be in the order of 10–100 gigabytes.

Data Confidentiality

We do not plan to collect any personally identifiable information as part of our research. Since this project does not involve any data which has privacy concerns, no anonymization will be necessary before release. We do not expect that any of the research documents, presentations, experimental measurements, course materials to be private and will release them under a open use and extend model.

In addition, as none of the research components in our proposed project constitutes human subjects research, none of the data that we are collecting as part of this project is subject to review by an Institutional Review Board.

Data Reusability

We intend to publish the data associated with our project on a project website hosted on GitHub. We will use standard data formats for all the data generated during our project. For pages that we crawl from the web, we will store all page resources in the WARC format. We will store all metadata in either CSV or JSON format, for which extensive documentation and code support exists.

In addition to data, we believe that the software that we develop in this project will be very useful for other researchers as well as the providers and users of web archival services. We will release all related software under suitable licensing models (e.g., BSD, Apache) that allow both academic and industry endeavors to build on our work. For any changes that we make to code written by others (e.g., Brozzler and Browsertrix), we will push our changes back to their public code repositories.

Documentation

For all page snapshots that we analyze from the End Of Term and CARTA datasets or from the Internet Archive, we will publish pointers to the relevant page snapshots in those collections. For pages that we ourselves crawl from the live web, we will store the crawled pages in the WARC format so that others can repeat our analyses. We will publicly share the code for all of our analyses to ease reproducibility.

Data Preservation

Data that we collect as part of our experiments in this project will be retained for at least the duration of the project. Except for content that we analyze from others' page collections (e.g., End Of Term and CARTA), we plan to archive the results of our experiments on servers at USC as well as on GitHub. We will also host copies of our research data on sites such as https://datacite.org/ and https://www.re3data.org/ to ensure long-term preservation.

Adhering to Data Management Plan

The PI will work with the graduate students to ensure that all materials described above are made freely available for download in a timely manner. Immediately after academic publication of the relevant research results, we will release our source code as well as the data from our evaluations.

Organizational Profile

Mission statement. *Adopted by the USC Board of Trustees, February 1993, Source: https://about.usc.edu/policies/mission-statement/* The central mission of the University of Southern California is the development of human beings and society as a whole through the cultivation and enrichment of the human mind and spirit. The principal means by which our mission is accomplished are teaching, research, artistic creation, professional practice and selected forms of public service.

Our first priority as faculty and staff is the education of our students, from freshmen to postdoctorals, through a broad array of academic, professional, extracurricular and athletic programs of the first rank. The integration of liberal and professional learning is one of USC's special strengths. We strive constantly for excellence in teaching knowledge and skills to our students, while at the same time helping them to acquire wisdom and insight, love of truth and beauty, moral discernment, understanding of self, and respect and appreciation for others.

Research of the highest quality by our faculty and students is fundamental to our mission. USC is one of a very small number of premier academic institutions in which research and teaching are inextricably intertwined, and on which the nation depends for a steady stream of new knowledge, art, and technology. Our faculty are not simply teachers of the works of others, but active contributors to what is taught, thought and practiced throughout the world.

Governance structure. PI Madhyastha is a faculty member in the Department of Computer Science, which is in USC's Viterbi School of Engineering. The Dean of Engineering reports to the University Provost, who in turn reports to the University President, who is appointed by the Board of Trustees.

Service area. USC is pluralistic, welcoming outstanding men and women of every race, creed and background. We are a global institution in a global center, attracting more international students over the years than any other American university. And we are private, unfettered by political control, strongly committed to academic freedom, and proud of our entrepreneurial heritage.

The university currently enrolls 49,000 undergraduate and graduate students (source: https://about.usc.edu/facts/). Within the Computer Science department, 1,063 undergraduate and 2,291 graduate students are currently enrolled (source: https://www.cs.usc.edu/about/).

History of Computer Science and Engineering at USC. The USC Viterbi School of Engineering offers one of the best environments for interdisciplinary research and international collaboration in the nation. Our department has a record of innovation that includes the Domain Name System and the TCP/IP protocols, the invention of DNA computing, and historic interdisciplinary studies relating brains, machines and mathematics. A former student wrote one of the first computer viruses and led the theoretical study of the computer virus concept.

A detailed history about the Viterbi School of Engineering is at https://viterbischool.usc.edu/history/.