Summer 8-2023

Assessing the Prevalence and Archival Rate of URIs to Git Hosting Platforms in Scholarly Publications

Emily Escamilla
Old Dominion University, emily.l.escamill@gmail.com

Recommended Citation

Escamilla, Emily. "Assessing the Prevalence and Archival Rate of URIs to Git Hosting Platforms in Scholarly Publications" (2023). Master of Science (MS), Thesis, Computer Science, Old Dominion University, DOI: 10.25777/sczh-wh93
https://digitalcommons.odu.edu/computerscience_etds/142

ASSESSING THE PREVALENCE AND ARCHIVAL RATE OF URIS TO GIT HOSTING PLATFORMS IN SCHOLARLY PUBLICATIONS

by

Emily Escamilla
B.S. May 2021, Liberty University

A Thesis Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

COMPUTER SCIENCE

OLD DOMINION UNIVERSITY
August 2023

ABSTRACT

ASSESSING THE PREVALENCE AND ARCHIVAL RATE OF URIS TO GIT HOSTING PLATFORMS IN SCHOLARLY PUBLICATIONS

Emily Escamilla
Old Dominion University, 2023
Director: Dr. Michael L. Nelson

The definition of scholarly content has expanded to include the data and source code that contribute to a publication. While major archiving efforts to preserve conventional scholarly content, typically in PDFs (e.g., LOCKSS, CLOCKSS, Portico), are underway, no analogous effort has yet emerged to preserve the data and code referenced in those PDFs, particularly the scholarly code hosted online on Git Hosting Platforms (GHPs). Similarly, Software Heritage is working to archive public source code, but there is value in archiving the surrounding ephemera that provide important context to the code while maintaining their original URIs. In current implementations, source code and its ephemera are not preserved, which presents a problem for scholarly projects where reproducibility matters. To quantify the scope of this issue, we analyzed the use of GHP URIs in the arXiv and PMC corpora. In total, there were 253,590 URIs to GitHub, SourceForge, Bitbucket, and GitLab repositories across the 2.64 million publications. Authors have increasingly included GHP URIs in scholarly publications and, in 2021, one in five arXiv publications included a GitHub URI. Next, we analyzed the archival coverage of scholarly GHP URIs in Web archives and Software Heritage. Overall, 79.15% of GHP URIs were archived in the Web archives while only 62.06% of GHP URIs were archived in Software Heritage. We used a machine learning classifier to identify other Open Access Data and Software (OADS) URIs outside of the four GHPs previously studied. We found almost 50,000 unique OADS hostnames and more non-GHP OADS URIs than GHP URIs. The prevalence of OADS URIs and vast number of unique hostnames points to the utility of a classifier to identify OADS URIs as opposed to manual enumeration. Lastly, we found a statistically significant relationship between the popularity of a GitHub repository as determined by engagement metrics and archival coverage indicating that less popular repositories less likely to be archived and, thus, more vulnerable to being unrecoverable. The growing use of GHPs in scholarly publications points to an urgent and growing need for dedicated efforts to archive their holdings in order to preserve research code and its scholarly ephemera.

To my father, Brian Vogt. It looks like all of our late night conversations in the kitchen paid off.

ACKNOWLEDGMENTS

First, I would like to thank God. He has given me the opportunity and strength to pursue a Master's degree and complete this thesis. Soli Deo gloria.

Every former grad student talks about the importance of a great advisor and I was lucky enough to get two of the best. Dr. Nelson and Dr. Weigle always pushed me to pursue excellence and gave me skills I will use for the rest of my life. Dr. Nelson taught me how to leave the undergrad mentality of neat answers and lean into the mindset of constantly asking "why?" that is necessary for research. Dr. Weigle taught me both the importance of effective communication and how to do it through data visualizations and countless rounds of feedback on papers. This thesis would not have been possible without their expertise and guidance.

I would also like to thank Dr. Justin Brunelle for introducing me to the field of Web archiving and so persistently recommending that I pursue grad school at ODU. His guidance and support over the years is a large part of why I am the computer scientist that I am today.

This work would not have been possible without the funding of the Alfred P. Sloan Foundation in support of the CoSAI project. I would also like to thank my CoSAI team members, Dr. Martin Klein, Vicky Rampin, Talya Cooper, and David Calano. Their passion for archiving scholarly code was infectious and this thesis could not have happened without their collaboration, insights, and feedback. It has been an honor to work with them.

Lastly, this would not have been possible without the love and support of my husband, Chris Escamilla. He cheered me on every step of the way and did more than his fair share of dishes so I could write yet another paper. Through countless conversations about Web archiving and my research, he, undoubtedly, knows far more about Web archiving than he ever wanted to know.

TABLE OF CONTENTS

Page

LIST OF TABLES x

LIST OF FIGURES xi

Chapter

1. INTRODUCTION 1

2. BACKGROUND 6
   - 2.1 GIT VS GITHUB 6
   - 2.2 ARCHIVING SCHOLARLY CONTENT 13

3. RELATED WORK 22

4. MEASURING THE PREVALENCE OF GHP URIS IN SCHOLARLY PUBLICATIONS 26
   - 4.1 METHODOLOGY 26
   - 4.2 RESULTS 29
   - 4.3 DISCUSSION 37
   - 4.4 SUMMARY 38

5. QUANTIFYING THE ARCHIVAL RATE OF SCHOLARLY GHP URIS 39
   - 5.1 METHODOLOGY 39
   - 5.2 RESULTS 41
   - 5.3 DISCUSSION 49
   - 5.4 SUMMARY 50

6. IDENTIFYING NON-GHP OADS URIS IN SCHOLARLY PUBLICATIONS 51
   - 6.1 METHODOLOGY 52
   - 6.2 RESULTS 53
   - 6.3 DISCUSSION 57
   - 6.4 SUMMARY 59

7. RELATIONSHIP BETWEEN REPOSITORY POPULARITY AND ARCHIVAL RATE 60
   - 7.1 METHODOLOGY 60
   - 7.2 RESULTS 62
   - 7.3 DISCUSSION 69
   - 7.4 SUMMARY 70

8. FUTURE WORK AND CONCLUSIONS 71

Page

REFERENCES..73

APPENDICES

A. MEMGATOR API REQUEST AND RESPONSE 78
B. SOFTWARE HERITAGE API: REPOSITORY-LEVEL GHP URI 80
C. SOFTWARE HERITAGE API: DEEP LINK GHP URI 85
D. REGULAR EXPRESSIONS TO EXTRACT SOURCEFORGE URIS 86
E. REGULAR EXPRESSIONS TO EXTRACT GITLAB URIS............................ 87
F. REGULAR EXPRESSIONS TO EXTRACT BITBUCKET URIS 88
G. REGULAR EXPRESSIONS TO EXTRACT GITHUB URIS 89
H. GITHUB API REQUEST AND RESPONSE.. 90

VITA...96

LIST OF TABLES

Table Page

1. Repository Platforms 24
2. Sentences containing OADS and non-OADS URLs. 25
3. Five journals with the most articles in the PMC corpus 27
4. Examples of URIs that were considered to be out of scope for our study 28
5. Number of references to each GHP in the arXiv and PMC corpora 29
6. Top 10 publications with the most URIs to a single GHP 33
7. Number of URIs to each GHP and the percentage of all GHP URIs. 39
8. Percent of all mementos returned from each of the 12 Web archives. 43
9. Percent of URIs for each GHP and category 44
10. The top 15 most common hostnames for non-GHP OADS URIs and their frequencies. 58
11. Population statistics for each population for the three engagement metrics. 63
12. Results of the statistical analysis for each of the three engagement metrics 64

LIST OF FIGURES

Figure Page

1. TensorFlow implementation of DC-GAN 2
2. Software as methodology 3
3. GitHub 404 4
4. Git vs GitHub 7
5. GitHub Issues page 8
6. Comparing mementos of the Issues page: Top of page 10
7. Comparing mementos of the Issues page: First entries 11
8. Comparing mementos of the Issues page: Last entries 12
9. Comparing mementos of the main repository page 13
10. A memento of the TensorFlow GitHub repository from 2015 15
11. A memento of the TensorFlow GitHub repository from 2023 16
12. The live TensorFlow GitHub repository 17
13. A Software Heritage capture of the Tensorflow GitHub repository 19
14. Zenodo URI Search 20
15. The average number of URIs per publication over time 30
16. The percentage of arXiv publications with a URI to a repository platform over time 31
17. The percentage of PMC publications with a URI to a repository platform over time 32
18. If a publication links to GHP, how many links does it have? 35
19. Mapping the number of links to a GHP by discipline for the arXiv corpus 37
20. Results of running the three tests 42
21. Archival of GHP URIs 45
22. Archival of rotten GHP URIs 47

Figure Page

23. Time delta between a URI being referenced and captured by Software Heritage.............. 48
24. Time delta between a URI being referenced and captured by Software Heritage.............. 49
25. Citation of the ScienceDirect publication.. 52
26. Reference for a ScienceDirect publication 52
27. Average number of URIs per arXiv pre-print by publication date................................. 54
28. Percentage of URIs by category and publication date 55
29. Total number of GHP URIs and non-GHP OADS URIs by publication date. 56
30. Hostname frequency 57
31. Engagement metrics in GitHub UI 61
32. The distribution of forks in relation to Software Heritage archival status 64
33. The distribution of subscribers in relation to Software Heritage archival status 65
34. The distribution of stargazers in relation to Software Heritage archival status 66
35. The distribution of forks in relation to Web archive archival status 67
36. The distribution of subscribers in relation to Web archive archival status 68
37. The distribution of stargazers in relation to Web archive archival status.......................... 69

CHAPTER 1

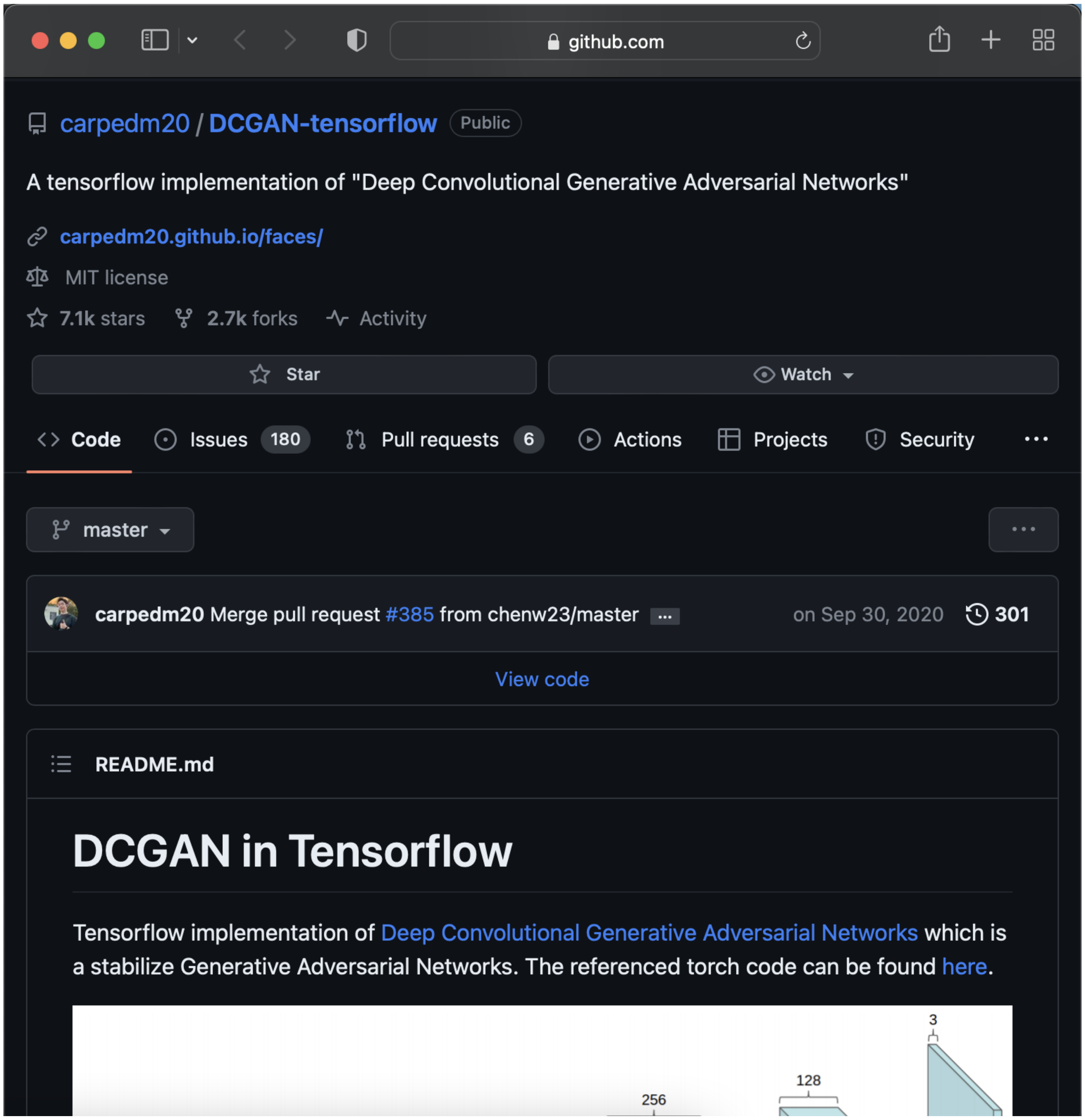
INTRODUCTION

“Reproducibility is obtaining consistent results using the same input data; computational steps, methods, and code; and conditions of analysis” [1] and it is a cornerstone of scientific research. Reproducibility is not to be confused with replicability or repeatability, two terms that are often, but erroneously, used interchangeably. To maintain consistency in terminology, we will adopt the definitions of reproducibility, replicability, and repeatability from the same publication. “Replicability is obtaining consistent results across studies aimed at answering the same scientific question, each of which has obtained its own data” [1]. Repeatability is demonstrated when precision is attained in repeated measurements of a single entity under the exact same conditions (laboratory, equipment, conditions, etc.). The ability to reproduce a study in order to verify or build on the results is an important part of a thriving scientific community and is contingent on access to the original data and methodology. In 2015, Radford et al. [2] published an article introducing deep convolutional generative adversarial networks (DC-GAN), a class of convolutional networks that works with unsupervised learning. The original Torch implementation of DC-GAN is available on GitHub[1] and was created by Soumith Chintala,[2] one of the co-authors of the original publication. The original DC-GAN implementation was modified to be used with TensorFlow, an alternative deep learning framework, and hosted on GitHub[3] with links to the original publication and original code as shown in Figure 1. In 2018, Tabassi et al. [3] published an article that used DC-GAN in a workflow to create a dataset to test the accuracy of software product. Tabassi et al. referenced the original publication and included a link to the TensorFlow implementation, the exact implementation they leveraged in their workflow. This is one example of researchers building on the software products created by previous research. If the original code had not been available in a public repository, Tabassi et al. would not have been able to implement DC-GAN in their workflow without significant time replicating what had already been created by the previous researchers.

[1]`https://github.com/soumith/dcgan.torch`

[2]`https://github.com/soumith`

[3]`https://github.com/carpedm20/dcgan-tensorflow`


github.com

carpedm20 / DCGAN-tensorflow Public

A tensorflow implementation of "Deep Convolutional Generative Adversarial Networks"

carpedm20.github.io/faces/

MIT license

7.1k stars 2.7k forks Activity

Star Watch

Code Issues 180 Pull requests 6 Actions Projects Security

master

carpedm20 Merge pull request #385 from chenw23/master ... on Sep 30, 2020 301

View code

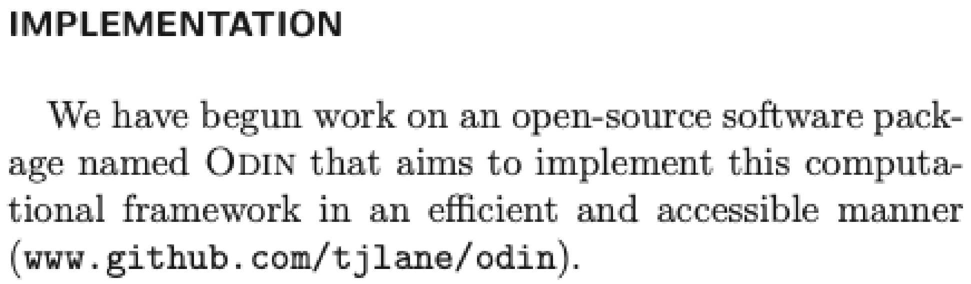
README.md

DCGAN in Tensorflow

Tensorflow implementation of Deep Convolutional Generative Adversarial Networks which is a stabilize Generative Adversarial Networks. The referenced torch code can be found here.

Fig. 1. TensorFlow implementation of DC-GAN. The TensorFlow implementation of the original Torch DC-GAN implementation includes links to the original publication and original code hosted in GitHub.

Not all researchers are as fortunate. Lane et al. [4] published an arXiv pre-print that references a GitHub repository[4] as the implementation of the framework proposed in the article (Figure 2). However, the repository is no longer available on the live Web as shown in Figure 3. Fortunately, both the Internet Archive and Software Heritage, two archives that will be further explained in Chapter 2, archived the repository while it was still alive. The Internet Archive captured the repository three times from August 19, 2017 to November 17, 2020[5] with the latest capture showing that the repository was no longer publicly available. Software Heritage captured the repository 11 times from August 4, 2015 to February 27, 2020.[6] If Internet Archive and Software Heritage had not captured the repository, future researchers would not have been able to access the implementation of the methodology detailed in the article.


IMPLEMENTATION

We have begun work on an open-source software package named ODIN that aims to implement this computational framework in an efficient and accessible manner (`www.github.com/tjlane/odin`).

Fig. 2. Software as methodology. The authors indicate that the exact implementation of the methodology detailed in the article can be found at the indicated URI.

[4]`https://github.com/tjlane/odin`

[5]`https://web.archive.org/web/20200000000000*/https://github.com/tjlane/odin`

[6]`https://archive.softwareheritage.org/browse/origin/visits/?origin_url=https://github.com/tjlane/odin`

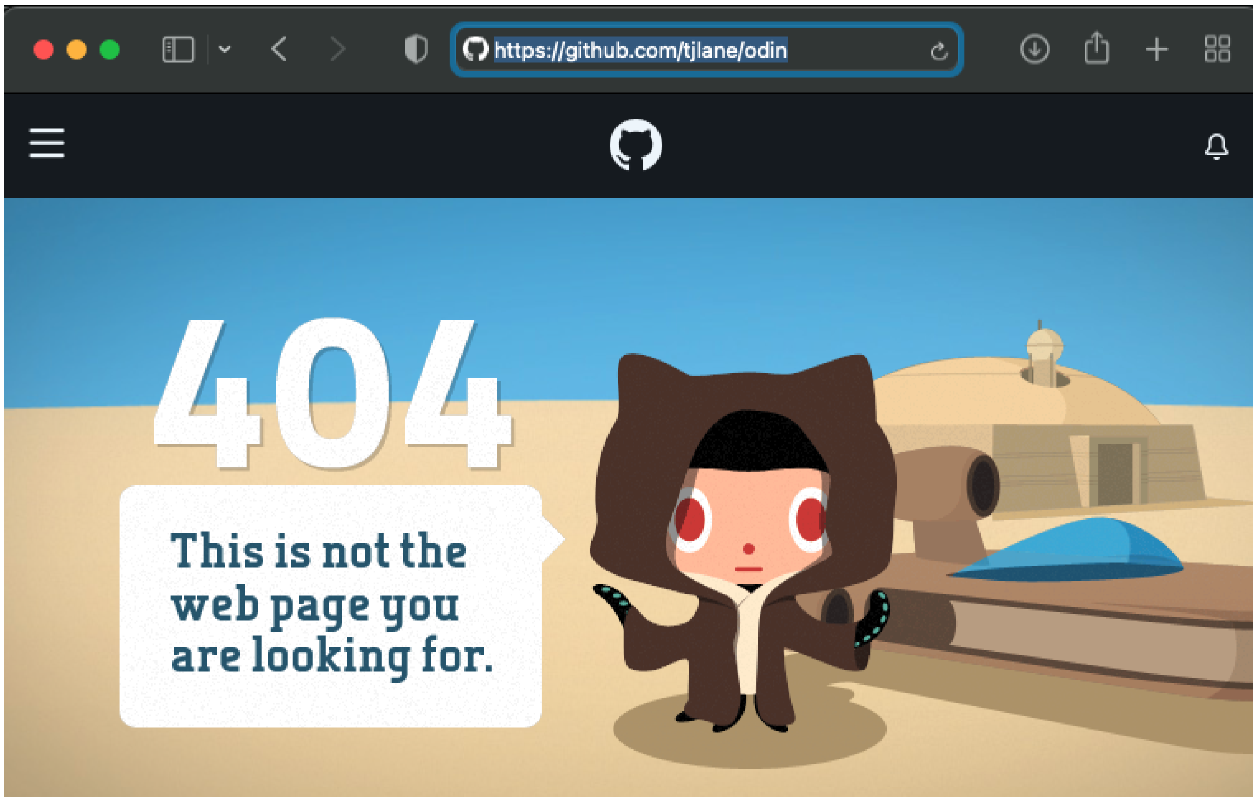

https://github.com/tjlane/odin
404
This is not the web page you are looking for.

Fig. 3. GitHub 404. The repository at `github.com/tjlane/odin` is no longer publicly available on the live Web.

Scientific researchers like Tabassi et al. and Lane et al. are increasingly including URLs to source code repositories to supplement the methodology they detail within their publications. Additionally, open access initiatives and mandates are increasing the availability of open access data and software. However, software availability that satisfies an open access requirement is different than software preservation. If a repository ceases to be available on the live Web, a lack of software preservation has the potential to nullify the advantages gained by open access data and software requirements. Additionally, the need for software preservation spans farther than the computer science discipline. Software development as a research output is increasingly prevalent in research across disciplines from biology to physics to economics. Software products contain the exact methodology used in the study and allow the study to be reproduced both by the original researchers and future researchers.

To address the problems faced by researchers, we investigated the prevalence of URIs to Git Hosting Platforms (GHPs) like GitHub, GitLab, SourceForge, and Bitbucket in scholarly publications. We also investigated the preservation of these scholarly code products in Software Heritage and the Web archives. Lastly, we identified other platforms used by scholars to host open-access data and software (OADS). This work will present the results of our studies in the form of four research questions:

- RQ1: How often do authors include GHP URIs in scholarly articles?
- RQ2: For the GHP URIs identified in scholarly articles, what is the prevalence of these GHP URIs (a) on the live Web, (b) in Software Heritage, and (c) in Web archives?
- RQ3: Outside of GHPs, what OADS URIs are scholars including in their publications?
- RQ4: Does popularity as measured by user engagement affect the likelihood of a GitHub repository being archived?

This thesis has eight chapters. In Chapter 2, we discuss GHPs and the existing efforts to archive software products hosted in GHPs. In Chapter 3, we discuss the need for Web archiving, specifically the need for archiving GHPs. We also survey other studies that have looked at the use of GHP URLs in scholarship. In Chapter 4, we measured the prevalence of GHP URLs in scholarly publications. In Chapter 5, we measured the archival coverage of the GHP URLs included in scholarly publications as well as the public availability of the GHP URLs. In Chapter 6, we identified data and software hosting platforms outside of GHPs that were included in scholarly publications using a classifier. In Chapter 7, we analyzed the correlation between popularity and archival for GitHub repositories included in scholarly publications. In Chapter 8, we identified future work and concluded that GHP URLs are increasingly prevalent in scholarly publications across disciplines resulting in an increased need for archival.

CHAPTER 2

BACKGROUND

In this chapter, we will outline the differences between git and GitHub that motivate this work. We will also discuss the current efforts to archive software products.

2.1 GIT VS GITHUB

Git is a software tool that facilitates a version control system and allows users to manage file versions on a local computer via the command line [5]. Git Hosting Platforms (GHPs) are Web based hosting platforms for git repositories. As shown in Figure 4, GHPs like GitHub support git functionality while also providing users with additional features. GHPs are commonly used by software developers, including researchers, to host software and facilitate collaboration. Examples of GHPs include GitHub, GitLab, Bitbucket, and SourceForge.

To understand the differences between git and GHPs, we will discuss the features that GitHub provides in addition to git functionality. GitHub offers pages for issues, pull requests, wikis, and other additional information that is outside the scope of the git version control system but adds to the development experience. As shown in Figure 5, GitHub Issue pages allow users to report errors and propose additional functionality. These additional pages add context to the development of the repository, but they are considered ephemera which, by definition, means that these pages are temporary and transient. From a Web archiving perspective, preserving repositories hosted in GHPs is focused on preserving the Web representation of the repository within GitHub and archiving the look and feel a user would have experienced at that date and time. This approach preserves the ephemera that would not be captured with a `git clone`, or retrieving a copy of the repository. In terms of Figure 4, a copy of the repository would only retrieve the information in the orange circle.

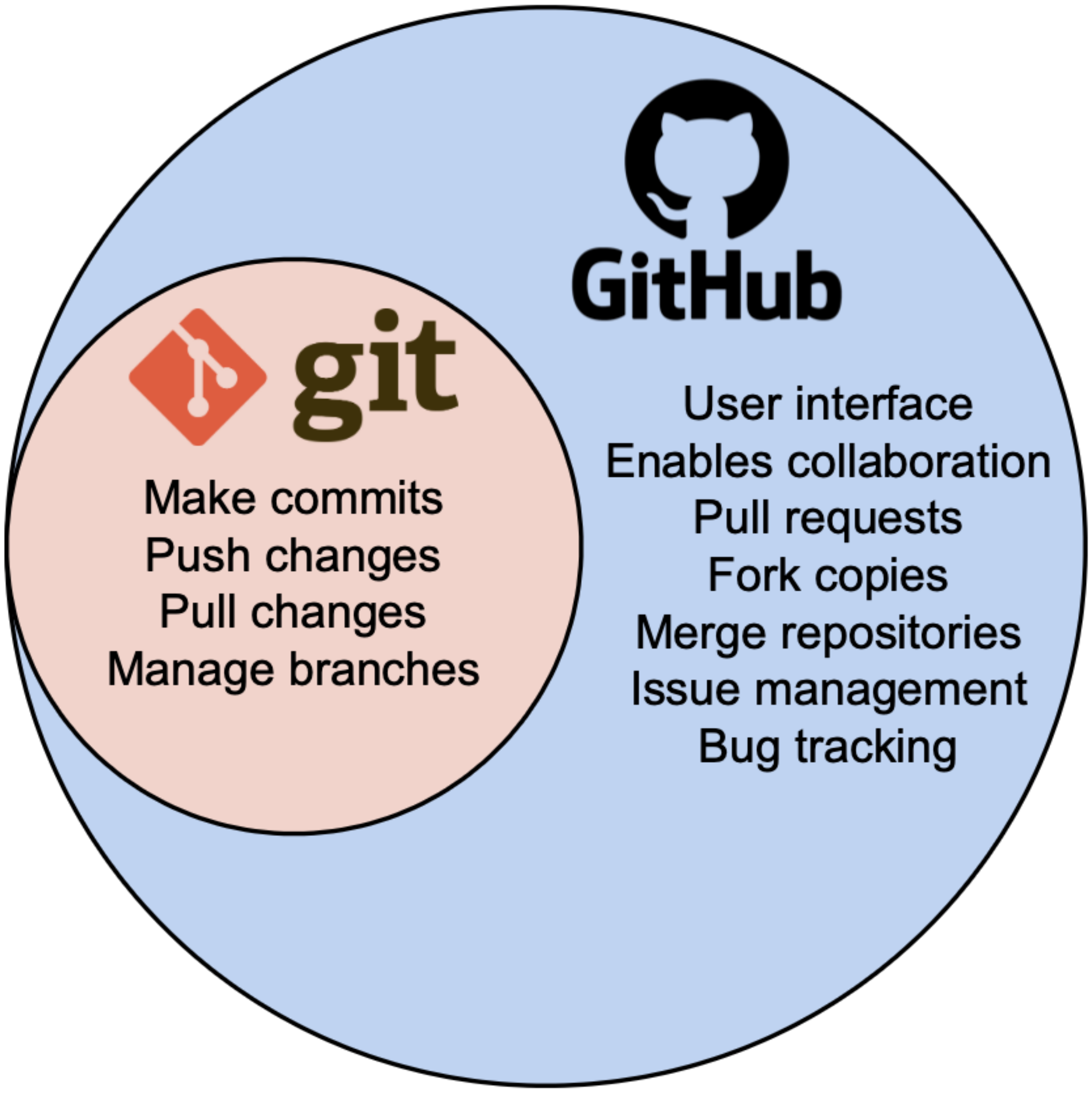
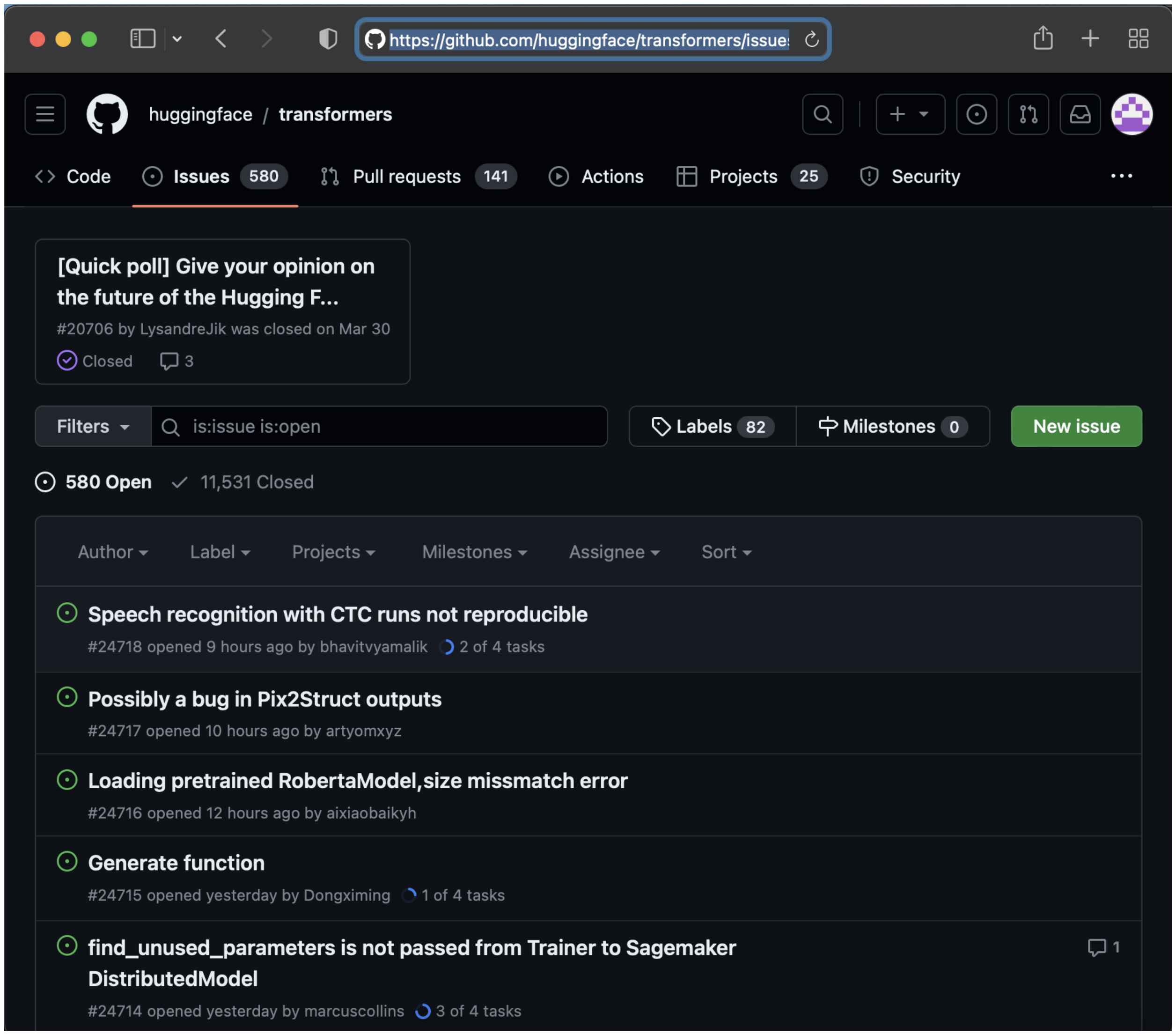


Fig. 4. Git vs GitHub. Comparing the functionality of git and GitHub to understand the differences.



[Quick poll] Give your opinion on the future of the Hugging F...
#20706 by LysandreJik was closed on Mar 30
Closed 3

580 Open ✓ 11,531 Closed

Speech recognition with CTC runs not reproducible
#24718 opened 9 hours ago by bhavitvyamalik 2 of 4 tasks

Possibly a bug in Pix2Struct outputs
#24717 opened 10 hours ago by artyomxyz

Loading pretrained RobertaModel,size missmatch error
#24716 opened 12 hours ago by aixiaobaikyh

Generate function
#24715 opened yesterday by Dongximing 1 of 4 tasks

find_unused_parameters is not passed from Trainer to Sagemaker DistributedModel 1
#24714 opened yesterday by marcuscollins 3 of 4 tasks

Fig. 5. GitHub Issues page. This functionality allows users to report bugs or recommend functionality as shown in the image.

To understand the importance of preserving ephemera, we will look at the GitHub repository for Keras,[1] a "deep learning API written in Python" and, by far, the most popular GitHub repository referenced in the corpora we studied. On June 14, 2023, we created two mementos, or archived Web pages, for the Keras repository: the home page[2] and the first page of issues.[3] After ten days,

[1]https://github.com/keras-team/keras

[2]URI-R: https://github.com/keras-team/keras, URI-M: https://web.archive.org/web/20230614175747/https://github.com/keras-team/keras

[3]URI-R: https://github.com/keras-team/keras/issues, URI-M: https://web.archive.org/web/

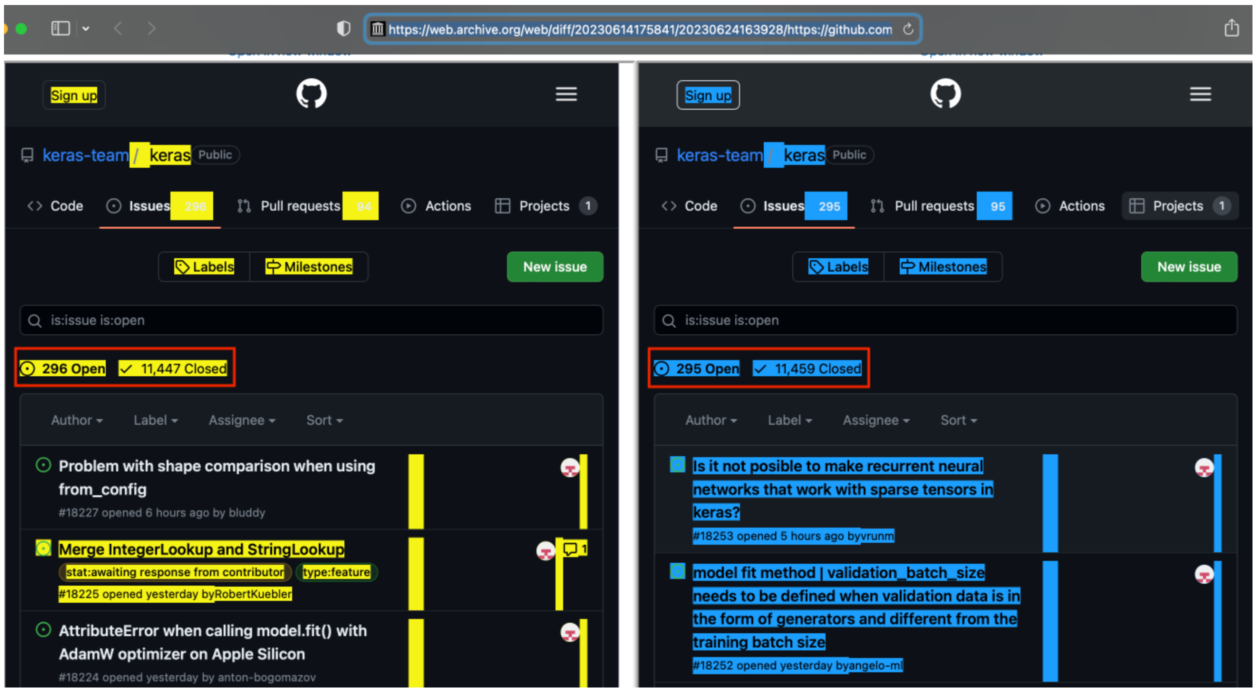
we created two more mementos,[4,5] one for each of the three pages we previously archived. We used the Compare tool [6] from the Internet Archive's Wayback Machine to compare the URI-Ms for each of the three pages.

We compared the mementos created for the Issues page and found numerous changes as shown in Figures 6 to 8. In the ten days between the captures, 12 issues have been closed and 11 issues have been created, as indicated by the number of open and closed issues in Figure 6. On the right side, we see two issues, highlighted in blue, that have been created since the capture on the left. In Figure 7, we can see the addition new issues highlighted in blue on the right side, but we can also see that the issue highlighted in yellow on the right side has been closed since the first memento. All of these changes to the Issues page are reflecting an engaged and active development community. That is also reflected when we notice the number of comments increasing. For example, the issue titled "AttributeError when calling model.fit() with AdamW optimizer on Apple Silicon" on the left side of Figure 7 has no comments, but, ten days later, the same issue now has three comments and a new tag. At the bottom of the Issues page shown in Figure 8, we see more issues that have been closed in the ten days between captures.

20230614175841/https://github.com/keras-team/keras/issues

[4]https://web.archive.org/web/20230624163808/https://github.com/keras-team/keras

[5]https://web.archive.org/web/20230624163928/https://github.com/keras-team/keras/issues



https://web.archive.org/web/diff/20230614175841/20230624163928/https://github.com

keras-team / keras Public

is:issue is:open

296 Open ✓ 11,447 Closed

Problem with shape comparison when using from_config
#18227 opened 6 hours ago by bluddy

Merge IntegerLookup and StringLookup
stat:awaiting response from contributor type:feature
#18225 opened yesterday by RobertKuebler

AttributeError when calling model.fit() with AdamW optimizer on Apple Silicon
#18224 opened yesterday by anton-bogomazov

keras-team / keras Public

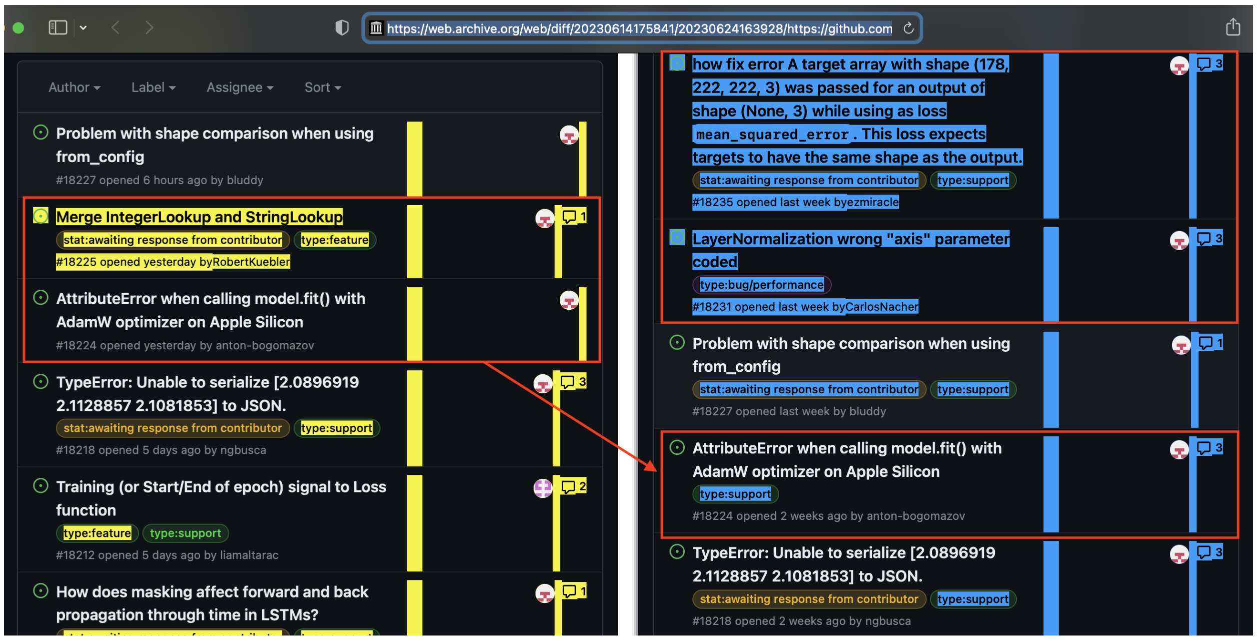
is:issue is:open

295 Open ✓ 11,459 Closed

Is it not posible to make recurrent neural networks that work with sparse tensors in keras?
#18253 opened 5 hours ago by vrunm

model fit method | validation_batch_size needs to be defined when validation data is in the form of generators and different from the training batch size
#18252 opened yesterday by angelo-ml

Fig. 6. Comparing mementos of the Issues page: Top of page. Comparing mementos of the Issues page. Comparison of a memento of the Issues page from June 14, 2023 (`https://github.com/keras-team/keras/issues`) on the left and June 24, 2023 (`https://web.archive.org/web/20230624163928/https://github.com/keras-team/keras/issues`) on the right (`https://web.archive.org/web/diff/20230614175747/20230624163808/https://github.com/keras-team/keras`). This image shows the comparison of the top of the Web page including the number of open and closed issues.


https://web.archive.org/web/diff/20230614175841/20230624163928/https://github.com

Author Label Assignee Sort

Problem with shape comparison when using from_config
#18227 opened 6 hours ago by bluddy

Merge IntegerLookup and StringLookup
stat:awaiting response from contributor type:feature
#18225 opened yesterday byRobertKuebler 1

AttributeError when calling model.fit() with AdamW optimizer on Apple Silicon
#18224 opened yesterday by anton-bogomazov

TypeError: Unable to serialize [2.0896919 2.1128857 2.1081853] to JSON.
stat:awaiting response from contributor type:support
#18218 opened 5 days ago by ngbusca 3

Training (or Start/End of epoch) signal to Loss function
type:feature type:support
#18212 opened 5 days ago by liamaltarac 2

How does masking affect forward and back propagation through time in LSTMs? 1

how fix error A target array with shape (178, 222, 222, 3) was passed for an output of shape (None, 3) while using as loss `mean_squared_error`. This loss expects targets to have the same shape as the output.
stat:awaiting response from contributor type:support
#18235 opened last week byezmiracle 3

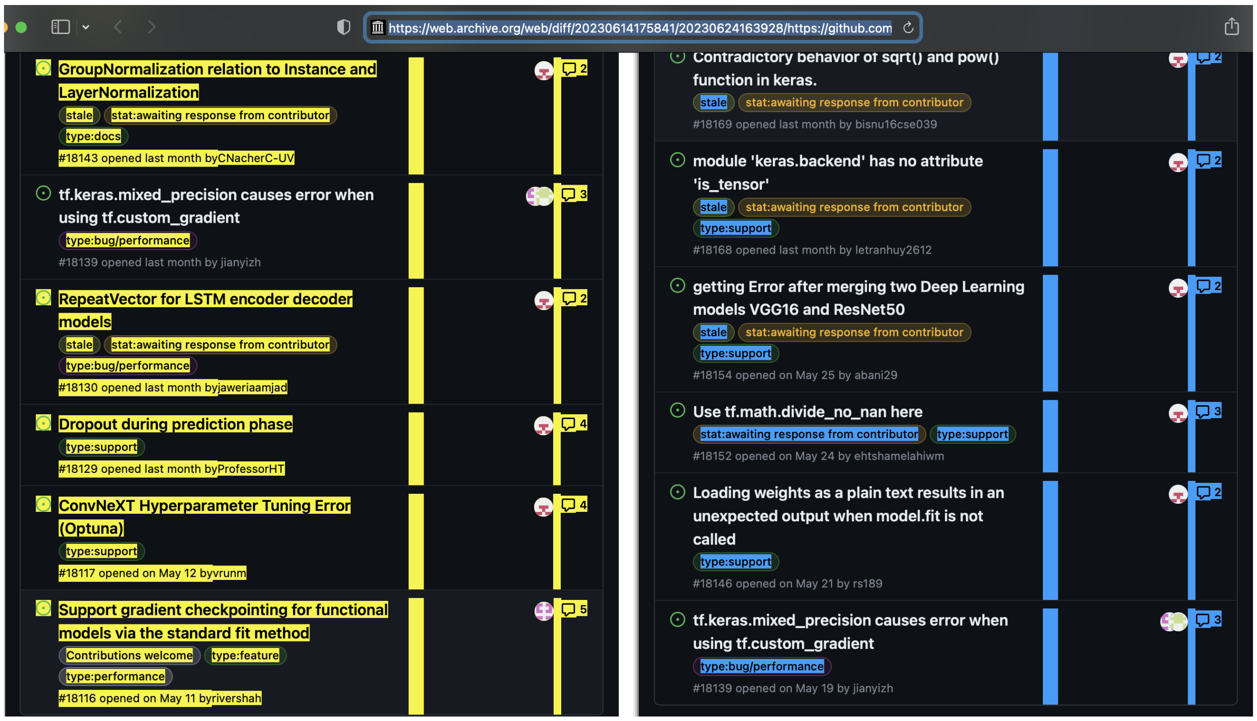
LayerNormalization wrong "axis" parameter coded
type:bug/performance
#18231 opened last week byCarlosNacher 3

Problem with shape comparison when using from_config
stat:awaiting response from contributor type:support
#18227 opened last week by bluddy 1

AttributeError when calling model.fit() with AdamW optimizer on Apple Silicon
type:support
#18224 opened 2 weeks ago by anton-bogomazov 3

TypeError: Unable to serialize [2.0896919 2.1128857 2.1081853] to JSON.
stat:awaiting response from contributor type:support
#18218 opened 2 weeks ago by ngbusca 3

Fig. 7. Comparing mementos of the Issues page: First entries. Comparison of a memento of the Issues page from June 14, 2023 (`https://github.com/keras-team/keras/issues`) on the left and June 24, 2023 (`https://web.archive.org/web/20230624163928/https://github.com/keras-team/keras/issues`) on the right (`https://web.archive.org/web/diff/20230614175747/20230624163808/https://github.com/keras-team/keras`). This image shows the comparison of the first entries of the list of issues.


https://web.archive.org/web/diff/20230614175841/20230624163928/https://github.com

GroupNormalization relation to Instance and LayerNormalization
stale stat:awaiting response from contributor type:docs
#18143 opened last month byCNacherC-UV

tf.keras.mixed_precision causes error when using tf.custom_gradient
type:bug/performance
#18139 opened last month by jianyizh

RepeatVector for LSTM encoder decoder models
stale stat:awaiting response from contributor type:bug/performance
#18130 opened last month byjaweriaamjad

Dropout during prediction phase
type:support
#18129 opened last month byProfessorHT

ConvNeXT Hyperparameter Tuning Error (Optuna)
type:support
#18117 opened on May 12 byvrunm

Support gradient checkpointing for functional models via the standard fit method
Contributions welcome type:feature type:performance
#18116 opened on May 11 byrivershah

Contradictory behavior of sqrt() and pow() function in keras.
stale stat:awaiting response from contributor
#18169 opened last month by bisnu16cse039

module 'keras.backend' has no attribute 'is_tensor'
stale stat:awaiting response from contributor type:support
#18168 opened last month by letranhuy2612

getting Error after merging two Deep Learning models VGG16 and ResNet50
stale stat:awaiting response from contributor type:support
#18154 opened on May 25 by abani29

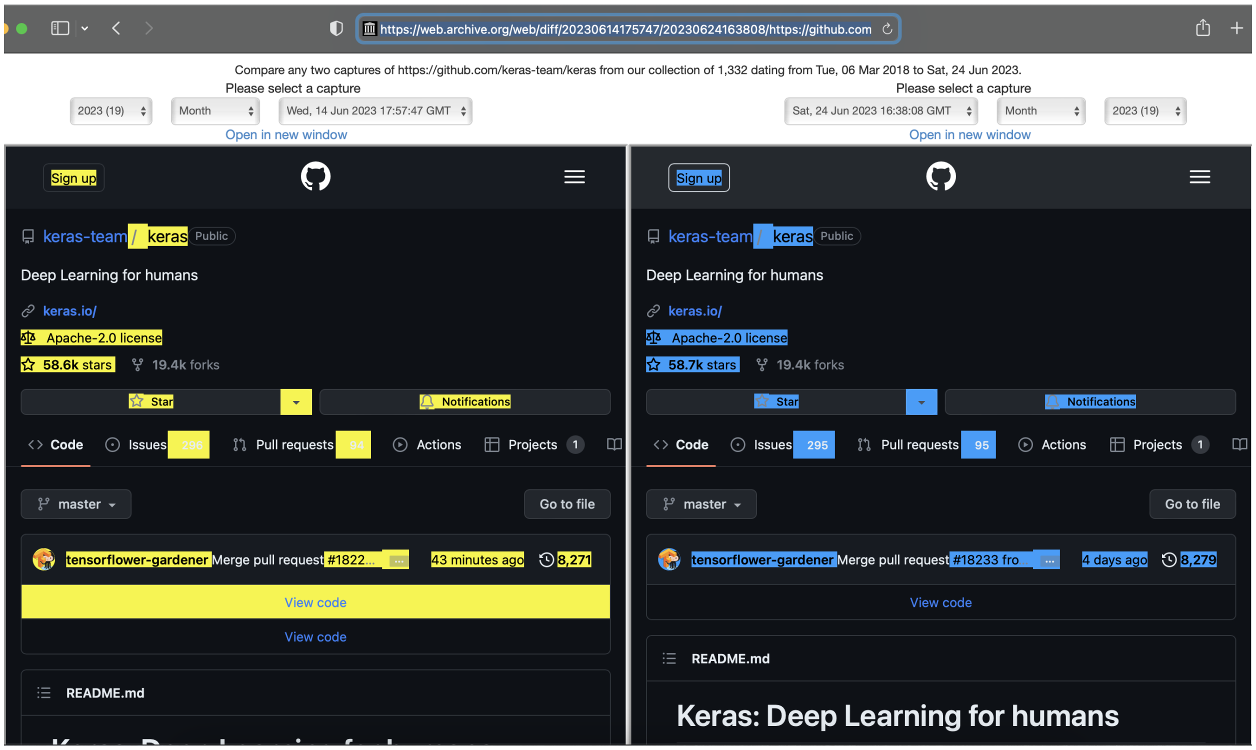
Use tf.math.divide_no_nan here
stat:awaiting response from contributor type:support
#18152 opened on May 24 by ehtshamelahiwm

Loading weights as a plain text results in an unexpected output when model.fit is not called
type:support
#18146 opened on May 21 by rs189

tf.keras.mixed_precision causes error when using tf.custom_gradient
type:bug/performance
#18139 opened on May 19 by jianyizh

Fig. 8. Comparing mementos of the Issues page: Last entries. Comparison of a memento of the Issues page from June 14, 2023 (`https://github.com/keras-team/keras/issues`) on the left and June 24, 2023 (`https://web.archive.org/web/20230624163928/https://github.com/keras-team/keras/issues`) on the right (`https://web.archive.org/web/diff/20230614175747/20230624163808/https://github.com/keras-team/keras`). This image shows the comparison of the last entries of the first page of issues.

Despite the large number of changes reflected in the Issues page, the main repository page reflects very few changes as shown in Figure 9. Changes to the commit identifier and associated commit message are the primary differences between the mementos. We found that five commits were made to the repository between the two captures, but those changes are reflected in the files themselves which are not always archived when the main repository page is archived.



https://web.archive.org/web/diff/20230614175747/20230624163808/https://github.com

Compare any two captures of https://github.com/keras-team/keras from our collection of 1,332 dating from Tue, 06 Mar 2018 to Sat, 24 Jun 2023.

Fig. 9. Comparison of a memento of the main repository page from June 14, 2023 (`https://web.archive.org/web/20230614175747/https://github.com/keras-team/keras`) on the left and June 24, 2023 (`https://web.archive.org/web/20230624163808/https://github.com/keras-team/keras`) on the right.

All of the commits and code changes would be preserved by archiving the code alone. However, the new and closed issues shown in Figures 6 to 8 would not be captured. The Issues page tells a story of the development of the code as well as the community that has created it. Users are able to ask questions, request functionality, and propose changes to be implemented in the code. Archiving the Issues page and other ephemera that surround the code provides the context for the living code product and aids in our knowledge of how the code works and why certain decisions were made.

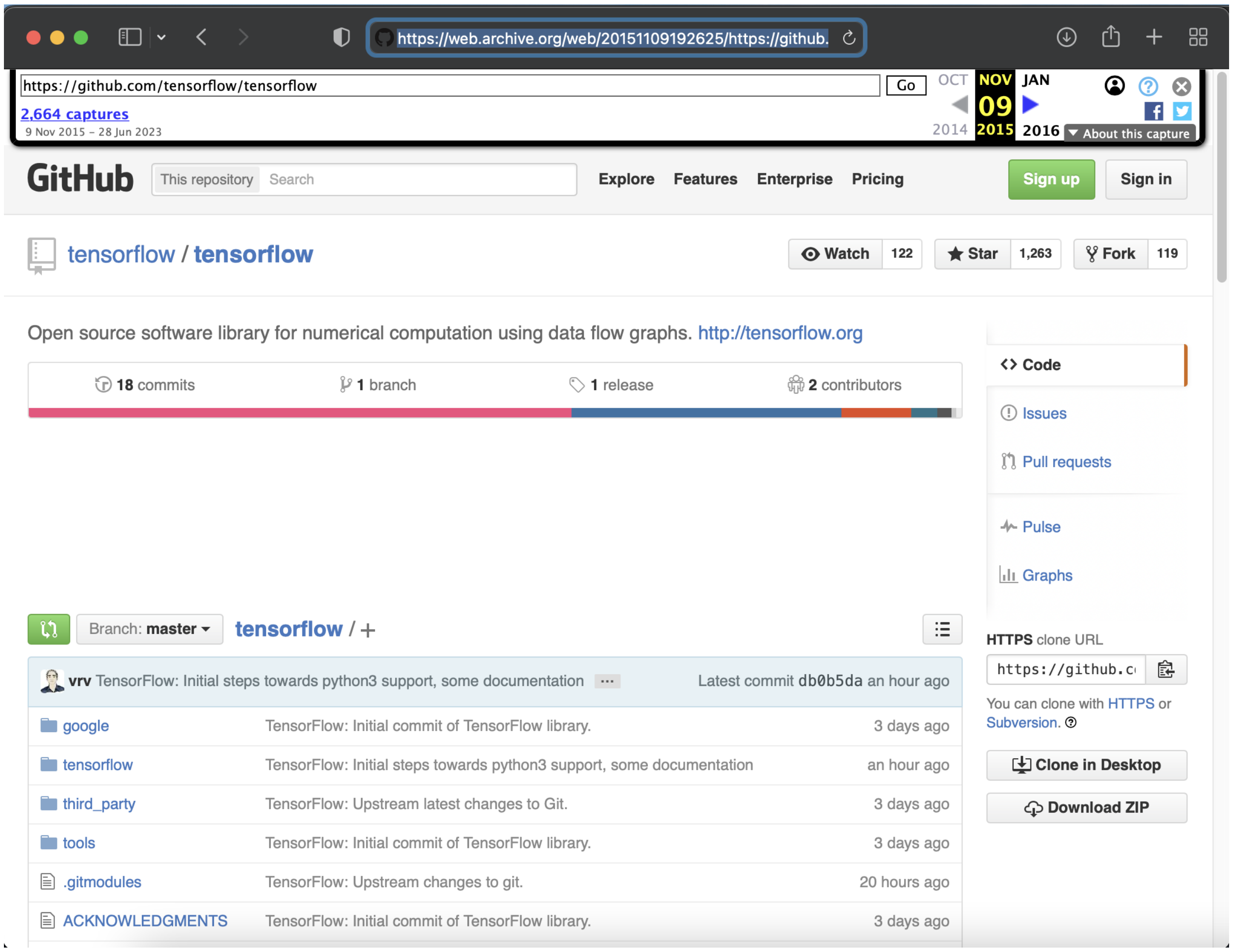
2.2 ARCHIVING SCHOLARLY CONTENT

The definition of scholarly content has expanded to include the data and source code that contribute to a publication. Of all scholarly content types, scholarly publications are the most well preserved due to organizations such as LOCKSS [7], CLOCKSS [8], and Portico [9] which are committed to the preservation of academic publications. However, they do not preserve the Web resources included in those academic publications. As a result, software products, datasets, and other important resources included via URI would not be archived alongside the publication.

Web archives, Software Heritage, and Zenodo are three of the primary archives that contain captures of code hosted in GHPs, but none of these archives provide a perfect solution for archiving source code and the surrounding ephemera.

2.2.1 Web Archives

The primary goal of Web archives [10, 11, 12] is the preservation of the Web at large with no special emphasis on the holdings of GHPs. Web archives crawl live Web pages, or **URI-Rs**, and create archived versions of the live Web pages called **mementos** or **URI-Ms** [13]. Each URI-M has an associated **Memento-Datetime**, the date and time that the URI-M was created. Each memento in Internet Archive is uniquely identified with a combination of the URI-R and the Memento-Datetime. Mementos also capture the HTTP response code of the URI-R. HTTP response codes reflect the status of the Web page [14]. If the Web page is available, the Web page will have a 2XX-level HTTP response code. However, if the page is not available, the Web page will most commonly have a 404 HTTP response code meaning that that Web page was not found. An example of a 404 HTTP response code is shown in Figure 3. Because Web archives, like the Internet Archive, capture more than just source code, mementos can be created for both the software product and the surrounding ephemera to allow users a complete picture of the hosted repository as it was available on the live Web. Referring back to Figure 4, Web archives would capture all of the features in both the orange and blue circles. Examples of mementos of GitHub in the Internet Archive are shown in Figures 10 and 11. Because Web archiving captures the Web page in addition to the source code, users are able to experience the Web page as it was available on the live Web (Figure 12) and see changes to the GitHub user interface (UI) over time.


https://web.archive.org/web/20151109192625/https://github.

https://github.com/tensorflow/tensorflow Go

2,664 captures
9 Nov 2015 – 28 Jun 2023

OCT NOV JAN
09
2014 2015 2016
About this capture

GitHub This repository Search Explore Features Enterprise Pricing Sign up Sign in

tensorflow / tensorflow

Watch 122 Star 1,263 Fork 119

Open source software library for numerical computation using data flow graphs. http://tensorflow.org

18 commits 1 branch 1 release 2 contributors

Code
Issues
Pull requests
Pulse
Graphs

Branch: master tensorflow / +

vrv TensorFlow: Initial steps towards python3 support, some documentation ... Latest commit db0b5da an hour ago

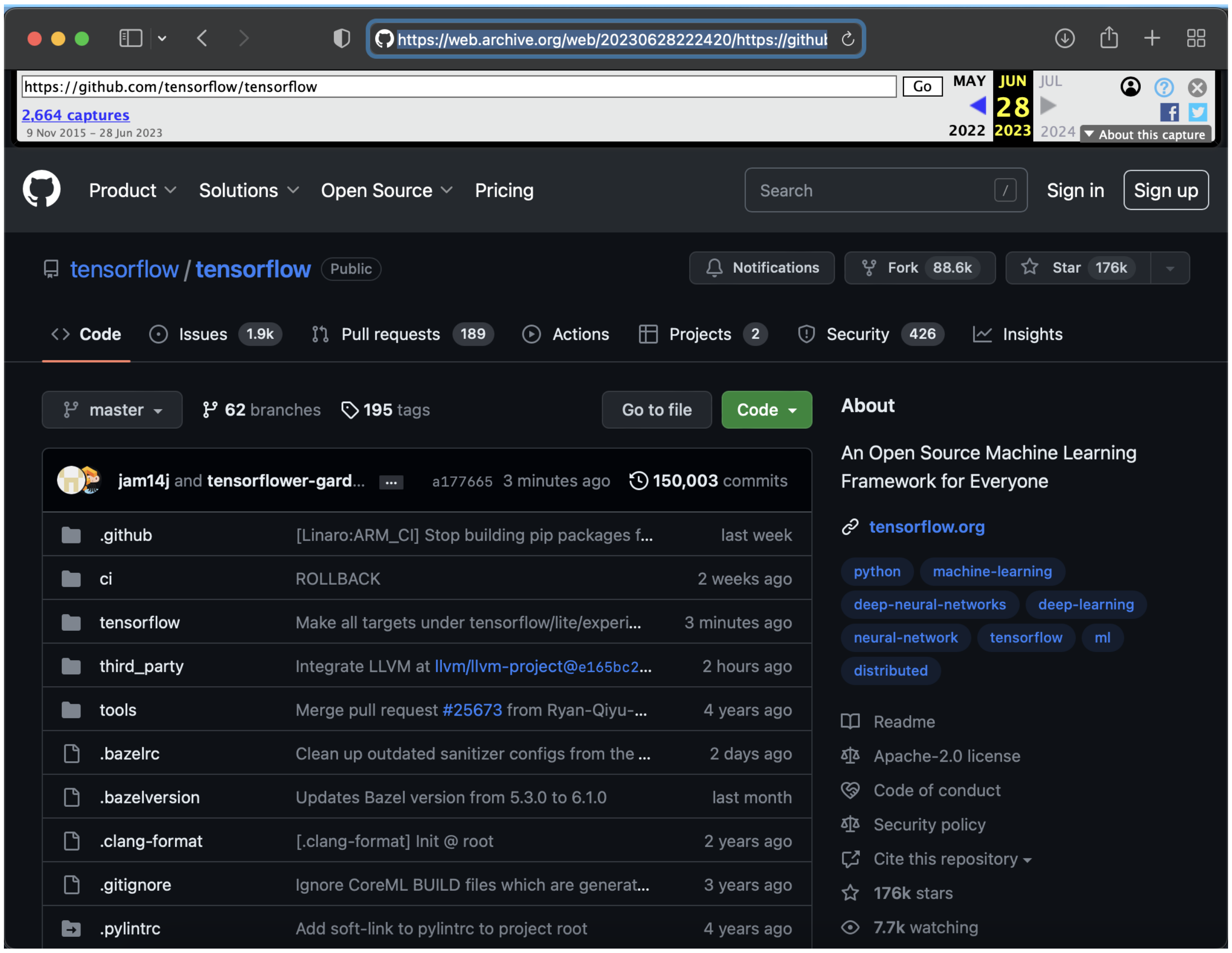
google	TensorFlow: Initial commit of TensorFlow library.	3 days ago
tensorflow	TensorFlow: Initial steps towards python3 support, some documentation	an hour ago
third_party	TensorFlow: Upstream latest changes to Git.	3 days ago
tools	TensorFlow: Initial commit of TensorFlow library.	3 days ago
.gitmodules	TensorFlow: Upstream changes to git.	20 hours ago
ACKNOWLEDGMENTS	TensorFlow: Initial commit of TensorFlow library.	3 days ago

HTTPS clone URL
https://github.c

You can clone with HTTPS or Subversion.

Clone in Desktop
Download ZIP

Fig. 10. A memento of `https://github.com/tensorflow/tensorflow` from November 9, 2015 (`https://web.archive.org/web/20151109192625/https://github.com/tensorflow/tensorflow`).



.github	[Linaro:ARM_CI] Stop building pip packages f...	last week
ci	ROLLBACK	2 weeks ago
tensorflow	Make all targets under tensorflow/lite/experi...	3 minutes ago
third_party	Integrate LLVM at llvm/llvm-project@e165bc2...	2 hours ago
tools	Merge pull request #25673 from Ryan-Qiyu-...	4 years ago
.bazelrc	Clean up outdated sanitizer configs from the ...	2 days ago
.bazelversion	Updates Bazel version from 5.3.0 to 6.1.0	last month
.clang-format	[.clang-format] Init @ root	2 years ago
.gitignore	Ignore CoreML BUILD files which are generat...	3 years ago
.pylintrc	Add soft-link to pylintrc to project root	4 years ago

About

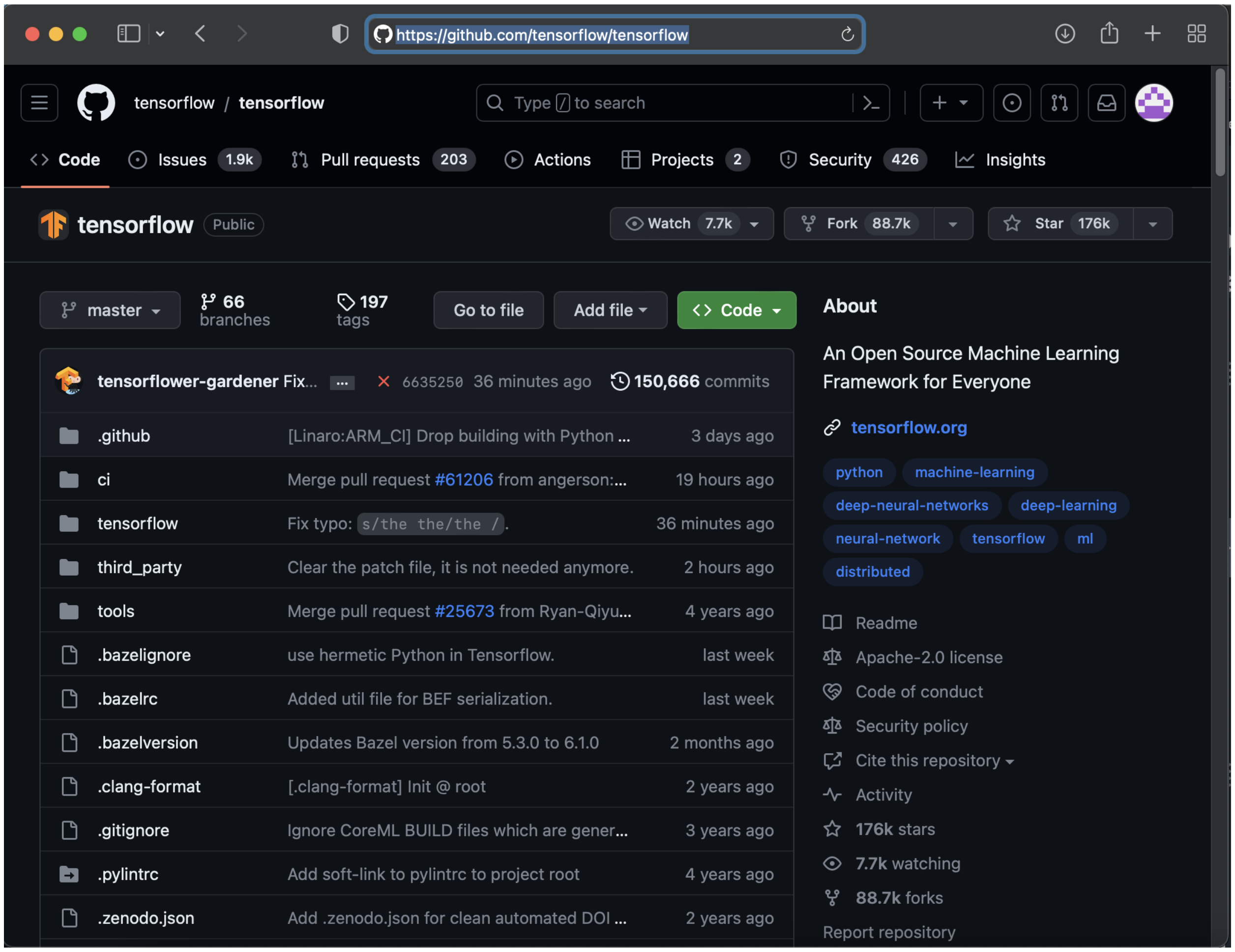
An Open Source Machine Learning Framework for Everyone

tensorflow.org

python machine-learning deep-neural-networks deep-learning neural-network tensorflow ml distributed

Readme
Apache-2.0 license
Code of conduct
Security policy
Cite this repository
176k stars
7.7k watching

Fig. 11. A memento of `https://github.com/tensorflow/tensorflow` from June 28, 2023 (`https://web.archive.org/web/20230628222420/https://github.com/tensorflow/tensorflow`).



tensorflower-gardener Fix... 6635250 36 minutes ago 150,666 commits

.github	[Linaro:ARM_CI] Drop building with Python ...	3 days ago
ci	Merge pull request #61206 from angerson:...	19 hours ago
tensorflow	Fix typo: `s/the the/the /`.	36 minutes ago
third_party	Clear the patch file, it is not needed anymore.	2 hours ago
tools	Merge pull request #25673 from Ryan-Qiyu...	4 years ago
.bazelignore	use hermetic Python in Tensorflow.	last week
.bazelrc	Added util file for BEF serialization.	last week
.bazelversion	Updates Bazel version from 5.3.0 to 6.1.0	2 months ago
.clang-format	[.clang-format] Init @ root	2 years ago
.gitignore	Ignore CoreML BUILD files which are gener...	3 years ago
.pylintrc	Add soft-link to pylintrc to project root	4 years ago
.zenodo.json	Add .zenodo.json for clean automated DOI ...	2 years ago

About

An Open Source Machine Learning Framework for Everyone

tensorflow.org

python machine-learning deep-neural-networks deep-learning neural-network tensorflow ml distributed

Readme
Apache-2.0 license
Code of conduct
Security policy
Cite this repository
Activity
176k stars
7.7k watching
88.7k forks
Report repository

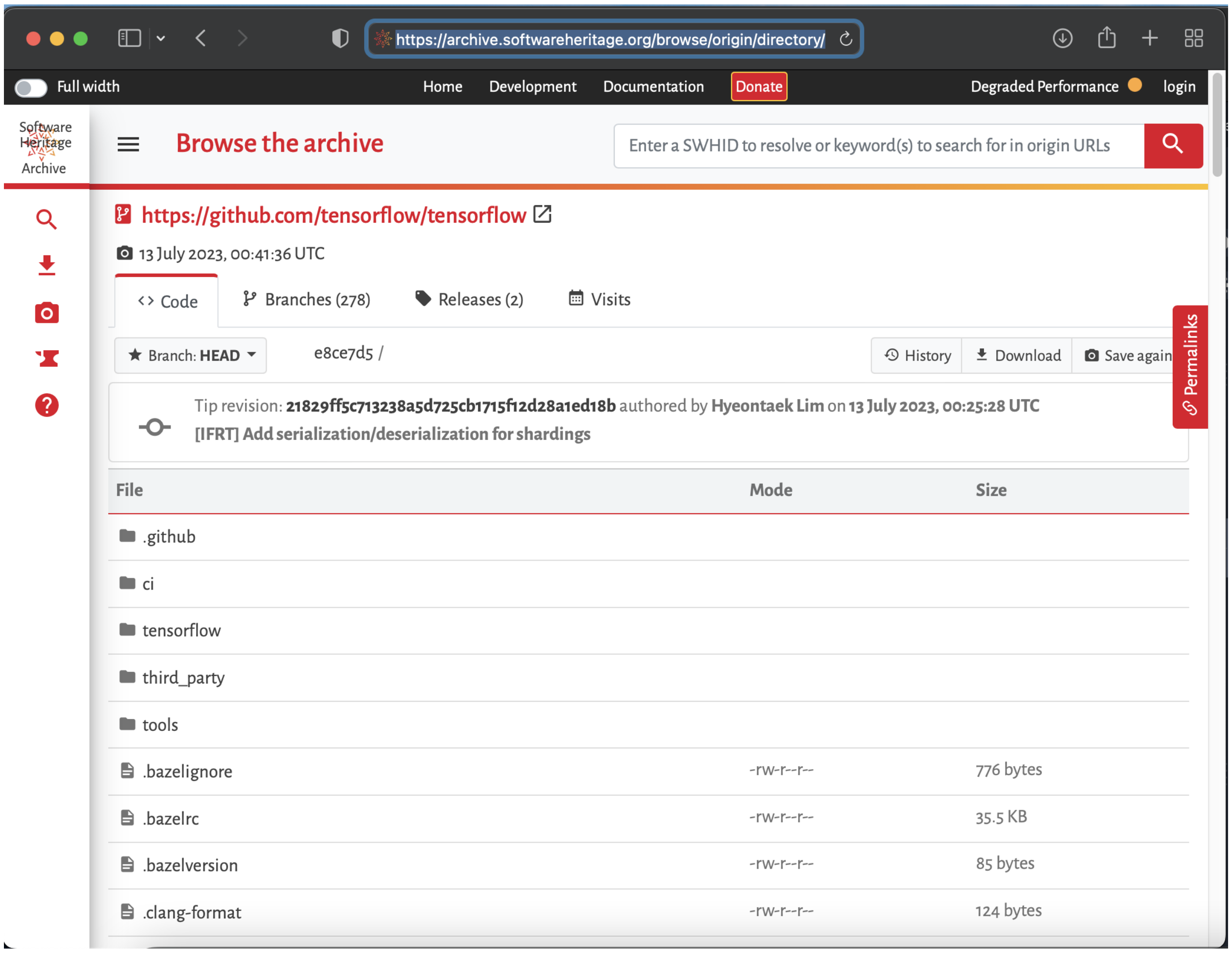
Fig. 12. The live Web page for `https://github.com/tensorflow/tensorflow`.

While the Internet Archive is a prominent Web archive, we chose to use MemGator to more broadly analyze the holdings of Web archives. MemGator [15] is a Memento aggregator that requests a given URI from each of 12 distinct Web archives: Archive.it, Archive.today, Australian Web Archive, BanQ, Bibliotheca Alexandria Web Archive, Icelandic Web Archive, Internet Archive, Library of Congress, Perma, Portuguese Web Archive, Stanford Web Archive, and UK Web Archive. After requesting the URI from each of the Web archives, MemGator compiles all of the archives' responses for the URI-R into a TimeMap that includes the URI-M of each memento and the corresponding Memento-Datetime. An example of a TimeMap returned by MemGator is shown in Appendix A.

2.2.2 Software Heritage

Software Heritage is a non-profit organization that works to “collect, preserve, and share all software that is publicly available in source code form” [16]. Software Heritage preserves source code and its development history from the perspective that source code is itself a valuable form of knowledge that should be captured, including the unique evolution of the source code to create the code product at a given point in time [17]. Repositories are harvested from various software sources and users are also able to submit repositories to “Save code now” to initiate a capture.

While Web archives typically have a large scope covering a wide variety of content types, Software Heritage is singularly focused on the archival of source code and its development history, so their captures solely archive the software product hosted in the GHP and do not archive the ephemera surrounding it. Referring back to Figure 4, Software Heritage captures would only reflect the functionality shown in the orange circle. As a result, Software Heritage provides a central repository containing the source code and development histories of millions of code products across programming languages, hosting platforms, and package repositories. This repository can be used by researchers to analyze source code as it is a more representative sample than a single hosting platform or package library. Software Heritage’s captures are accessible through both a UI and an API [18] as shown in Appendix B. As shown in Figure 13, Software Heritage only captures the repository and the commit history, so, unlike Web archives, users can see the source code but not the full GitHub UI. Software Heritage also differs from Web archiving in that their naming convention does not follow the terminology used in the Memento framework. In Software Heritage, the URI for a repository is an **origin** and each capture of an origin is a **visit** that creates **snapshots**. A persistent identifier is created for each artifact within the capture, called a SWHID. The API returns metadata for each visit including the origin (the original URI and the type of software origin), visit number, snapshot ID, date of the snapshot, and status of the snapshot [17] as shown in Appendix B.



https://archive.softwareheritage.org/browse/origin/directory/

Full width | Home | Development | Documentation | Donate | Degraded Performance | login

Software Heritage Archive

Browse the archive

Enter a SWHID to resolve or keyword(s) to search for in origin URLs

https://github.com/tensorflow/tensorflow

13 July 2023, 00:41:36 UTC

Code | Branches (278) | Releases (2) | Visits

Branch: HEAD | e8ce7d5 / | History | Download | Save again | Permalinks

Tip revision: 21829ff5c713238a5d725cb1715f12d28a1ed18b authored by Hyeontaek Lim on 13 July 2023, 00:25:28 UTC
[IFRT] Add serialization/deserialization for shardings

File	Mode	Size
.github		
ci		
tensorflow		
third_party		
tools		
.bazelignore	-rw-r--r--	776 bytes
.bazelrc	-rw-r--r--	35.5 KB
.bazelversion	-rw-r--r--	85 bytes
.clang-format	-rw-r--r--	124 bytes

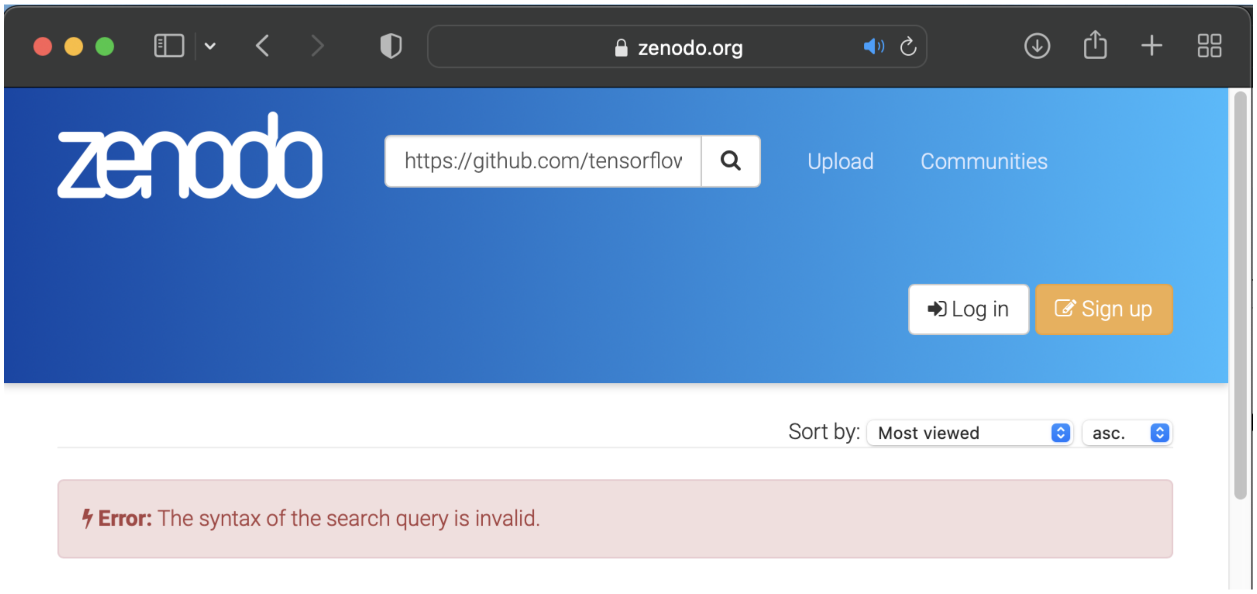
Fig. 13. A Software Heritage capture of `https://github.com/tensorflow/tensorflow` from Jul 13, 2023 (`https://archive.softwareheritage.org/swh:1:snp:64e2f22e9229000665c5be188f27795834417d69;origin=https://github.com/tensorflow/tensorflow`).

2.2.3 Zenodo

Zenodo,[6] a non-profit repository maintained by CERN that supports open data and open access to digital scholarly resources, is one example of a repository with specific functionality to support researchers who wish to self-archiving their code for long-term access. Zenodo provides a web-hook that allows users to deposit new releases from GitHub repositories. Zenodo makes a copy of

[6]`https://zenodo.org`

the code, rather than simply linking out to the GitHub page, creates relationships to previous and subsequent versions of the code, and mints a DOI for the record with software-specific metadata attached. However, we excluded Zenodo from our study because it does not support URI searches through its Web interface or API (Figure 14). Users can conduct text searches or find resources through direct links, but they cannot search for a URI which was prohibitive for our study.



Error: The syntax of the search query is invalid.

Fig. 14. Zenodo URI search. Zenodo does not support URI searches through the Web interface. Searching for `https://github.com/tensorflow/tensorflow` resulted in an error message.

2.2.4 Need for a Better Solution

Presently, neither self-archived code nor programmatically captured code incorporates the scholarly ephemera that can help secondary readers understand and evaluate the source code being cited. This is where Web archiving may be beneficial. Web archiving's goal lies in preserving the Web so that users can see a Web page as it existed at a certain point in time, which is helpful for archiving source code and the accompanying scholarly ephemera. However, because of the resources it takes to archive the Web, automated Web archiving services like the Internet Archive will crawl the most visited Web pages frequently, while the least visited Web pages, including scholarly content, may never be fully captured. Although the Internet Archive includes some GHP sites, it cannot be depended upon to preserve any given page in its entirety. Other Web archiving tools like the Webrecorder suite [19], provide higher quality captures of source code and ephemera, but take more time, resulting in decreased scalability for archiving the Web at large. Also, while current Web archiving implementations are well-suited for archiving the scholarly ephemera around scholarly code, they are less effective with the source code itself, which has different metadata and reuse needs than a typical Web page.

CHAPTER 3

RELATED WORK

In this chapter, we will discuss the impact of reference rot on URIs to the Web at large which motivates the need to understand the scope of scholarly code represented in scholarly literature. Next, we will discuss previous works that have studied links between scholarly literature and scholarly source code. We will then discuss previous studies on the holdings of Software Heritage as well as studies on the use of other archival methods and services. Lastly, we will discuss a machine learning classifier that can identify URIs to open-access data and software.

In 2014, Klein et al. [20] analyzed the use of URIs to the Web at large in 3.5 million scholarly articles published in arXiv, Elsevier, and PMC corpora from 1997 to 2012. They found that the number of general URIs used in scholarly publications rapidly increased from 1997 to 2012. However, they also found that reference rot affects nearly 20% of Science, Technology, and Medicine (STM) publications. When looking specifically at publications with at least one Web reference, seven out of ten publications are affected by reference rot. Reference rot is a general term that indicates that either link rot or content drift has altered the content of the Web page to be different than the content to which the author was originally referring [21]. Link rot occurs when the URI that was originally referenced is completely inaccessible. Link rot can cause the "404: Page not found" error that most Web users have experienced. Content drift occurs when the content that was originally referenced by a URI is different from the content currently available at the URI. Both link rot and content drift are a result of the dynamic and ephemeral nature of the Web. In a study on the same arXiv, Elsevier, and PMC corpora studied by Klein et al. [20], Jones et al. [22] found that 75% of references suffer from content drift. Additionally, they found that the occurrence and impact of content drift increases over time. In 2015, only 25% of referenced resources from 2012 publications were unchanged and, worse yet, only 10% of publications from 2006 were unchanged. Zittrain et al. [23] analyzed link rot and content drift in the New York Times and found that links were rotting at a consistent rate over time. In another study, Agata et al. [24] studied a dataset of 10 million URIs collected in 2001 and found that 90% of Web pages had disappeared between 2001 and 2013. A survey by Teixeira Da Silva and Nazarovets [25] found that reference rot is a well-documented problem that plagues URLs in all disciplines and categories, including academic literature. The authors advocate for a solution to reference rot in acadmeic literature saying "The preservation of web-based references that are cited in-text or in reference lists of academic papers is essential to ensure the integrity of knowledge preservation."

Because scholarly materials hosted on the Web are vulnerable to decay in the same manner as Web resources in general, we need to understand the extent to which scholarly articles reference source code. Understanding the scope of how scholarly source code is represented in scholarly literature is vital to strengthening efforts to preserve this code and make it available for the long-term, as a part of the scholarly record. This thesis is one of few studies that looks at the representation of links *to* scholarly source code in scholarly literature. Previous works have investigated the opposite: representation of links *to* scholarly literature *from* scholarly source code repositories. Wattanakriengkrai et al. [26] studied the extent to which scholarly papers are cited in public GitHub repositories to gain key insights into the landscape of scholarly source code production, and uncovered potential problems with long-term access, tracing, and evolution of these repositories. Färber [27] analyzed data from Microsoft Academic Graph, which maps publications to their source code repositories, in order to look at the content and popularity of academic source code related to published work. Other related work addresses finding scholarly source code repositories hosted in GHPs, either by looking through the content of the repository or by searching for links to scholarly literature in the repositories themselves. Hasselbring et al. [28] investigated the characteristics of public repositories on GitHub that either (a) referenced by a scholarly publication or (b) referenced a scholarly publication to understand the current state of. Bhattarai et al. [29] investigated the correlation between citations and repository interaction features and trained a classifier to predict whether a paper would be highly cited based on the interaction features of the repository included in the publication. The study looked at a range of interaction features and found that user engagement metrics, namely forks, stars, subscriptions, and issues, are the only attributes that had statistically significant correlation with citations across the analyzed timespan.

In Chapter 2, we introduced some of the efforts to archive scholarly products and, specifically, scholarly software products. As previously stated, Software Heritage's archival holdings also function as a central repository that allows researchers to study a variety of software products. Pietri et al. [30] and Bhattacharjee et al. [31] leverage the scope of the Software Heritage dataset to analyze trends in software development across a more heterogeneous dataset than could be found in a single hosting platform. While studies like these have made use of the holdings of Software Heritage, they do not analyze what has or has not been preserved in Software Heritage.

Instead of the relying on Software Heritage or the other archives introduced in Chapter 2, some scholars take an active role in the long-term preservation of their software and engage in a strategy known as self-archiving. Self-archiving puts the responsibility on scholars to deposit their code product into a repository that guarantees long-term preservation, like Zenodo [32] or the Open Science Framework [33]. However, a study by Milliken et al. [34] found that only 47.2% of the

Table 1. Repository Platforms

Name	Start Date	Protocol	URI
SourceForge	1999	git and SVN	`https://sourceforge.net`
Bitbucket	2008	git	`https://bitbucket.org`
GitHub	2008	git	`https://github.com`
GitLab	2014	git	`https://gitlab.com`

academics who create software products self-archive their software. While self-archiving can help safeguard research software, it has yet to become common practice for scholars with most scholars taking a passive role in the archiving of their source code. To understand the archival quality available through a variety of archival methods, another study by Milliken [34] conducted initial testing of GitHub, GitLab, SourceForge, and Bitbucket. Our project is a continuation of that study and, as a result, we chose to analyze the use of those four GHPs in the arXiv and PMC corpora. The GHPs are summarized in Table 1.

In addition to Zenodo or the Open Science Framework, there are a vast number of hosting platforms for open-access data and software (OADS). Salsabil et al. [35] created a machine learning classifier to classify URIs in scholarly publications as OADS or non-OADS. For their study, an OADS URI was simply defined as a URI linking to open access dataset and/or software. To be classified as OADS, the URI must be open access and a dataset or software product. The classifier transforms each article into a text file using the PDFMiner[1] Python library. By employing a regular expression, it scans the text to identify and extract sentences that contain URIs. Given the extracted sentences, the hybrid classifier combines two approaches: a heuristic classifier and a learning-based classifier. The heuristic classifier removes URIs that fall into two categories: those belonging to 54 major publishers such as Springer, Wiley, and Sagepub, and those that end with ".pdf". This is because publisher URIs are typically not associated with datasets or software repositories, and .pdf files are typically not datasets or software. The learning-based classifier was trained on a dataset of labeled sentences that contain URIs. The labeled samples were classified as either open access datasets/software (OADS), or not (non-OADS) as shown in Table 2. The OADS cases in the training set were verified to be both open access and data and/or software. The learning-based classifier used this information to learn how to classify new URIs. In the study by Salsabil et al., they found that the hybrid classifier is more accurate than either the heuristic classi-

[1] `https://pypi.org/project/pdfminer/`

Table 2. Sentences containing OADS and non-OADS URLs.

Sentences containing the URI	Category
The dataset is available at http://ibm.biz/multishapeinsertion.	OADS
Code and materials for reproducing the experiment as well as all data and analysis scripts are open and available at https://github.com/hawkrobe/pragmatics_of_perspective_taking.	OADS
The codebase that we adapted was developed by Laurent Haan (https://github.com/haan/Lightbot)	OADS
This article is available from: http://www.nature.com/articles/srep01037.	Non-OADS
All these scenes can be seen in our video at https://youtu.be/RcWHXL2vJPc.	Non-OADS
Their contributions are individually acknowledged at http://www.galaxyzoo.org/volunteers.	Non-OADS

fier or the learning-based classifier alone. This is because the heuristic classifier eliminates many non-relevant URIs, and the learning-based classifier is able to accurately classify the remaining URIs.

We know that: a) materials hosted on the Web and cited in scholarly literature are subject to reference rot, b) source code and its important scholarly ephemera are particularly at risk because of a lack of holistic archiving, and c) source code is being cited more in our scholarly literature. To understand the scope of source code citations and quantify the risk of loss, we analyzed a corpus of scholarly publications and the URIs to GHPs that the publications contain. This study will also investigate the prevalence of URIs to data and software products and identify the other Web-based repositories and hosting services that scholars are using and citing in scholarly work.

CHAPTER 4

MEASURING THE PREVALENCE OF GHP URIS IN SCHOLARLY PUBLICATIONS

In order to understand the scope of the threat of reference rot and missing resources to GHP URIs in scholarly publications, we first need to quantify the prevalence of GHP URIs in scholarly publications. In this chapter, we will investigate and answer RQ1 from Chapter 1: How often do authors include GHP URIs in scholarly articles? These results were first presented in our publication "The Rise of GitHub in Scholarly Publications" [36].

4.1 METHODOLOGY

We analyzed the arXiv and PubMed Central corpora as a representative sample of scholarly publications across Science, Technology, Engineering, and Math (STEM) disciplines, in order to understand how scholarly code is being referenced over time and, therefore, both woven into the fabric of our scholarly conversation and worthy of preservation. arXiv is one of the largest and most popular pre-print services, and the corpus contains over 2 million submissions [37] from eight disciplines: physics, mathematics, computer science, quantitative biology, quantitative finance, statistics, electrical engineering and systems science, and economics. The arXiv corpus does not allow for anonymous submissions, is publicly available, and is accessible for programmatic acquisition and analysis. The PubMed Central (PMC) corpus [38] contains publicly available full-text articles from a wide range of biomedical and life sciences journals. Only peer-reviewed journals are eligible for inclusion.[1] The size and availability of the arXiv and PMC corpora make them suitable for the purposes of our study.

In April 2007, the arXiv identifier scheme changed to accommodate a larger number of submissions and to address other categorization issues.[2] The previous scheme was in the form of `subject/YYMMnumber` as shown in `https://arxiv.org/pdf/cs/0511077.pdf` while the updated scheme excludes the class in the identifier and uses the `YYMM.number` form as shown in `https://arxiv.org/pdf/2208.04895.pdf`. We decided that beginning our arXiv corpus in April 2007 would suit our analysis, because three of the four repository platforms that we analyzed began after 2007. Each pre-print in arXiv can have multiple versions. When an author uploads a new version of the pre-print to the service, the version number increments by one. All versions of a pre-print are accessible in arXiv via a version-specific URI. For our analysis, we

[1] `https://www.ncbi.nlm.nih.gov/pmc/pub/addjournal/`

[2] `https://arxiv.org/help/arxiv_identifier`

Table 3. Five journals with the most articles in the PMC corpus. Reproduced from ref. [36] with permission from Springer Nature

Journal	Articles	Earliest	Latest
The Indian Medical Gazelle	29,143	1866	1955
The Journal of Cell Biology	24,349	1962	2022
The Journal of Experimental Medicine	24,207	1896	2022
BMJ Open	21,565	2011	2022
Edinburg Medical Journal	20,160	1855	1954

considered only the latest version of each submission, assuming that the final submission was the most complete and most representative of the author's intentions. With only the latest version of each submission, our arXiv corpus contained 1.56 million publications in PDF format from April 2007 to December 2021.

The PMC corpus includes articles from the late 1700s to present from peer-reviewed journals. The most prevalent journals in the corpus are listed in Table 3 along with the number of articles in the corpus, the date of the first article available, and the date of the latest article. In order to more easily compare the corpora and because, as previously noted, three of the four repository platforms we analyzed began after 2007, we decided that beginning our PMC corpus in January 2007 was appropriate for our analysis. Additionally, the PMC corpus separates articles that are available for commercial use from those that are only available for non-commercial use. We chose to analyze the articles that were only available for non-commercial use. Our PMC corpus contained 1.08 million publications in PDF format from 2007 to 2021. Between the arXiv and PMC corpora, we analyzed 2,641,041 publications.

URIs are not exclusively found in the References section of a publication; they also commonly appear in footnotes and the body of the text. To extract all of the URIs in each publication, regardless of location, we leveraged two Python libraries: PyPDF2[3] and PyPDFium2.[4] We used PyPDF2 to extract annotated URIs and PyPDFium2 to extract URIs from the PDF text. We followed a similar URI characterization method as that done by Klein et al. [20] who identified URIs to "Web at large" resources in-scope for their study. Since we are investigating links to GHPs, our primary goal with extraction was to identify URIs to one of the four GHPs. However, we also identified

[3] `https://pypi.org/project/PyPDF2/`

[4] `https://pypi.org/project/pypdfium2/`

URIs to the Web at large to provide context for the frequency and use of URIs to the GHPs. To do this, we filtered out a number of URIs that were out of scope for this study. We dismissed URIs with a scheme other than HTTP or HTTPS, including localhost and private/protected IP ranges. We also dismissed URIs to arXiv, Elsevier RefHub,[5] CrossRef Crossmark [39], and HTTP DOIs and, as such, follow the definition of URIs to "Web at large" resources that are in-scope for our work. Examples of the URIs that were excluded under the above conditions are shown in Table 4. DOIs resolve to artifacts, most commonly papers but increasingly also to data (e.g., via Dryad) and source code (e.g., via Zenodo). Links to Elsevier RefHub and CrossRef Crossmark function similarly to DOIs and are often added by the publisher. We decided to exclude DOI and DOI-like references following Klein et al.'s assumption that, for the most part, such artifacts are in-scope for existing archiving and preservation efforts such as LOCKSS, CLOCKSS, and Portico. Our source code is available on GitHub [40].

Table 4. Examples of URIs that were considered to be out of scope for our study. Reproduced from ref. [36] with permission from Springer Nature

URI	Category
http://localhost:8000/transfers	localhost
http://192.245.169.66:8000/FCCeeMC/wiki/kkmc	private IP
http://arxiv.org/abs/1311.4158	arXiv
http://refhub.elsevier.com/S1384-1076(15)00089-5/sbref0009	Elsevier RefHub
https://crossmark.crossref.org/dialog/?doi=10.1098/rsif.2013.0568 &domain=pdf&date_stamp=2013-09-04	CrossRef Crossmark
http://creativecommons.org/licenses/by-nc-nd/4.0/	Creative Commons
http://doi.acm.org/10.1145/1040305.1040306	HTTP DOIs

After extracting URIs from the PDFs in our corpora, we found 7,746,682 in-scope URIs: 4,039,772 URIs from the arXiv corpus and 3,706,910 URIs from the PMC corpus. Out of 2.64 million PDF files, 1,439,177 files (54.06%) contained a URI. Once we had collected all of the URIs from the PDFs, we used the regular expressions shown in Appendices D to G to filter and categorize

[5] `https://refhub.elsevier.com`

Table 5. Number of references to each GHP in the arXiv and PMC corpora. Reproduced from ref. [36] with permission from Springer Nature

	arXiv		PMC	
Repository Platform	Number	Percentage	Number	Percentage
GitHub	215,621	93.26	18,471	82.52
SourceForge	9,412	4.07	3,309	14.78
Bitbucket	3,525	1.52	437	1.95
GitLab	2,648	1.15	167	0.75

the URIs that referenced one of the four GHPs. For each GHP, we identified the base URIs that belong to the sitemap and, as such, are not repository URIs. The regular expression for SourceForge URIs shown in Appendix D is fairly straightforward. If the URI contained `sourceforge.net`, and was not in the sitemap, it was considered a repository URI. For both GitLab (Appendix E) and Bitbucket (Appendix F), we identified and removed links to Bitbucket pages and `GitLab.io`. Both of these types of pages are used to host Web pages, not the repositories that we were identifying. GitHub repository URIs (Appendix G) were the most complicated to identify. We excluded URIs to `github.com/gist`, a code snippet sharing service, and `GitHub.io` as these URIs were not repository URIs. We also excluded URIs to the Internet Archive. While these Internet Archive URIs contained URIs to GitHub, they were not themselves GitHub URIs. Because we used regular expressions to capture known GHP URI patterns, URIs to repository pages with custom domain names [41] were not captured. We found a total of 253,590 URIs to one of the four GHPs: 231,206 URIs from the arXiv corpus and 22,384 URIs from the PMC corpus. All GHP URIs in a publication have been deemed by the authors to be important enough for inclusion in the publication. As a result, we do not differentiate links to GHPs regardless of link depth or location in the publication. Inclusion of a GHP URI does not indicate an authorship or ownership claim. GHP URIs in a publication indicate that a resource either (1) impacted the work presented in the publications or (2) was a product of the study. Both cases communicate the importance of the repository and need for preservation. The number and percentage of URIs for each GHP are shown in Table 5.

4.2 RESULTS

By extracting URIs for the four repository platforms, we made a number of interesting observations. As shown in Figure 15, we found a continuation of the significant increase in the prevalence of URIs in publications that Klein et al. [20] found in 2014. Figure 15 shows the average number of in-scope URIs and the average number of URIs to one of the four GHPs in each publication by month of submission for both the arXiv and PMC corpora. The URIs to one of the four GHPs are a subset of in-scope URIs extracted from the publications. From 2007 to 2021, the average number of URIs per publications has steadily risen. In 2007, publications contained an average of 1.02 URIs. In 2021, publications contained an average of 5.06 URIs. The average number of in-scope URIs in each publication is indicated by the red and orange lines in Figure 15.

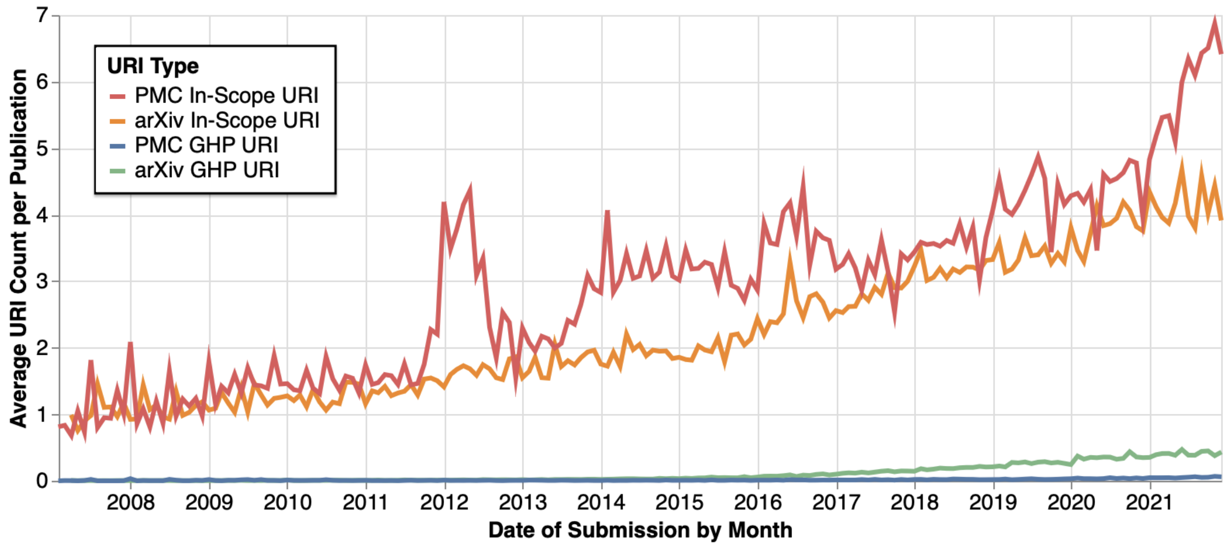


Fig. 15. The average number of in-scope URIs and URIs to repository platforms per publication over time. Reproduced from ref. [36] with permission from Springer Nature.

While the prevalence of URIs in general has increased, the number of URIs to repository platforms has also grown from 2007 to 2021. Just as there was a shift from not including Web resources in scholarly publications to including Web resources, there has also been a shift to referencing repository platforms in scholarly publications. Figures 16 and 17 show that references to GitHub have steadily risen from 2014 to 2021 while the frequency of references to the other three platforms have remained low during that time period. In the arXiv corpus shown in Figure 16, less than 1% of publications contain a URI to GitLab, Bitbucket, or SourceForge in any given year from 2007 to 2021. However, an average of 20% of publications contained a URI to GitHub in 2021. The PMC corpus in Figure 17 shows the initial dominance of SourceForge as the most popular GHP beginning in 2007. Beginning in 2015, GitHub replaced SourceForge as the most popular GHP. Both Figures 16 and 17 show a steady increase in the use of GitHub URIs in scholarly publications. Like URIs to the Web at large, URIs to repositories contribute to the context and argument of the publication. As the prevalence of GitHub URIs in publications increases, so does the importance of archiving source code repositories with their scholarly ephemera.

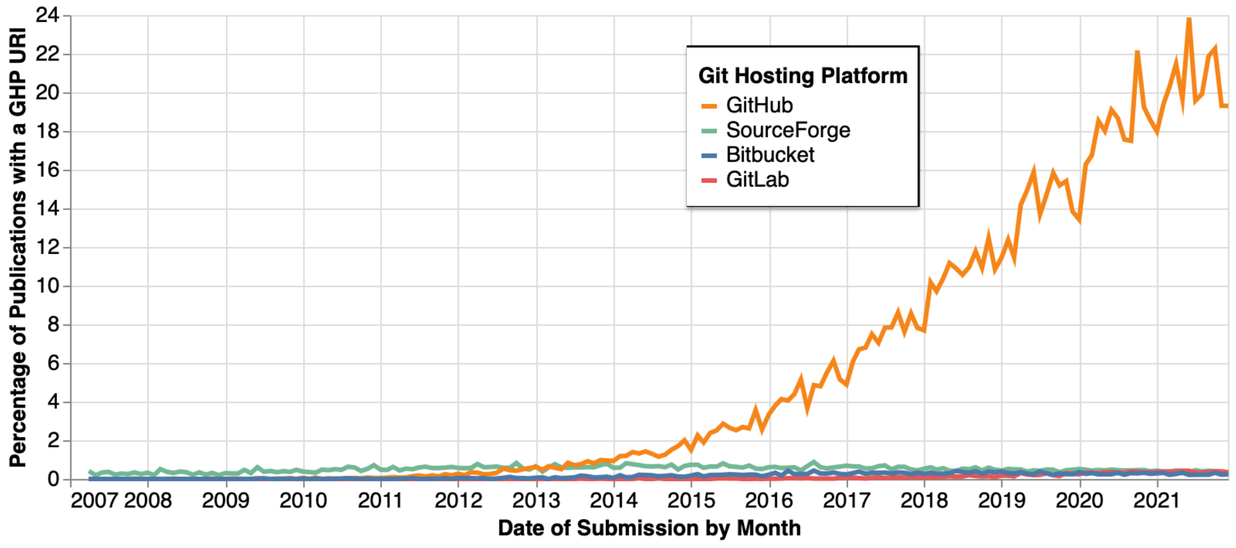


Fig. 16. The percentage of arXiv publications with a URI to a repository platform over time. Reproduced from ref. [36] with permission from Springer Nature.

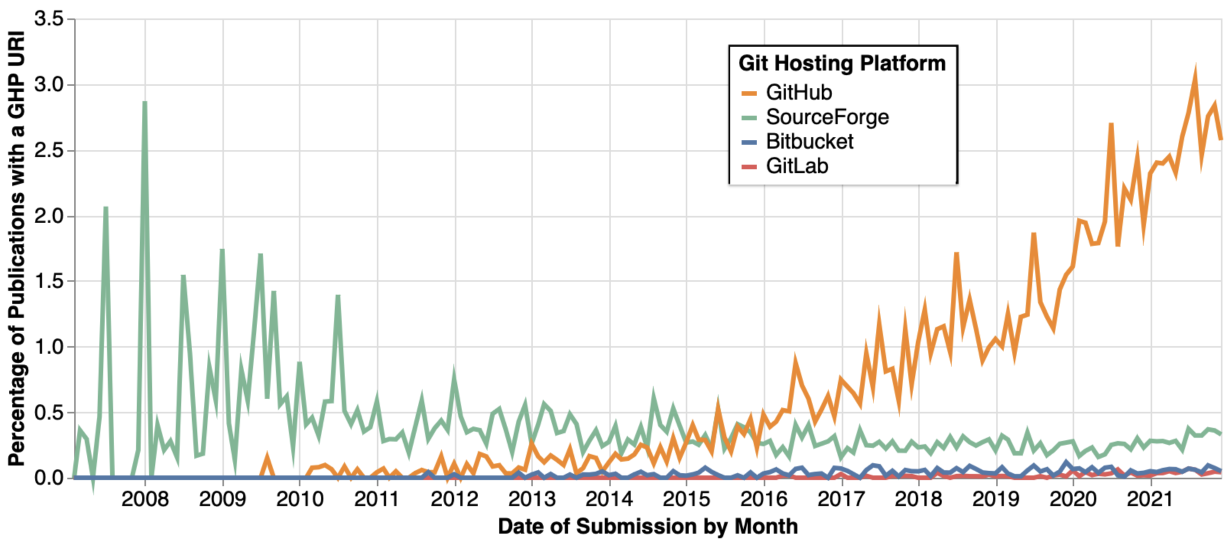


Fig. 17. The percentage of PMC publications with a URI to a repository platform over time. Reproduced from ref. [36] with permission from Springer Nature.

Additionally, while 67% of publications only reference a given repository once, 45,780 publications reference a given platform's holding more than once. Figure 18 shows the frequency of GHP URIs in publications that contain one or more GHP URI. For example, as shown in Figure 18A, of the 125,711 publications in the arXiv corpus that reference GitHub, 83,328 publications (66.3%) reference GitHub once, 42,383 publications (33.7%) reference GitHub more than once, and 863 publications (0.687%) reference GitHub more than ten times. We manually inspected a sample of the publications with the most URIs to one of the four GHPs and found these publications tend to detail a software product or provide an overview of a topic, such as survey paper. The top ten publications containing the most URIs to a GHP, which happen to all be arXiv publications, are shown in Table 6. The top three publications contain 153 [42], 160 [43], and 896 [44] URIs to GitHub. Dhole et al. [42] developed a software product and included URIs to the implementation of the features listed in the publication. Agol et al. [43] created an open-source package and linked to the implementation of the algorithms and processes described in the publication. Truyen et al. [44] wrote a survey paper comparing frameworks. A majority of the frameworks surveyed are documented in GitHub, so the survey contains numerous URIs to the documentation. The publication by Truyen et al. with 896 URIs to GitHub is not included in Figure 18, because it represents such a large outlier compared to the other publications in the corpus.

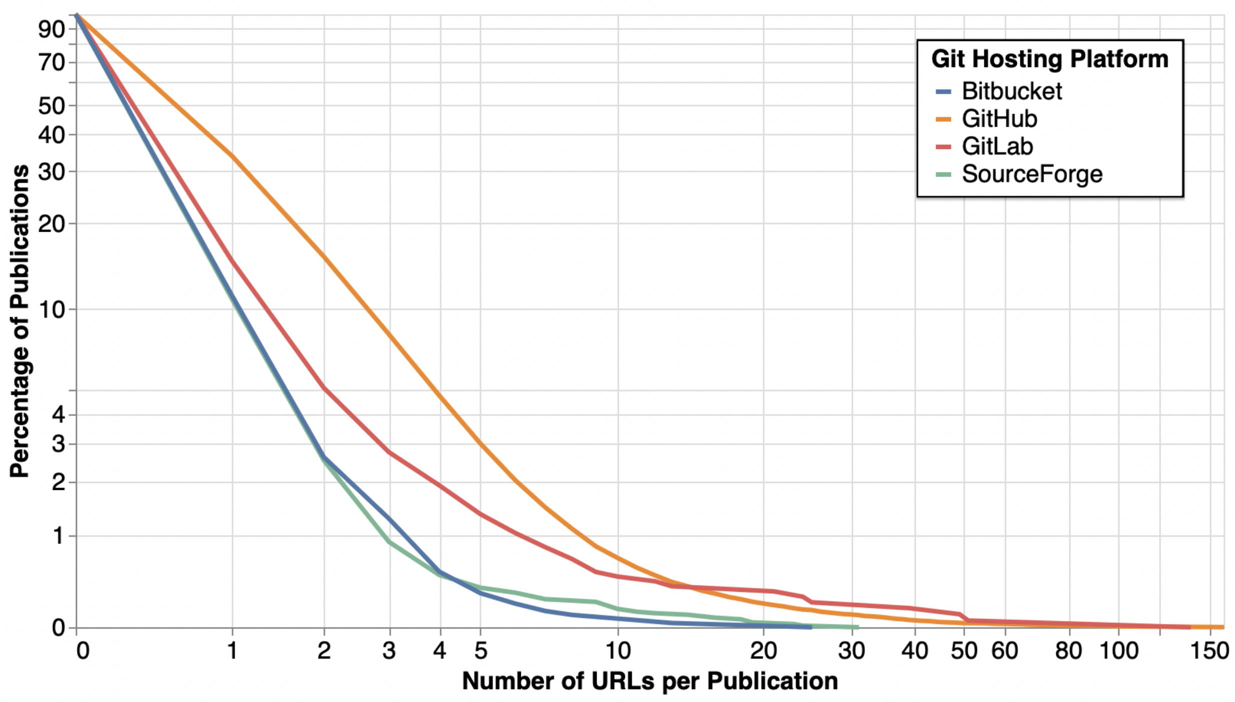
As shown in Figure 18B, of the 11,386 publications in the PMC corpus that reference GitHub,

Table 6. Top 10 publications with the most URIs to a single GHP. The publication identifier, number of URIs to the listed GHP, GHP, and title of the publication for the top 10 publications with the most URIs to a single GHP. Reproduced from ref. [36] with permission from Springer Nature

URI Count	arXiv ID	GHP	Title
893	2002.02806	GitHub	A Comprehensive Feature Comparison Study of Open-Source Container Orchestration Frameworks
160	2106.02188	GitHub	A differentiable N-body code for transit timing and dynamical modeling. I. Algorithm and derivatives
153	2112.02721	GitHub	NL-Augmenter: A Framework for Task-Sensitive Natural Language Augmentation
122	1708.04058	GitHub	Science-Driven Optimization of the LSST Observing Strategy
116	2102.07636	GitHub	Formalized Haar Measure
107	2109.11677	GitHub	Security Review of Ethereum Beacon Clients
92	1908.07883	GitHub	Scala Implicits are Everywhere: A large-scale study of the use of Implicits in the wild
90	2101.10632	GitHub	A Survey on Data Plane Programming with P4: Fundamentals, Advances, and Applied Research
88	2112.15439	GitHub	Facial-Sketch Synthesis: A New Challenge
84	1812.04202	GitHub	Deep Learning on Graphs: A Survey

7,983 publications (70.1%) reference GitHub once, but 3,403 publications (29.9%) reference GitHub more than once and 60 publications (0.527%) reference GitHub more than ten times. The top four publications with the most URIs to a GHPs contain 39 [45, 46], 40 [47], and 45 [48] URIs to GitHub. Like the arXiv corpus, each of these four publications provides a survey of the computation tools available in a given discipline.

A



B

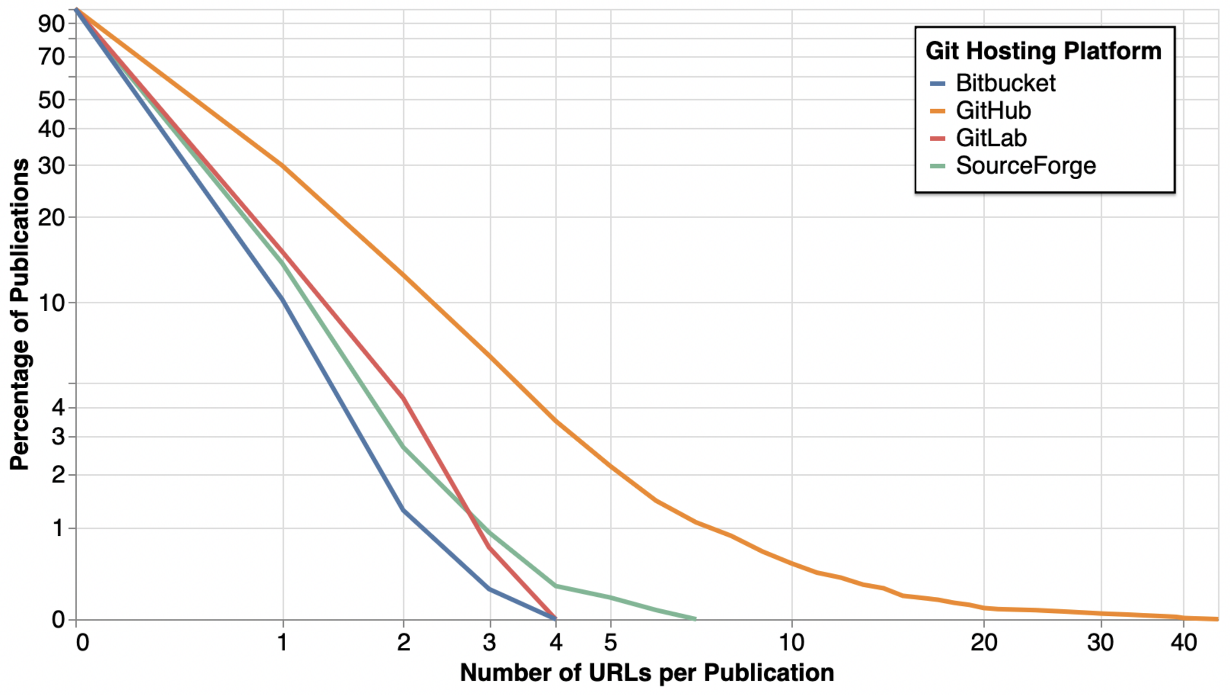


Fig. 18. If a publication links to GHP, how many links does it have? This figure is a Complementary Cumulative Distribution Function (CCDF) graphing the frequency of GHP URIs in publications with 1 or more GHP URI. Reproduced from ref. [36] with permission from Springer Nature.

Publications with multiple references to GitHub imply that the repositories have significant value and relevance for the authors, indicating that they deemed the repository contents important to the content of the publication. As a result, these repositories should be preserved in archives to guarantee that future readers can access the publication's full context.

We also analyzed the use of URIs to GHPs by discipline for the arXiv corpus. When submitting an article to arXiv, authors are prompted to select the primary discipline of the article. We used the metadata associated with each article to map each discipline to the four GHPs based on the number of URIs to each GHP. Figure 19 shows a visualization of the relationship between GHPs and STEM disciplines. The top half of the figure shows the GHPs and the bottom half of the figure shows the discipline of the publication. The ribbons that connect a GHP and a discipline represent the portion of URIs to a GHP within a discipline. For example, Physics and Computer Science contain the highest number of GHP URIs and most of those GHP URIs are to GitHub. Considering the prevalence of software products and models in the Computer Science and Physics disciplines as well as the popularity of GitHub, these results are not surprising.

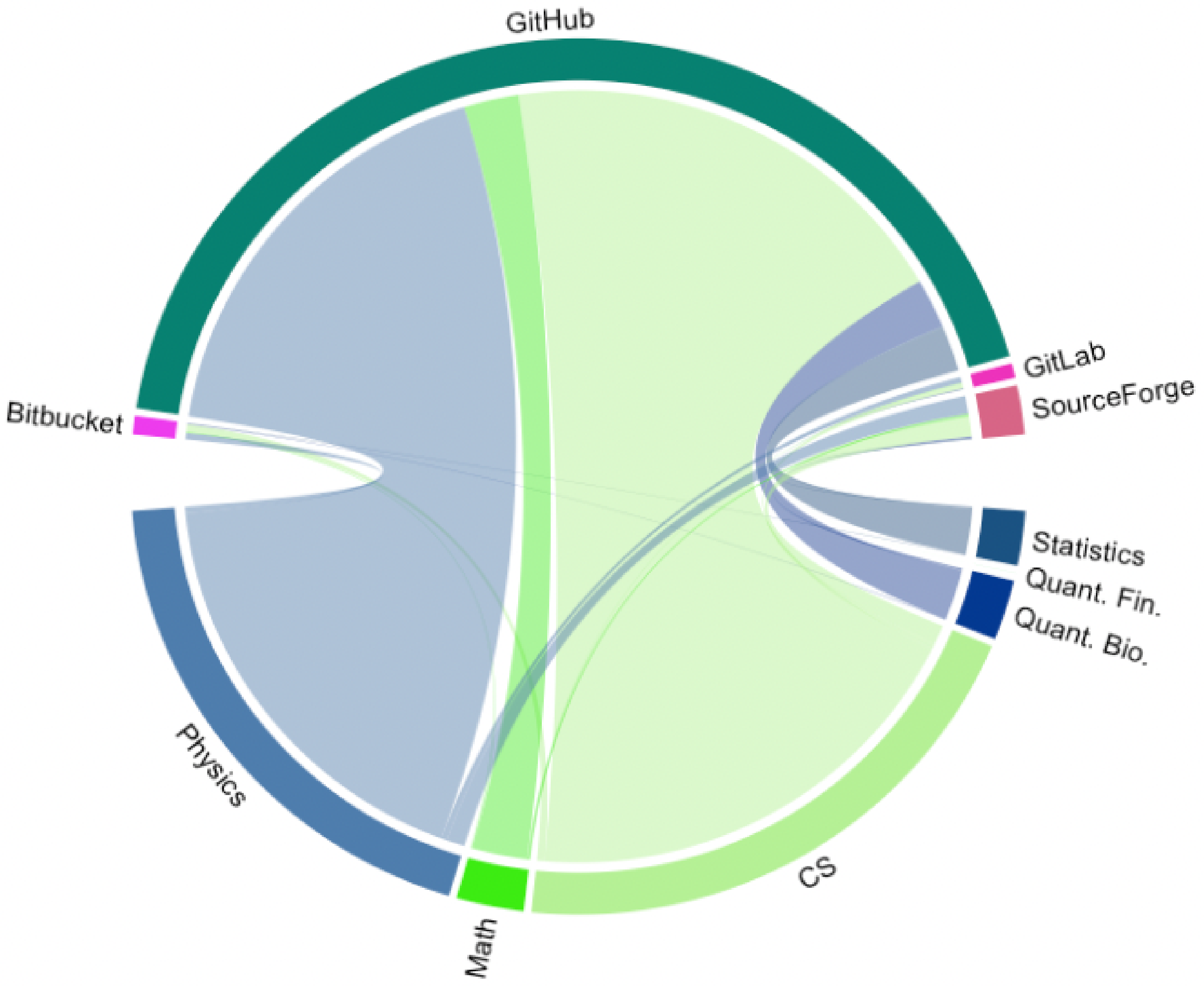


Fig. 19. Mapping the number of links to a GHP by discipline for the arXiv corpus. The GHPs are shown on the top half of the diagram and the discplines are shown on the bottom half of the diagram. Reproduced from ref. [36] with permission from Springer Nature.

4.3 DISCUSSION

We analyzed the holdings of the arXiv and PMC corpora, but other corpora that service a wider variety of disciplines could provide additional perspectives. Additionally, authors must submit their paper to the arXiv corpus. This could create another source of bias in that authors must be able to navigate the submission process and must choose to submit their publication. Authors who intentionally submit their paper to arXiv are proving that they value open source and resource sharing, so this may be one reason that links to GHPs are more prevalent in the arXiv corpus. The PMC corpus is an example of a corpus that does not require action by the authors. Journals apply to be included in the PMC archive and all articles from the journal are automatically included. In future work, we will look at aggregating additional corpora to obtain a more representative sample of disciplines.

4.4 SUMMARY

In this chapter, we addressed the first research question: How often do authors include GHP URIs in scholarly articles? We found that the average number of URIs in scholarly articles has steadily risen along with the number of GHP URIs. In 2021, publications contained an average of 5.06 URIs and 20% of arXiv publications contained at least one GitHub URI. Scholars across a wide range of disciplines are placing increasing importance on the holdings of GHPs, especially GitHub, in their publications.

CHAPTER 5

QUANTIFYING THE ARCHIVAL RATE OF SCHOLARLY GHP URIS

In Chapter 4, we established the increasing prevalence of GHP URIs in scholarly publications. Next, we studied whether the resources at the URIs were still available on the live Web or in an archive. In this chapter, we will address RQ2: For the GHP URIs identified in scholarly articles, what is the prevalence of these GHP URIs (a) on the live Web, (b) in Software Heritage, and (c) in Web archives?

5.1 METHODOLOGY

We used the dataset of GHP URIs created in Chapter 4. In total, the dataset contained 253,590 GHPs URIs that were referenced in scholarly publications. The distribution of the URIs in each GHP is shown in Table 7. For each URI in the dataset, we conducted three tests: (1) is it available on the live Web?, (2) is it available in Software Heritage?, (3) is it available in Web archives? We also analyzed the relationship between the publication date of the earliest article to reference a URI and the date of the first Software Heritage and Web archive capture of the URI.

In this study, we adopt the terminology used by Klein et al. [20]. As introduced in Chapter 2, a URI is *publicly available* if a curl request results in a 2XX-level HTTP response code. If a URI is publicly available, we consider the URI to be **active** on the live Web. Private repositories respond to a curl request with a 404 HTTP response code. While a private repository exists and is available to the owner, it is not publicly available and accessible via the URI provided in the scholarly publication; therefore, because a URI to a private repository is not active to general users, it is not considered an active URI. Any URI that does not result in a 2XX-level HTTP response code is

Table 7. Number of URIs to each GHP and the percentage of all GHP URIs.

GHP	Number of URIs	Percent of GHP URIs
GitHub	234,092	92.31%
SourceForge	12,721	5.01%
Bitbucket	3,962	1.56%
GitLab	2,815	1.11%

considered inactive, meaning that the URI is **rotten**, or is subject to link rot. In our curl requests, we opted to follow redirects and considered the resulting HTTP response code as the final status of the URI.

We utilized the Software Heritage API introduced in Chapter 2 to determine if Software Heritage contained a snapshot of the URI. However, Software Heritage only supports searching for URIs at the repository level, whether through their browser search interface or API request. Searching for deep links to a specific file or directory will not result in a match, even if the file or directory is available within Software Heritage's snapshot of the repository. For example, `https://github.com/aliasrobotics/RVD/blob/master/rvd_tools/database/schema.py` is a URI that was extracted from an article in the arXiv corpus. As it is written, the Software Heritage API was not able to find a matching origin. The HTTP request and response are shown in Appendix C. When we truncate the URI to the repository-level (`https://github.com/aliasrobotics/RVD`), the Software Heritage API returned a matching origin URL. To accommodate the requirements of the Software Heritage API, we transformed all deep URIs to shallow, repository-level URIs and requested the resulting URI from the Software Heritage API using a personal authorization token to increase the rate limit. Software Heritage uses repository URIs to capture hosted software products; however, and not all SourceForge projects support access to the code repository via a repository URI. Therefore, we excluded SourceForge URIs from this portion of the analysis. We used the Software Heritage API for each of the GitHub, GitLab, and Bitbucket URIs we extracted in Chapter 4. From the API response, we extracted the date of the first and last snapshot and the total number of snapshots for each URI.

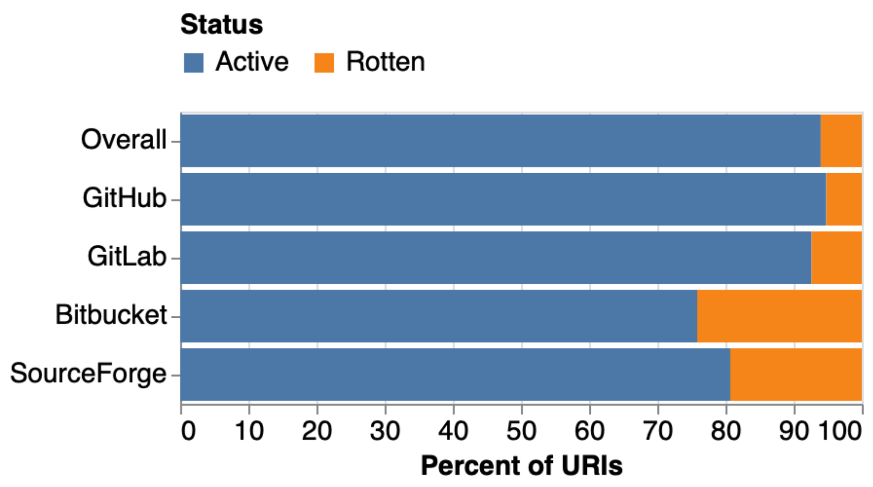
To determine if the GHP URI was archived by the Web archives, we used MemGator introduced in Chapter 2 to search for each GHP URI. A TimeMap, a list of all URI-Ms for the URI-R, is returned by MemGator. We extracted the Memento-Datetime for the first and last memento and the total number of mementos for each URI-R from the associated TimeMap.

McCown and Nelson [49] developed a framework for discussing the intersection of Web archiving and the life span of a Web resource, which we have adapted to discuss the intersection of Web archiving and the life span of source code in a GHP. We define a GHP URI resource as **vulnerable** if it is publicly available on the live Web but has not been archived. If a GHP URI resource is publicly available on the live Web and has been archived, we define the GHP URI resource as **replicated**. Lastly, we define a GHP URI resource as **unrecoverable** if it is no longer publicly available on the live Web and has not been archived.

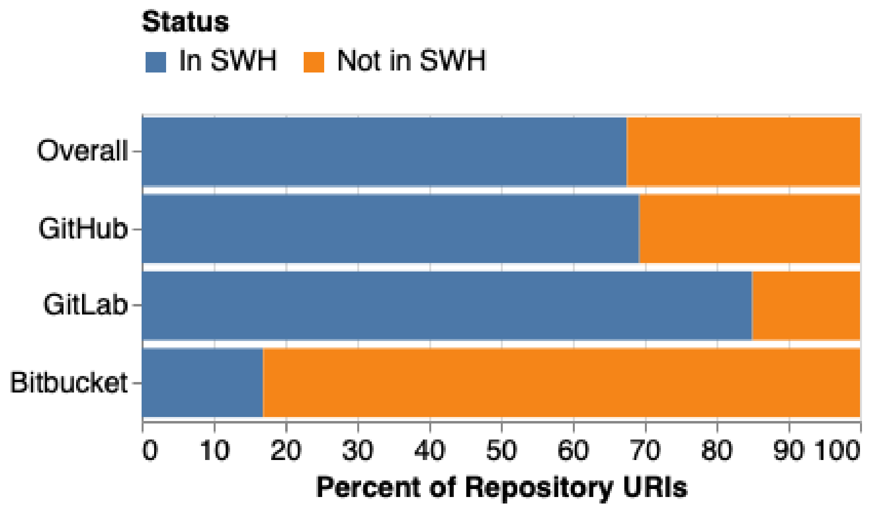
5.2 RESULTS

Across all four GHPs, 93.98% of all GHP URIs referenced in scholarly publications were active, as shown in Figure 20A. However, 6.02%, or 8,882 URIs of the unique GHP URIs in scholarly publications, were rotten. GitHub had the highest percentage of active URIs with 94.79%. Bitbucket had the lowest percentage of active URIs with 75.86% resulting in 641 rotten URIs.

A



B



C

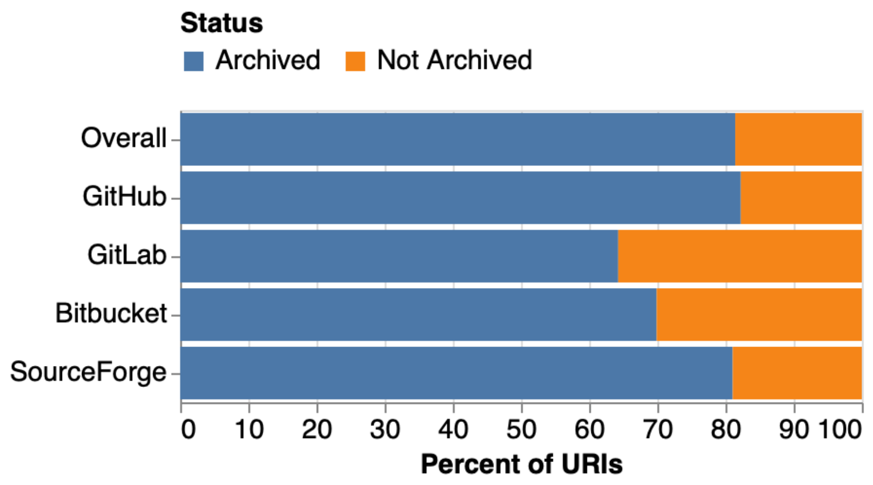


Fig. 20. Results of running the three tests: (1) is the URI active?, (2) has the URI been archived by Software Heritage?, and (3) has the URI been archived by Web archives? *(A)* Percent of active repository URIs. *(B)* Percent of repository URIs captured by Software Heritage (SWH). *(C)* Percent of repository URIs with at least one memento.

Table 8. Percent of all mementos returned from each of the 12 Web archives.

Web Archive	Percent of Mementos
Internet Archive	58.68
Bibliotheca Alexandria Web Archive	23.29
Archive.today	8.06
Archive.it	3.07
Portuguese Web Archive	2.83
Library of Congress	2.53
Icelandic Web Archive	0.88
Australian Web Archive	0.35
UK Web Archive	0.12
Perma	0.11
Stanford Web Archive	0.08
BAnQ	0.0005 (1 URI-M)

As shown in Figure 20B, 67.52% of all repository-level GHP URIs have at least one snapshot in Software Heritage. GitLab had the highest percentage of repository URIs captured by Software Heritage with 85.03%. Bitbucket has a relatively small percentage of repository URIs available in Software Heritage 16.93%.

Across all four GHPs, 81.43% of GHP URIs have at least one memento in the Web archives queried by MemGator, as shown in Figure 20C. GitHub had the highest percentage of URIs available in Web archives with 82.25% and SourceForge was a close second with 81.06%. GitLab had the smallest percentages of URIs available in Web archives with 64.29%. The distribution of the percent of mementos returned from each of the twelve Web archives is shown in Table 8. Internet Archive had the largest percent of all returned mementos with 58.68%. Internet Archive is followed by Bibliotheca Alexandrina Web Archive,[1] which returned 23.29% of all mementos. However, we note that since 2022 Bibliotheca Alexandrina has functioned as a backup to the Internet Archive and provides a mirror of the Internet Archive's holdings [50]. This could explain the high percentage of GHP URIs available in both the Internet Archive and Bibliotheca Alexandrina Web archives. The remaining 18.03% of mementos are distributed across the remaining 10 Web archives. All of the statistics shown in Figure 20 are summarized in Table 9.

[1]`https://www.bibalex.org/isis/frontend/archive/archive_web.aspx`

Table 9. The percent of URIs that were active, archived by Software Heritage, and archived by Web archives for all URIs and GitHub, GitLab, Bitbucket, and SourceForge.

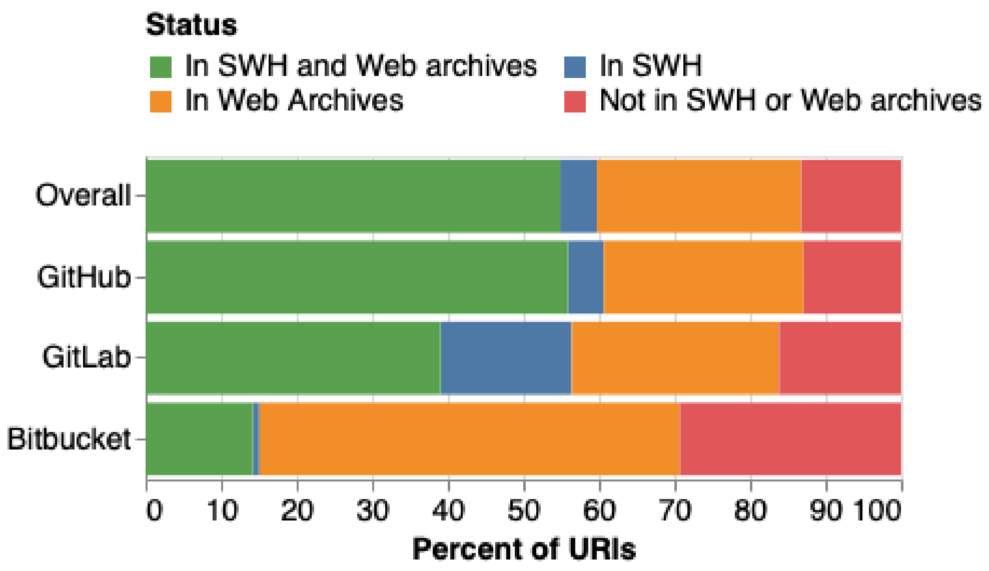
	Percent of URIs available in		
GHP	Live Web	SWH	Web archives
Overall	93.98%	67.52%	81.43%
GitHub	94.79%	69.34%	82.25%
GitLab	92.62%	85.03%	64.26%
Bitbucket	75.86%	16.93%	69.91%
SourceForge	80.71%		81.06%

Figure 21A depicts the percent of URIs archived by both Software Heritage and Web archives, only Software Heritage, only Web archives, and neither Software Heritage nor Web archives. Overall, 54.87% (66,855 URIs) of all URIs were captured by both Software Heritage and Web archives and 13.26% (16,456 URIs) were not captured by either Software Heritage or the Web archives, making their resources unrecoverable. Across all four GHPs, there are a higher percentage of GHP URIs that have only been archived by Web archives (27.02%) than the percentage of GHP URIs that have only been archived by Software Heritage (4.85%). Bitbucket has the highest percentage of URIs unique to the Web archives with 55.61% of Bitbucket URIs only archived by Web archives. Bitbucket also has the highest percentage of unrecoverable URI resources with 29.31%. Figures 21B and 21C give a more detailed look at the relationship between each category for GitHub and Bitbucket URIs.

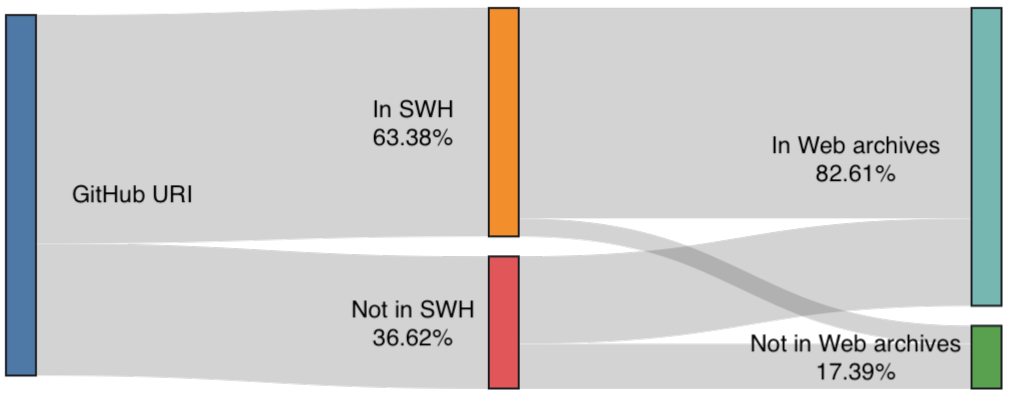
As shown in Figure 21B, 58.86% of GitHub URIs have been archived by both Software Heritage and Web archives, while 26.46% of GitHub URIs have only been archived by Web archives. These percentages noticeably differ from the distribution of Bitbucket URIs as depicted in Figure 21C. Of all Bitbucket URIs, 14.18% have been archived by both Software Heritage and Web archives while 55.61% have only been archived by Web archives. Additionally, only 0.90% of Bitbucket URIs are archived by Software Heritage and not archived by Web archives, compared to 4.77% of GitHub URIs.

Because they have active URIs, vulnerable resources still have the opportunity to be archived by Web archives and Software Heritage. However, rotten URIs are no longer able to be preserved. Figure 22A depicts the percent of rotten URIs that have been archived by both Software Heritage and Web archives, only Software Heritage, only Web archives, and neither Software Heritage or

A



B



C

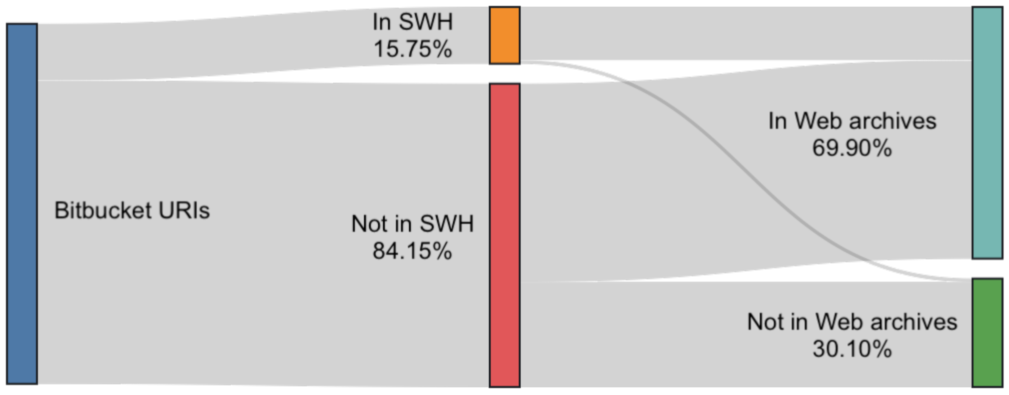


Fig. 21. Archival of GHP URIs. Percentage of URIs that have been archived by Software Heritage (SWH) and Web archives, only Software Heritage, only Web archives, and neither Software Heritage or Web archives *(A)* Percent of repository-level URIs overall and for each GHP *(B)* Relationship between the number of GitHub URIs preserved in each archive *(C)* Relationship between the number of Bitbucket URIs preserved in each archive.

Web archives. For GitHub, GitLab, and Bitbucket, 2,762 repository-level URIs are rotten and 33.35% of rotten URIs are unrecoverable, as they have not been archived by Software Heritage or Web archives. In total, there are 921 unrecoverable URI resources across the three GHPS. GitLab has the smallest percentage of rotten URIs that have been archived by both Software Heritage and Web archives with 5.77%. Conversely, Bitbucket has the largest percentage of rotten URIs that are captured by both Software Heritage and Web archives with 50.85%. For rotten URIs, there is a smaller percentage of URIs that have only been archived by Software Heritage (6.73%) than the percentage of URIs that have only been archived by Web archives (32.01%). This trend is similar to what we saw for all GHP URIs in Figure 21A. Figures 22B and 22C provide a more detailed look at the relationship between each category for rotten GitHub and Bitbucket URIs.

As shown in Figure 22B, 36.13% of rotten GitHub URIs are unrecoverable. Inversely, 23.64% of rotten GitHub URIs have been archived by both Software Heritage and Web archives. GitHub has a larger percentage of rotten URIs that have only been archived by Software Heritage (7.63%) than Bitbucket (2.14%) as shown in Figure 22C. Again, the distribution of rotten Bitbucket URIs is distinguishable from the distribution of rotten GitHub URIs. We found that 19.44% of rotten Bitbucket URIs are unrecoverable while 48.91% of rotten Bitbucket URIs have been archived by both Software Heritage and Web archives.

For both Software Heritage and Web archives, we calculated the time between the date of the first publication to reference a URI and the date of the first capture of the URI. Software Heritage was created on June 30, 2016 [51], so we only analyzed articles that were published starting July 1, 2016. We found an average of 443 days (median of 360 days) between the first reference to the repository URI in a scholarly publication and the first capture by Software Heritage, if the repository-level URI did not have a snapshot at the time of publication. Additionally, 7,440 repository URIs that were captured before the publication date of the referencing article had not been captured since the article's publication. For these URIs, there is an average of 253 days between the last Software Heritage snapshot and the publication date of the reference article.

As shown in Figure 23, the maximum time delta between the first reference to the repository URI in a scholarly publication and the first capture by Software Heritage has steadily decreased from 78 months for articles published in July 2016 to 9 months for articles published in April 2022. We also see that the median time delta follows a trend similar to the average time delta. The median and average time deltas have both decreased since 2021.

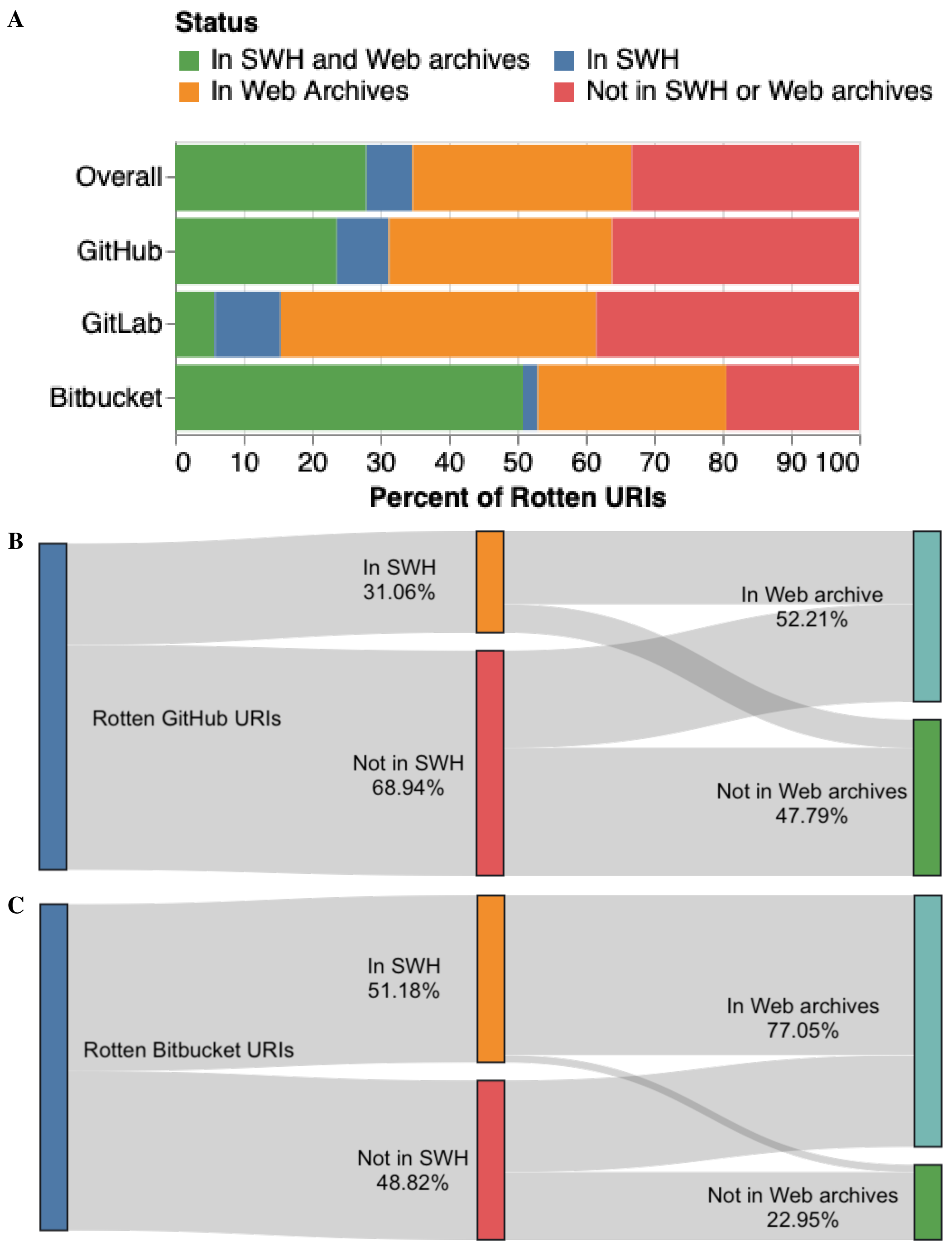


Fig. 22. Archival of GHP URIs. Percentage of rotten URIs that have been archived by Software Heritage (SWH) and Web archives, only Software Heritage, only Web archives, and neither Software Heritage or Web archives *(A)* Percent of rotten repository-level URIs overall and for each GHP *(B)* Relationship between the number of rotten GitHub URIs preserved in each archive *(C)* Relationship between the number of rotten Bitbucket URIs preserved in each archive.

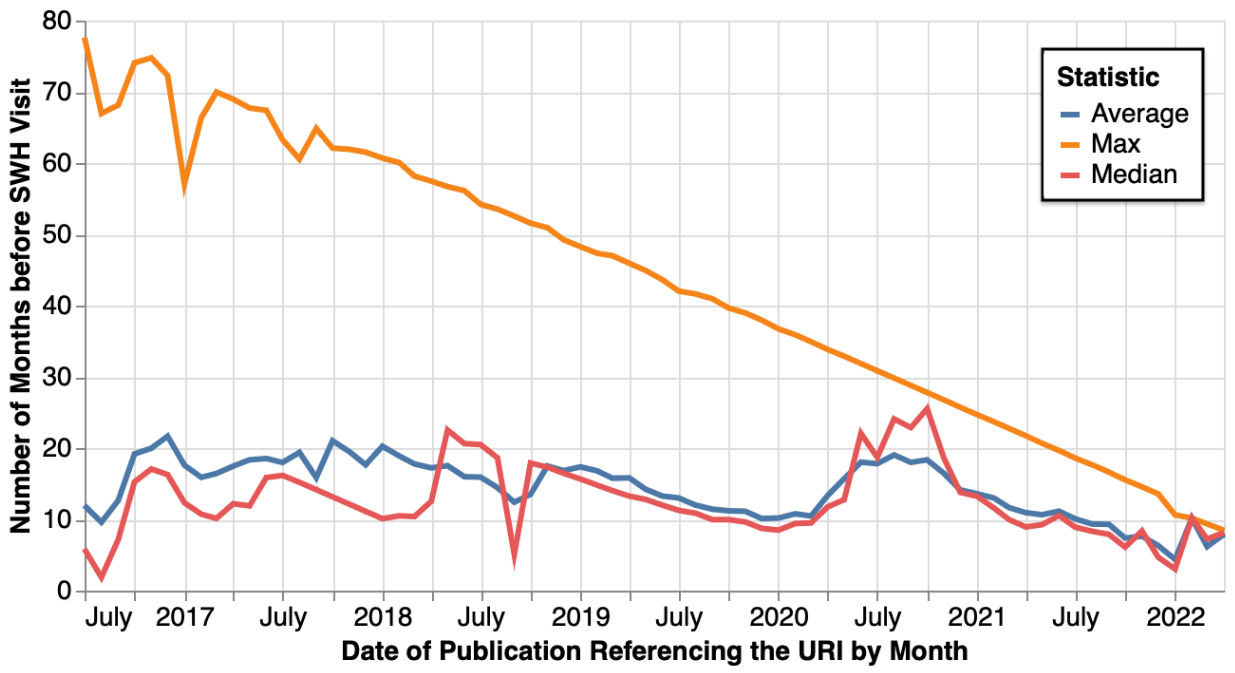


Fig. 23. Time delta (in months) between a publication referencing a URI and the URI being captured by Software Heritage Only includes URIs not been captured by Software Heritage before the publication date of the referencing article.

These trends for are similar for Web archives. There was an average of 468 days and a median of 341 days between the first reference to the URI in a scholarly publication and the first memento in a Web archive, if there were no mementos of the URI prior to the publication date of the referencing article. Of the URIs that had a memento in the Web archives prior to the publication date of the article, 4,356 URIs have not been archived since the article was published, with an average of 201 days between the latest memento and the publication date. Figure 24 shows that the average and maximum time deltas have followed similar trends. Additionally, the maximum time delta has steadily decreased from 128 months in January 2012 to 1 month in April 2022. While the steady decline seen in maximum and average time deltas for Software Heritage and Web archives is promising, there is still a large period of time for the URI resource to move from vulnerable to unrecoverable before Software Heritage or Web archives are able to archive it.

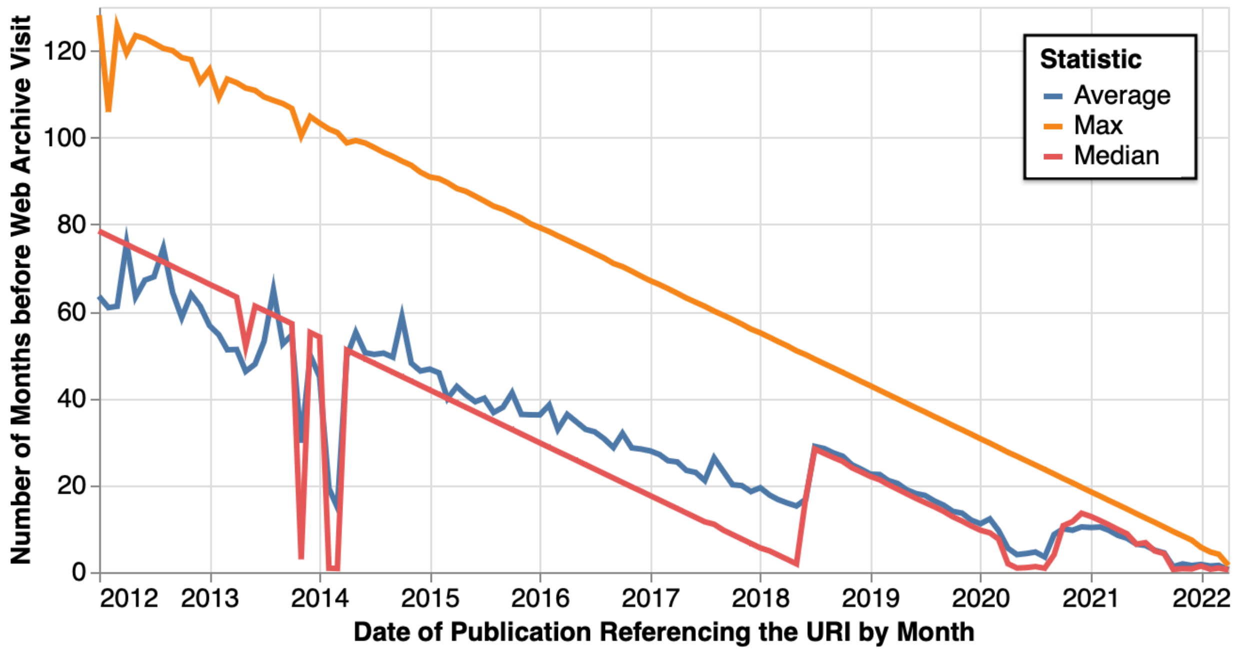


Fig. 24. Time delta (in months) between a publication referencing a URI and the URI being captured by the Web archives over time. Only includes URIs not captured by the Web archives before the publication date of the referencing article.

5.3 DISCUSSION

We analyzed the GHP URIs that were extracted in Chapter 4 from the arXiv and PMC corpora. GHP URIs from other corpora may produce a different result. For example, authors must proactively submit their paper to arXiv, which demonstrates an inclination to participate in open research. As such, authors who submit to arXiv may be more likely to submit source code projects to Software Heritage and Web archives for preservation and research reproducibility.

The smaller percentage of Bitbucket URIs publicly available on the live Web and preserved in Software Heritage may be correlated to the usage trends we observed in Chapter 4. Bitbucket was referenced in scholarly publications more than GitHub from 2008 to 2014, which could result in older publications containing a link to Bitbucket over other GHPs. As Klein et al. [20] found, the likelihood of reference rot grows as the age of the URI increases, which could be reflected in the lower percentage of SourceForge and Bitbucket URIs still publicly available on the live Web. Additionally, older GHP URIs may be less likely to be preserved in Software Heritage given that Software Heritage was launched June 30, 2016. Some of the GHP URIs that are not publicly available on the live Web may have disappeared long before Software Heritage even existed to preserve them.

Our analysis looked for the presence of a snapshot in Software Heritage and a memento in Web archives. However, we did not analyze whether the snapshot or memento captured the state of the software at the time it was referenced in the publication. We also did not assess the quality of the mementos provided by Web archives. In future work, the quality of mementos could be assessed to determine if the memento provides an adequate capture of the Web page to support reproducibility.

With 93.98% of URIs still publicly available on the live Web, these URIs still have the opportunity to be preserved. The 1.82% of URIs, or 921 repository URIs, that are unrecoverable should serve as a warning of what could happen if the research community does not act to preserve code products as integral research products. Researchers need to take initiative to submit code products to services like Software Heritage and Web archives to ensure the code they reference is preserved for long term access.

5.4 SUMMARY

In this chapter, we addressed the second research question: For the GHP URIs identified in scholarly articles, what is the prevalence of these GHP URIs (a) on the live Web, (b) in Software Heritage, and (c) in Web archives? We found that 93.98% of all GHP URIs in the corpus were still active, meaning that 8,882 GHP URIs were rotten. SourceForge had the highest rate of rotten URIs of the four GHPs with almost 19.29%. We also found that 67.52% of GHP URIs were archived in Software Heritage and 81.43% were archived in Web archives. However, the GHP URIs archived by Software Heritage and Web archives differed. Overall, 54.87% of the GHP URIs were captured by both Software Heritage and Web archives, 27.02% were captured only by Web archives, 4.85% were only captured by Software Heritage, and 13.26% were not captured by Software Heritage or Web archives. Additionally, the large period of time between the first reference to the URI in a scholarly publication and the first capture in a Web archive or Software Heritage can result in the existing captures not being representative of the resource the scholar intended. We found that 921 GHP URIs are unrecoverable, but content drift may result in more GHP URIs not having a representative capture in an archive.

CHAPTER 6

IDENTIFYING NON-GHP OADS URIS IN SCHOLARLY PUBLICATIONS

In the study described in Chapter 4, we decided to identify URIs to GitHub, GitLab, Bitbucket, and SourceForge as representative GHPs. However, there are numerous platforms that scholars use to publish the open access data and software (OADS) required for reproducibility. To identify other Web-based software and data hosting platforms, we implemented the classifier created by Salsabil et al. [35] introduced in Chapter 3. In this chapter, we investigated RQ3: Outside of GHPs, what OADS URIs are scholars including in their publications? These results were first presented in our publication "It's Not Just GitHub: Identifying Data and Software Sources Included in Publications" [52].

While the delineation between OADS and non-OADS may seem clear at first glance, it is more nuanced when we look at current citation trends. For example, authors may reference a publication that introduces or discusses a dataset or software product instead of including a direct link to the hosting platform itself. This tendency may be due to the value of publication citations within academia or due to established practices within a discipline or institution, but it results in the possibility of indirect links to OADS via paper publications. For example, ScienceDirect is a digital library of journal articles and book chapters which are non-OADS, but indirect links to OADS could result in ScienceDirect URIs being classified as OADS. Figure 25 shows a reference to a ScienceDirect publication being cited in an arXiv article[1] in the context of the author listing out available packages for solving DMFT equations. Figure 26 shows the reference for the ScienceDirect publication. The ScienceDirect publication was classified as an OADS URI by our machine learning classifier model despite it being a paper publication. While the citation itself is to a paper publication, the author is using the citation to indicate a software package discussed in the publication.

[1]`https://arxiv.org/abs/2002.00068`

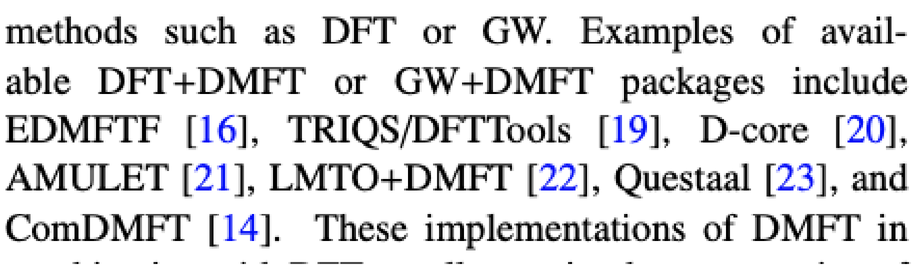

methods such as DFT or GW. Examples of available DFT+DMFT or GW+DMFT packages include EDMFTF [16], TRIQS/DFTTools [19], D-core [20], AMULET [21], LMTO+DMFT [22], Questaal [23], and ComDMFT [14]. These implementations of DMFT in

Fig. 25. Citation of the ScienceDirect publication. In the arXiv publication (`https://arxiv.org/abs/2002.00068`), the author is listing out available software packages and includes a citation to the ScienceDirect publication and URI.

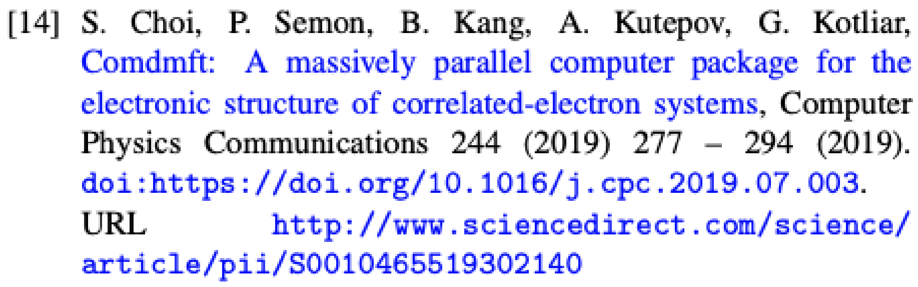

[14] S. Choi, P. Semon, B. Kang, A. Kutepov, G. Kotliar, Comdmft: A massively parallel computer package for the electronic structure of correlated-electron systems, Computer Physics Communications 244 (2019) 277 – 294 (2019). doi:https://doi.org/10.1016/j.cpc.2019.07.003. URL http://www.sciencedirect.com/science/article/pii/S0010465519302140

Fig. 26. The reference for a ScienceDirect publication cited in an arXiv publication (`https://arxiv.org/abs/2002.00068`) and classified as an OADS URI despite being a paper publication.

6.1 METHODOLOGY

For this experiment, we used the same arXiv corpus analyzed in Chapters 4 and 5. To determine whether a URI links to an open access dataset or software resource, we used the classifier introduced in Chapter 3 for each article in the corpus. After all of the URIs have been classified, we filtered out URIs that were out of scope for this study. Because we wanted to focus on data and software repositories, we filtered out URIs that would likely point to publications such as URIs to arXiv, Elsevier RefHub, CrossRef Crossmark, and some HTTP DOIs, similar to the filtering process in Chapter 4. However, because we were working to identify URIs to data and software, we chose to include DOIs to Zenodo, Dryad, figshare, and Open Science Framework (OSF), as they are known to resolve to data and software artifacts, while removing all other DOIs.

We used the regular expressions introduced in Chapter 4 to identify URIs to GitHub, GitLab, SourceForge, and Bitbucket from the extracted URIs. Collectively, we will refer to URIs to one of these four Git hosting platforms (GHPs) as GHP URIs. OADS URIs that are not URIs to one of these four GHPs will be referred to as non-GHP OADS URIs.

6.2 RESULTS

With the extracted and classified URIs, we looked at the overall distribution of URIs and the distributions of URIs classified as OADS and non-OADS. In Figure 27, we looked at the average number of OADS, non-OADS, and total URIs per publication. The average number of URIs, OADS URIs, and non-OADS URIs per publication rose steadily from 2007 to 2021. In 2007, there were an average of 0.416 URIs per publication with 0.111 OADS URIs per publication and 0.306 non-OADS URIs per publication. Those averages nearly tripled across all three categories by 2021. In 2021, there were an average of 1.273 URIs per publication with 0.433 OADS URIs per publication and 0.841 non-OADS URIs per publication. This shows that authors have been increasingly including URIs, both OADS and non-OADS URIs, in their publications. With a growing number of included URIs comes a growing need to archive the resources that these authors are including in their research with the understanding that authors included the URIs because they were important to their study or were a result of their research.

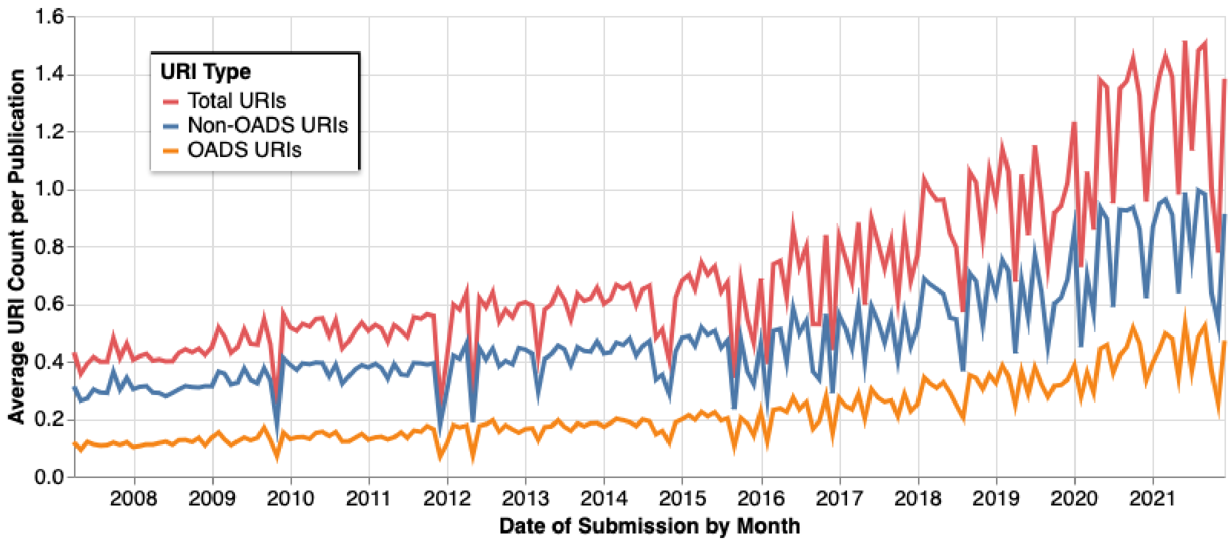


Fig. 27. Average number of URIs per arXiv pre-print by publication date. The blue line represents the number of URIs our machine learning model classified as non-OADS as an average per publication (y-axis) per publication month (x-axis). The orange line represent the number of URIs our machine learning model classified as OADS as an average per publication. The red line represents the total number of URIs we extracted from the publications as an average per publication.

With an understanding of the general trends of URI usage, we next looked at the distribution of OADS and non-OADS URIs. We also separated the GHP URIs from the other OADS URIs to gain an understanding of the prevalence of GHP URIs over time. We chose GitHub, GitLab, Bitbucket, and SourceForge as popular GHPs to represent GHP URIs. As shown in Figure 28, we found that both the prevalence and the distribution of the URIs changed across the time period. The percentage of non-OADS URIs has slightly declined meaning that authors are including a higher proportion of OADS URIs to non-OADS URIs in recent years. The percentage of GHP URIs has significantly increased from less than 1% in 2007 to around 15% of all URIs in 2021. Despite the overall increase in the prevalence of OADS URIs seen in Figure 27, there has been a decrease in the percentage of non-GHP OADS URIs as shown in Figure 28. This means that the growth in the prevalence of OADS URIs has largely been due to an increase in the inclusion of GHP URIs within publications.

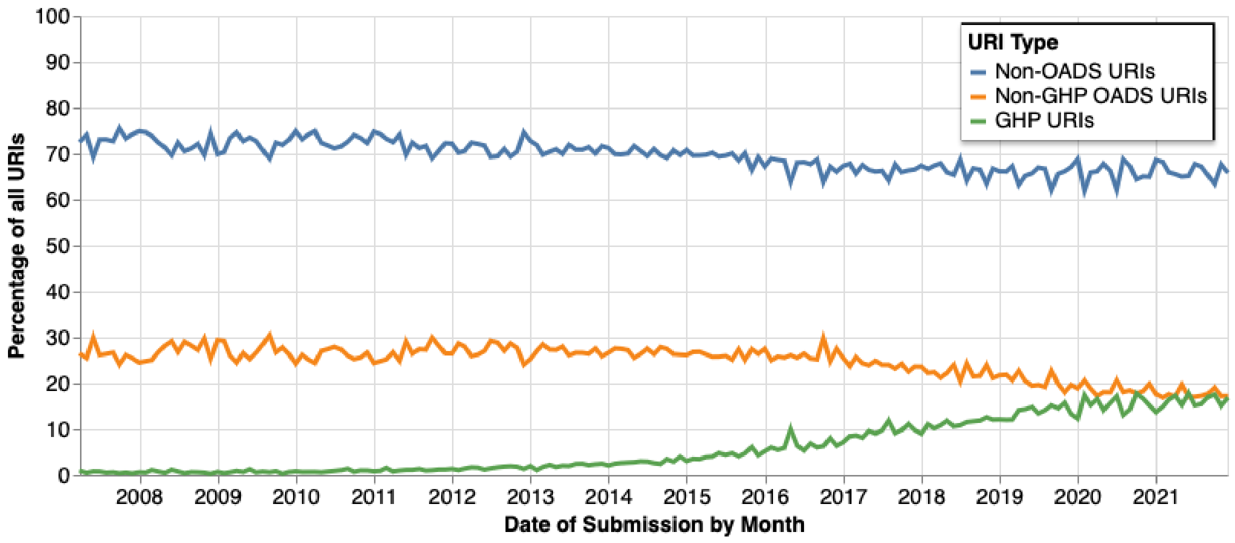


Fig. 28. Percentage of GHP URIs, non-GHP OADS URIs, and non-OADS URIs by publication date. The blue line represents the percent of URIs our machine learning model classified as non-OADS (y-axis) per publication month (x-axis). The orange line represents the percent of URIs our machine learning model classified as OADS, excluding GHP URIs. The green line represents the percent of URIs that were GHP URIs.

The increase of the prevalence of GHP URIs is also reflected when we look at the total number of GHP and OADS URIs over time in Figure 29. From 2007 to 2015, there were a 500 to 1000 more non-GHP URIs than GHP URIs. In 2015, the number of GHP URIs started to steadily increase. In 2020 and 2021, for every GHP URI, there is a non-GHP OADS URI. This shows that utility of using a classifier to identify OADS URIs, especially in older publications from 2007 to 2015. We also see that, while GitHub is an independently popular GHP, we must look beyond GitHub to identify and discover the full breadth of OADS resources being referenced and produced by researchers even in recent year.

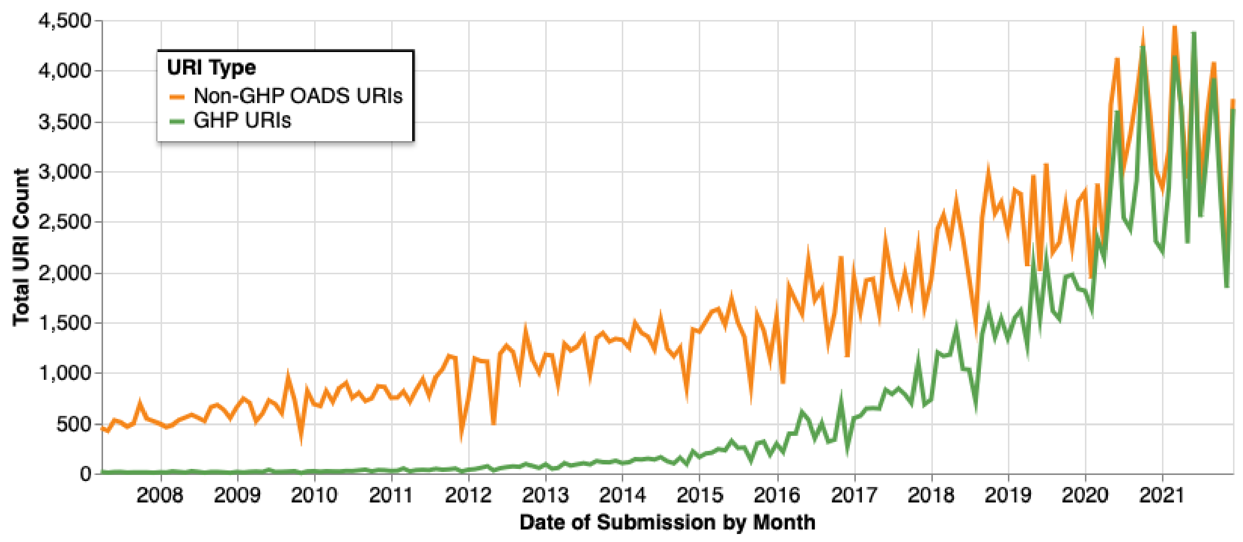


Fig. 29. Total number of GHP URIs and non-GHP OADS URIs by publication date.

After seeing the trends over time, we wanted to identify the most common non-GHP OADS URIs. We chose to compare URI hostnames and the frequency of those hostnames to determine the most common OADS websites outside of GHPs. In total, we found 258,288 non-GHP OADS URIs included in arXiv publications and almost 50,000 unique hostnames[2] within those URIs. Figure 30 shows that 49,392 hostnames are included in between 0 and 50 non-GHP OADS URIs. We found that 63% of non-GHP OADS URIs are the only URIs to that hostname and only 10% of URIs reference a hostname that is referenced more than five times. Even with a large number of hostnames referenced a few number of times, there are 19 hostnames that were referenced over 1000 times. Table 10 shows the the top fifteen most common hostnames of non-GHP OADS URIs. However, it is worth noting that the most popular hostname, cds.cern.ch, only accounts for 1.92% of all non-GHP OADS URIs. Therefore, there are a large number of platforms used by scholars to host data and software which increases the difficulty of archiving data and software products for reproducibility. The diversity of the platforms used and referenced by scholars makes it difficult to manually identify OADS URIs and lends itself to automation like we used with the machine learning classifier model.

[2]The full dataset is available at `https://github.com/oduwsdl/Extract-URLs/blob/main/classifier_results/count_oads_non_ghp_hostnames.csv`

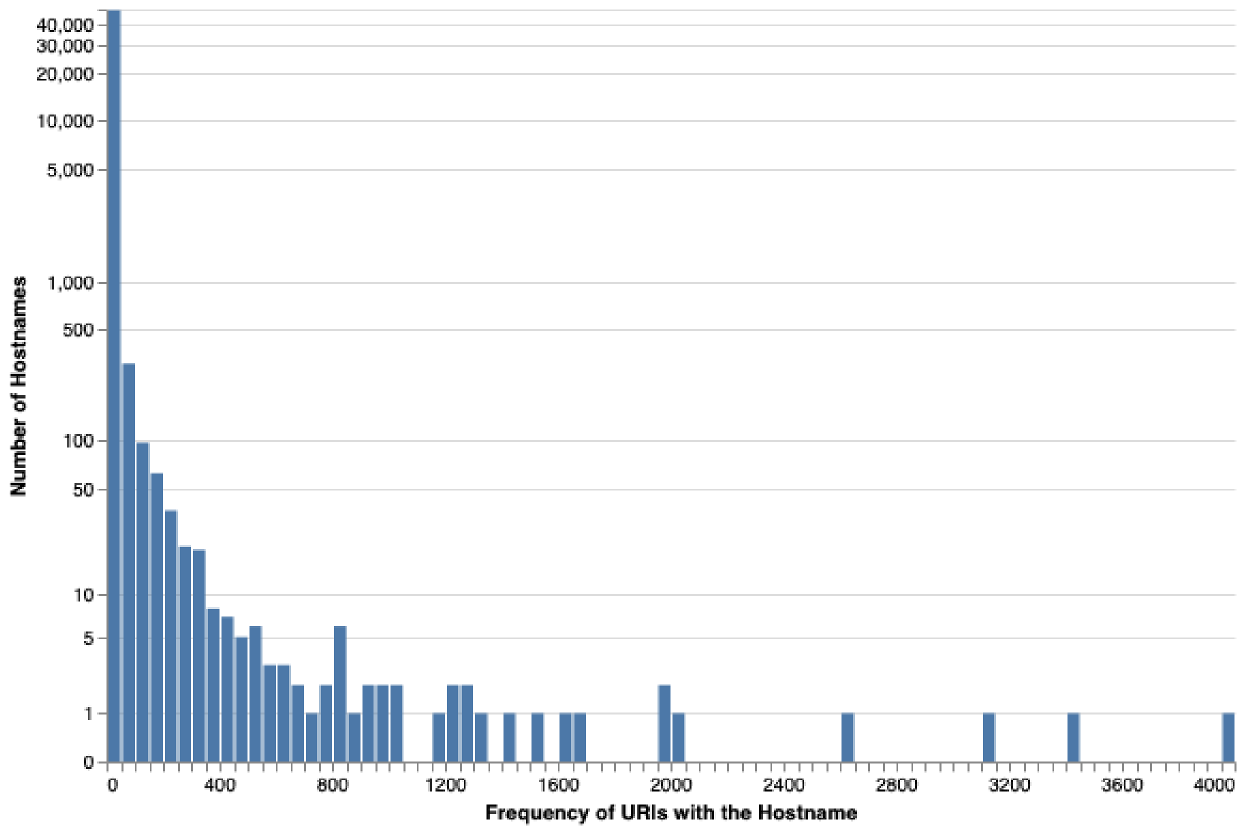


Fig. 30. Hostname frequency. A histogram showing the frequency of the hostname in non-GHP OADS URIs (x-axis) and the number of hostnames that shared that frequency (y-axis).

6.3 DISCUSSION

Our analysis found that a significant portion of OADS URIs are not GHP URIs. This shows that, while it is simple to search for OADS URIs by regular expression, regular expressions cannot detect all OADS URIs. Additionally, while the top fifteen most common hostnames are popular platforms for research artifacts, a majority of OADS URIs were to platforms archivists may not know to look for like `www.physics.wisc.edu`, `www.broadinstitute.org`, `fuse.pha.jhu.edu`. Using a classification system like we used in this study, allows us to cast a broader net and detect OADS URIs to lesser known platforms for preservation. While GitHub, GitLab, SourceForge, and Bitbucket accounted for 127,529, or 33%, of the 385,817 OADS URIs extracted from the arXiv corpus, focusing archival efforts on these and other popular GHPs would miss 67% of the OADS resources included by researchers.

Table 10. The top 15 most common hostnames for non-GHP OADS URIs and their frequencies.

Hostname	Frequency
cds.cern.ch	4,953
www.sciencedirect.com	3,119
archive.ics.uci.edu	2,632
adsabs.harvard.edu	2,031
www.ncbi.nlm.nih.gov	1,998
www.cosmos.esa.int	1,996
physics.nist.gov	1,651
fermi.gsfc.nasa.gov	1,627
heasarc.gsfc.nasa.gov	1,500
cran.r-project.org	1,446
doi.org	1,337
www.w3.org	1,289
www.nature.org	1,275
archive.stsci.edu	1,243
en.wikipedia.org	1,228

Our machine learning classifier model, despite good performance found in previous studies [35], fine tuning, and a large training set, was not perfect, as can be expected when extracting and classifying millions of URIs from 1.58 million scholarly articles. It incorrectly classified some GHP URIs as Non-OADS. In some cases, these GHP URIs were located in the footnote or in other locations that lacked the necessary context sentence around the target URI for proper classification. Despite the limitations and inaccuracies, we remain confident that utilizing machine learning models to classify OADS and non-OADS URIs will allow researchers and archivists to more easily identify less popular or niche platforms for preservation.

6.4 SUMMARY

In this chapter, we addressed the third research question: Outside of GHPs, what OADS URIs are scholars including in their publications. Similar to the trend for GHP URIs in Chapter 4, we found that the prevalence of OADS URIs has increased. In 2021, there were an average of 1.273 URIs per publication with 0.433 OADS URIs per publication with a higher proportion of OADS URIs to non-OADS URIs. We also found that the number of GHP URIs and non-GHP OADS URIs is nearly equal in 2020 and 2021. However, there are more non-GHP OADS URIs from 2007 to 2020. We identified 49,392 unique hostnames across the corpus. This shows that using a classifier can help identify valuable resources from a variety of hosting platforms that would otherwise be difficult to enumerate.

CHAPTER 7

RELATIONSHIP BETWEEN REPOSITORY POPULARITY AND ARCHIVAL RATE

In Chapter 5, we investigated the prevalence of GHP URIs in Software Heritage (SWH) and Web archives (WA). However, as we discussed in Chapter 3, the popularity of a URI is related to its archival. In this chapter, we investigated RQ4: Does popularity as measured by user engagement affect the likelihood of a GitHub repository being archived?

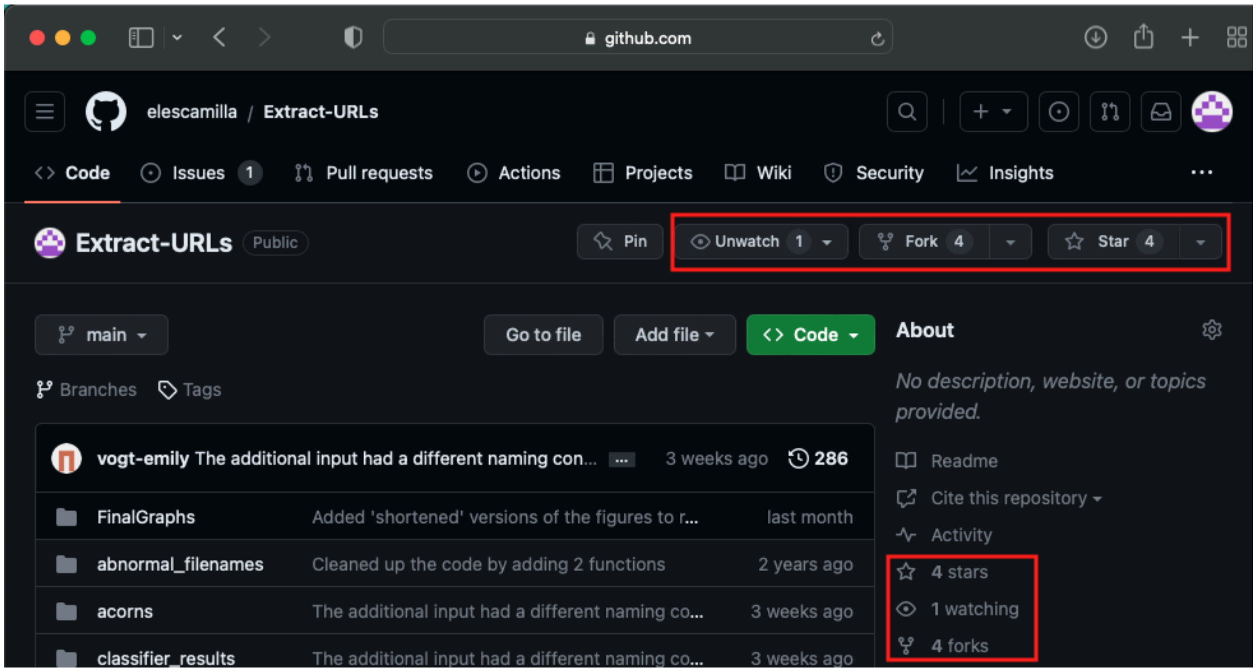
7.1 METHODOLOGY

For this study, we analyzed the relationship between user engagement and archival. In following the studies by Färber [27] and Bhattarai et al. [29] as well as our finding that 94% of GHP URIs are to GitHub alone, we decided to focus on GitHub repository URIs. We used the GitHub API to extract engagement metrics for all GitHub repository URIs identified in Chapter 4. An example of the GitHub API response is shown in Appendix H. We used forks, subscribers, and stargazers metrics to define engagement. A fork is a new copy of the repository that is separately hosted in GitHub.[1] This is different from cloning the repository, a git feature which makes a local copy of the repository on your local machine. Forks also allow for easy collaboration and interaction with the upstream repository that the repository was forked from. The terminology for watchers, subscribers, and stargazers has evolved. As of 2012, the watchers metric shown in the GitHub API has been split into two metrics: subscribers and stargazers.[2] Subscribers have subscribed to notifications for activity in the repository, while stargazers have bookmarked the repository but do not receive notifications. The subscriber terminology in the API response allows for continued support of watch endpoints in legacy apps.[3] In the browser UI, subscribers are shown as watchers (Figure 31) which adds to the confusion with the terminology. For our purposes, we used the `forks_count`, `subscribers_count`, and `stargazers_count` values from the GitHub API response.

[1]https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/working-with-forks/about-forks

[2]https://docs.github.com/en/rest/activity/watching?apiVersion=2022-11-28

[3]https://github.blog/2012-09-10-watcher-api-changes/



No description, website, or topics provided.

vogt-emily The additional input had a different naming con...	3 weeks ago	286
FinalGraphs	Added 'shortened' versions of the figures to r...	last month
abnormal_filenames	Cleaned up the code by adding 2 functions	2 years ago
acorns	The additional input had a different naming co...	3 weeks ago
classifier_results	The additional input had a different naming co...	3 weeks ago

Fig. 31. Engagement metrics in GitHub UI. The engagement metrics are shown on the main page for each GitHub repository. Notice that the term "watch" is used in the browser UI.

We conducted the same statistical analysis as Bhattarai et al. [29] by using a Mann Whitney U test to compare the distribution of the populations and Cliff's Delta (δ) to measure the effect size, or the strength of the relationship between two variables in the population. The Mann-Whitney U [53] test is a nonparametric test of the null hypothesis that

$$Pr(X > Y) = Pr(Y > X)$$

for randomly selected values X and Y from two populations. We established six null hypotheses, one for each engagement metric and archive:

1. For randomly selected *fork counts* X and Y from two populations (in SWH and not in SWH), the probability of X being greater than Y is equal to the probability of Y being greater than X.

2. For randomly selected *subscriber counts* X and Y from two populations (in SWH and not in SWH), the probability of X being greater than Y is equal to the probability of Y being greater than X.

3. For randomly selected *stargazer counts* X and Y from two populations (in SWH and not in SWH), the probability of X being greater than Y is equal to the probability of Y being greater than X.

4. For randomly selected *fork counts* X and Y from two populations (in WA and not in WA), the probability of X being greater than Y is equal to the probability of Y being greater than X.

5. For randomly selected *subscriber counts* X and Y from two populations (in WA and not in WA), the probability of X being greater than Y is equal to the probability of Y being greater than X.

6. For randomly selected *stargazer counts* X and Y from two populations (in WA and not in WA), the probability of X being greater than Y is equal to the probability of Y being greater than X.

We used the Mann-Whitney U test function (`mannwhitneyu()`) from SciPy[4] for each of three engagement metrics.

Cliff's Delta is a measure of how often the values in one distribution are larger than the values in a second distribution [54]. Cliff's Delta can be understood as the probability that a randomly selected value from one group is larger than a randomly selected value from the second group If the Mann-Whitney U test answers the question"Is there a difference between the populations?", Cliff's Delta answers the question "How big is the difference?" We used the `cliffs-delta` Python package[5] to implement the Cliff's Delta calculation. We used the same categorization values used by Bhattarai et al. [29] to transform the numeric value to a meaningful category: negligible if $|\delta| < 0.12$, small if $|\delta| \in (0.12, 0.28)$, medium if $|\delta| \in (0.28, 0.43)$, and large if $|\delta| > 0.43$. The statistics for each population for each of the three engagement metrics is shown in Table 11

7.2 RESULTS

The numerical results of the Mann Whitney U test and Cliff's Delta statistical analysis for each of the three engagement metrics and both archives is shown in Table 12. In this case, the p-value is so close to zero that it cannot be represented with a floating point. Therefore, there is nearly 100% confidence that all six of the null hypotheses stated above can be rejected, indicating that there is a statistically significant difference between the distribution of each population. When comparing

[4]`https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.mannwhitneyu.html`
[5]`https://pypi.org/project/cliffs-delta/`

Table 11. Population statistics for each population for the three engagement metrics.

Metric	Archival Status	Median	Population Size
Fork	In SWH	5	71,921
	Not In SWH	2	37,054
	In WA	4	105,816
	Not In WA	1	26,723
Subscribers	In SWH	4	71,921
	Not In SWH	2	37,054
	In WA	4	105,816
	Not In WA	2	26,723
Stargazers	In SWH	14	71,921
	Not In SWH	6	37,054
	In WA	14	105,816
	Not In WA	4	26,723

the populations of repositories archived in Software Heritage and not in Software Heritage, the effect size calculated with Cliff's Delta and effect categorization show that both the number of forks and the number of stargazers are correlated with a medium difference between the two populations while the number of subscribers is correlated with a small difference between the two populations. The difference in the distributions is also shown in Figures 32, 33, and 34. The repositories that have been archived in Software Heritage are indicated in orange while the repositories that have not been archived in Software Heritage are indicated in blue. The repositories that have been archived in Software Heritage have a longer tail than the repositories that have not been archived, showing that the repositories with higher engagement are more likely to be archived by Software Heritage.

Table 12. Results of the statistical analysis for each of the three engagement metrics: forks, subscribers, and stargazers.

Metric	Archive	U-value	p-value	Effect Size	Effect Categorization
Forks	SWH	143456778.5	0.0	-0.31946	Medium
	WA	1125018125.0	0.0	-0.20429	Small
Subscribers	SWH	157292028.0	0.0	-0.25383	Small
	WA	1113908954.5	0.0	-0.21215	Small
Stargazers	SWH	142883042.0	0.0	-0.32218	Medium
	WA	1125201331.5	0.0	-0.20416	Small

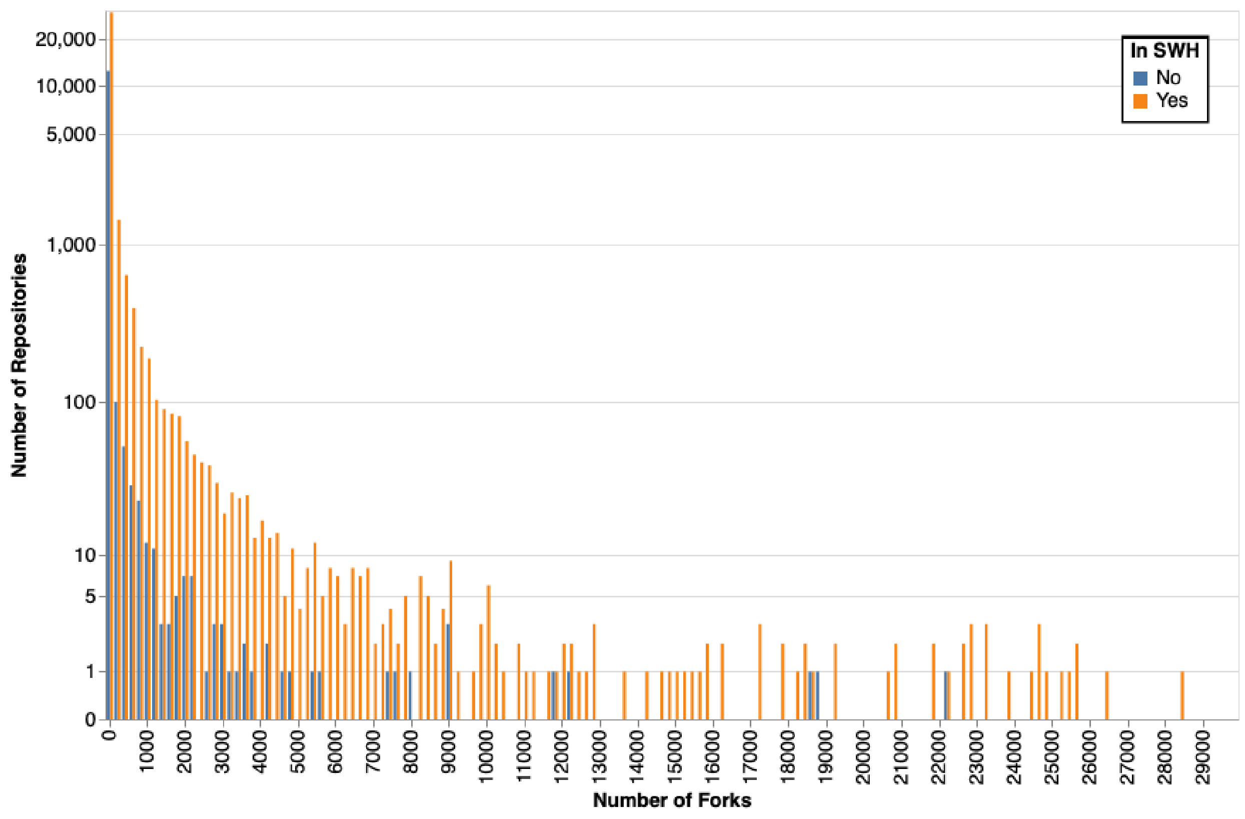


Fig. 32. The distribution of forks for repositories archived and not archived in Software Heritage.

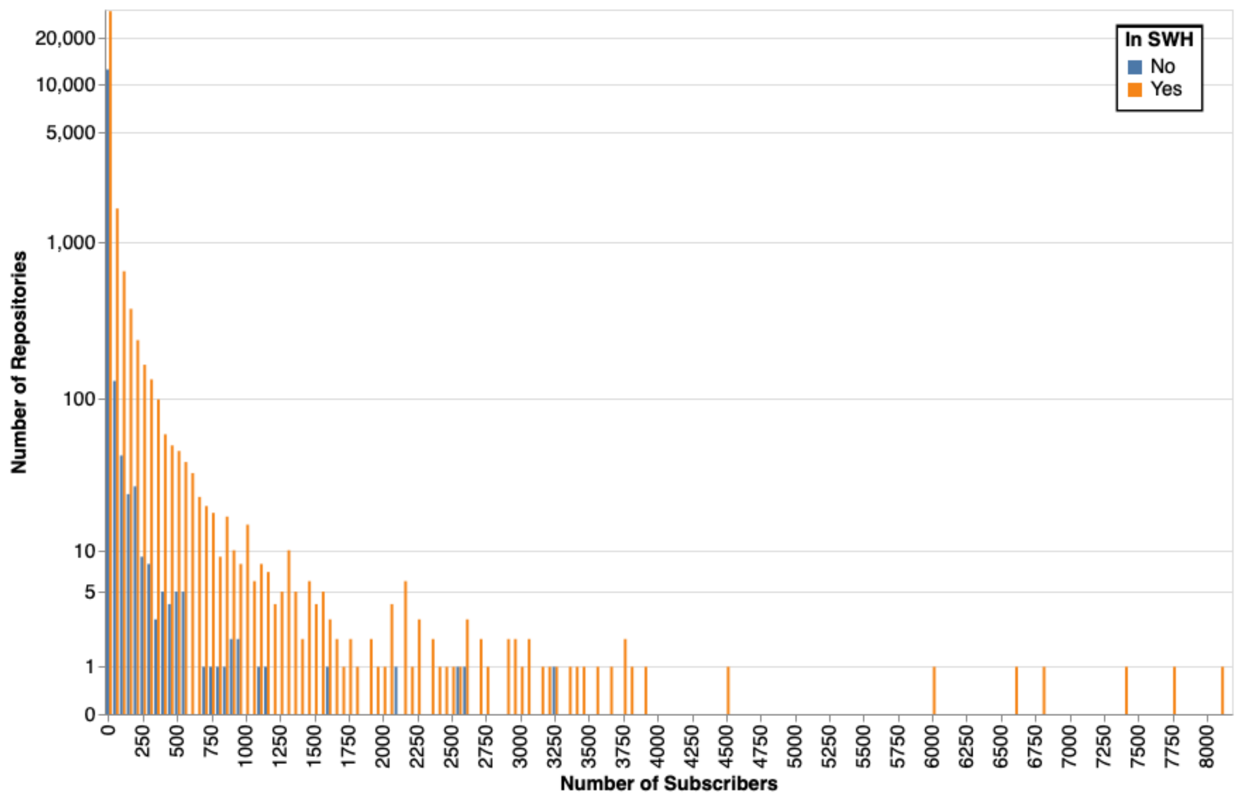


Fig. 33. The distribution of subscribers for repositories archived and not archived in Software Heritage.

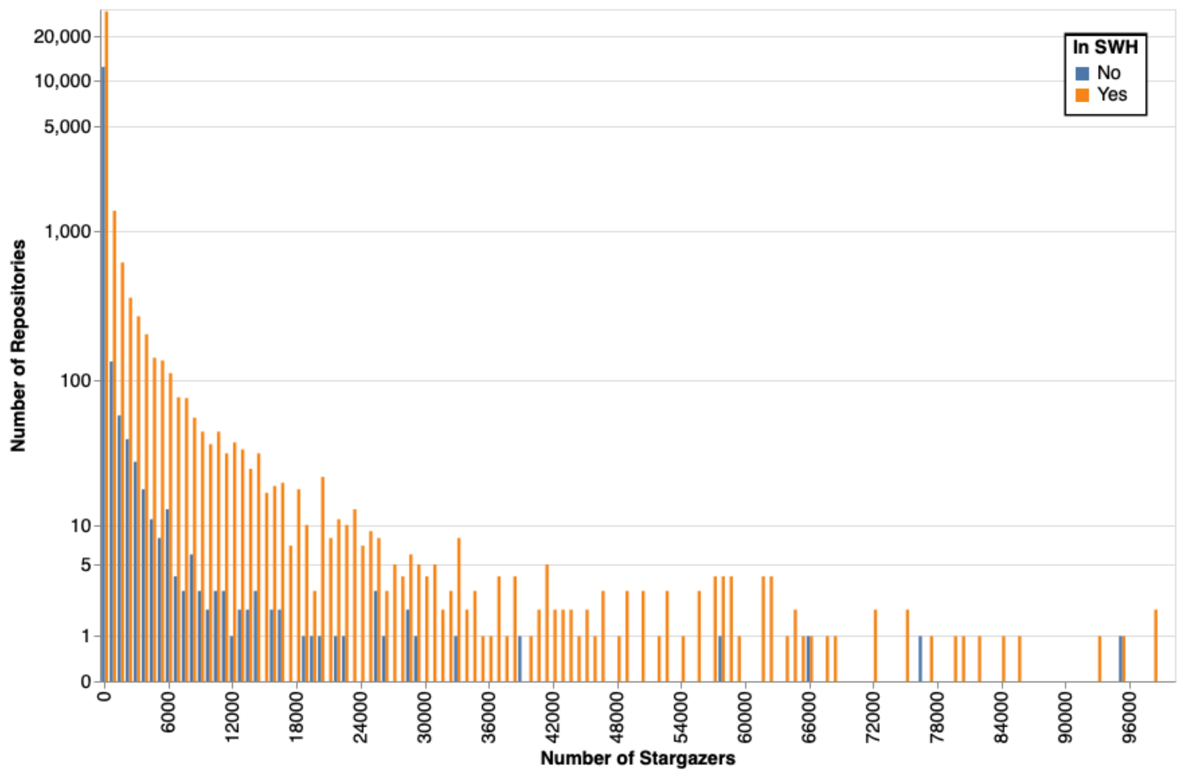


Fig. 34. The distribution of stargazers for repositories archived and not archived in Software Heritage.

As shown in Table 12, the effect size was categorized as small for all three engagement metrics when comparing repositories archived by Web archives and not archived by Web archives. As stated above, Cliff's Delta can be understood as a probability. Therefore, for the subscribers engagement metric, there is a one in four chance that the number of subscribers for a randomly selected repository that has not been archived by Software Heritage will be less than the number of subscribers for a randomly selected repository that has been archived by Software Heritage. This understanding adds context to the effect size for the Web archives populations. For all three metrics, the probability that a randomly selected repository that is not in the Web archives will have a smaller engagement value than a randomly selected repository that is in the Web archives is one in five. This decrease in the effect size can be seen when we use a histogram to plot the engagement metrics for each population as shown in Figures 35, 36, and 37. The repositories that have been archived in Web archives are indicated in orange while the repositories that have not been archived in Web archives are indicated in blue. Unlike the Software Heritage populations, the distinctions between in the Web archives populations is not as clear for all three engagement metrics.

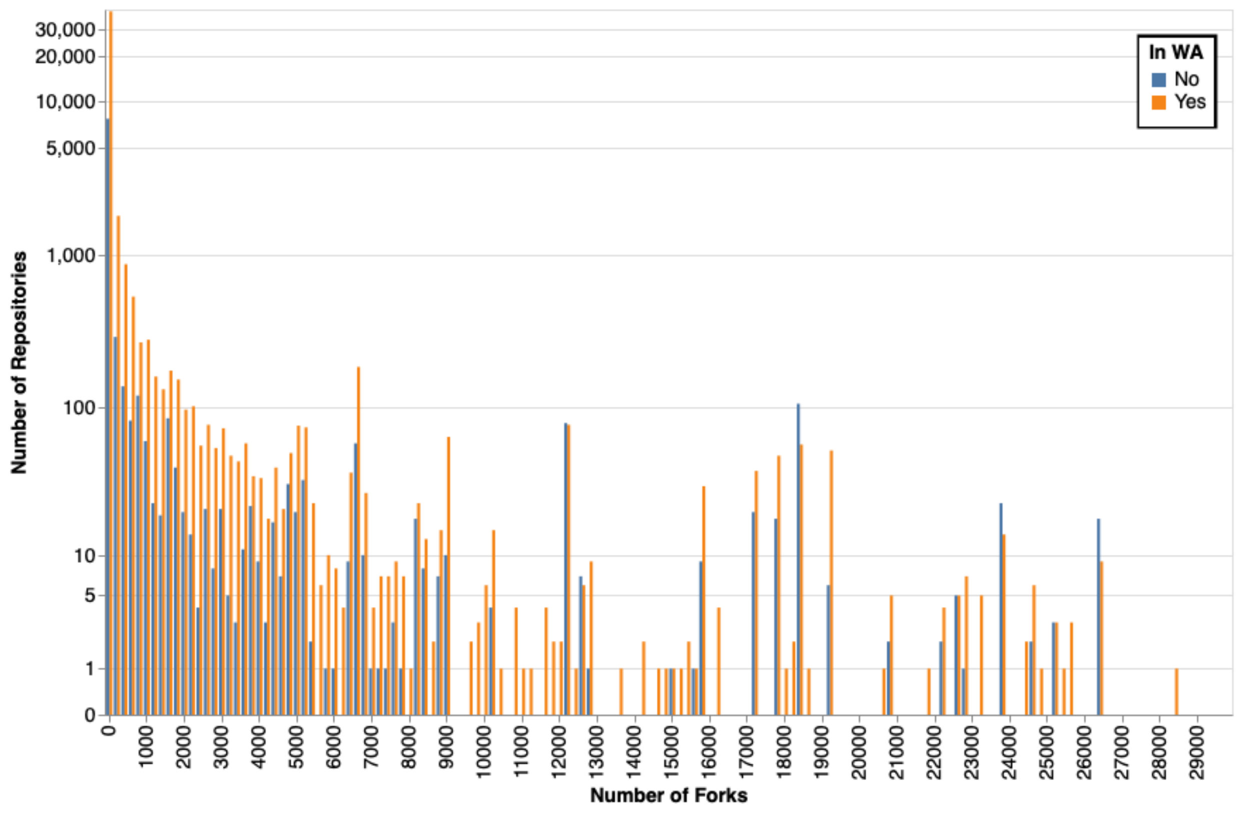


Fig. 35. The distribution of forks for repositories archived and not archived in Web archives.

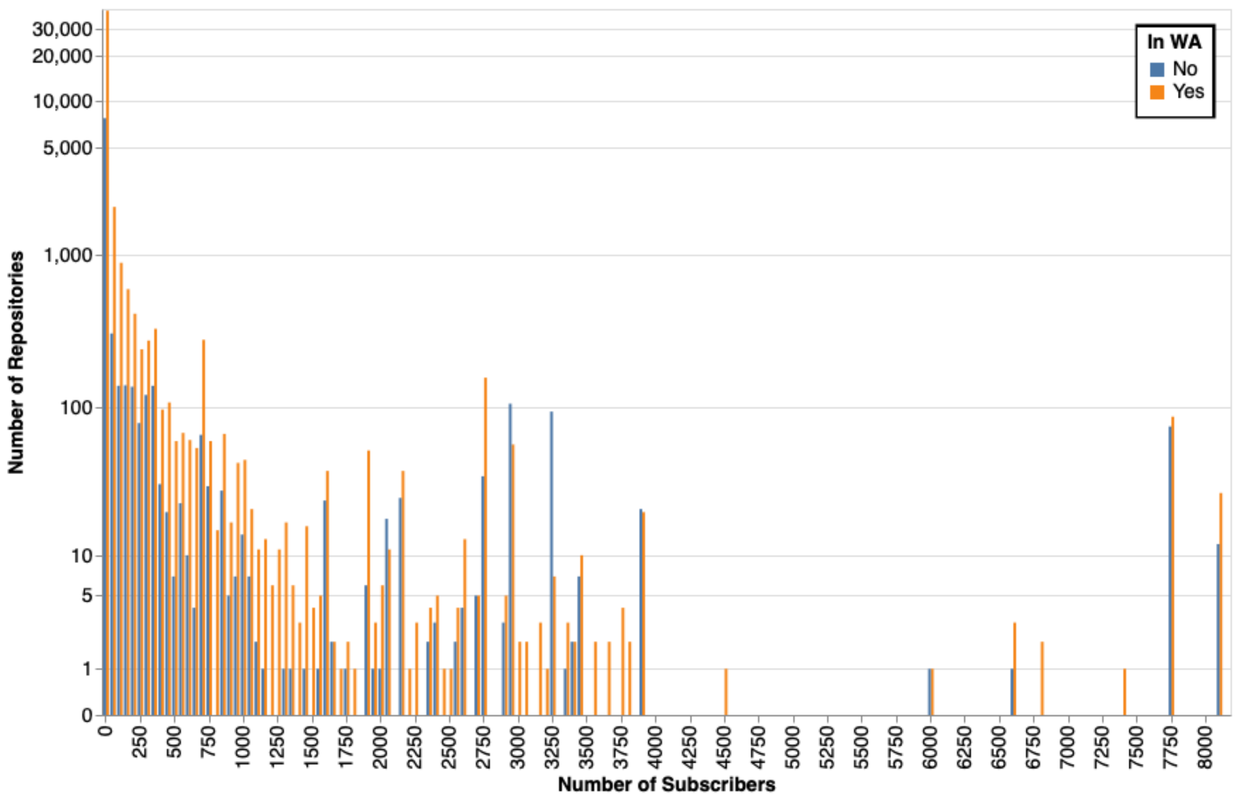


Fig. 36. The distribution of subscribers for repositories archived and not archived in Web archives.

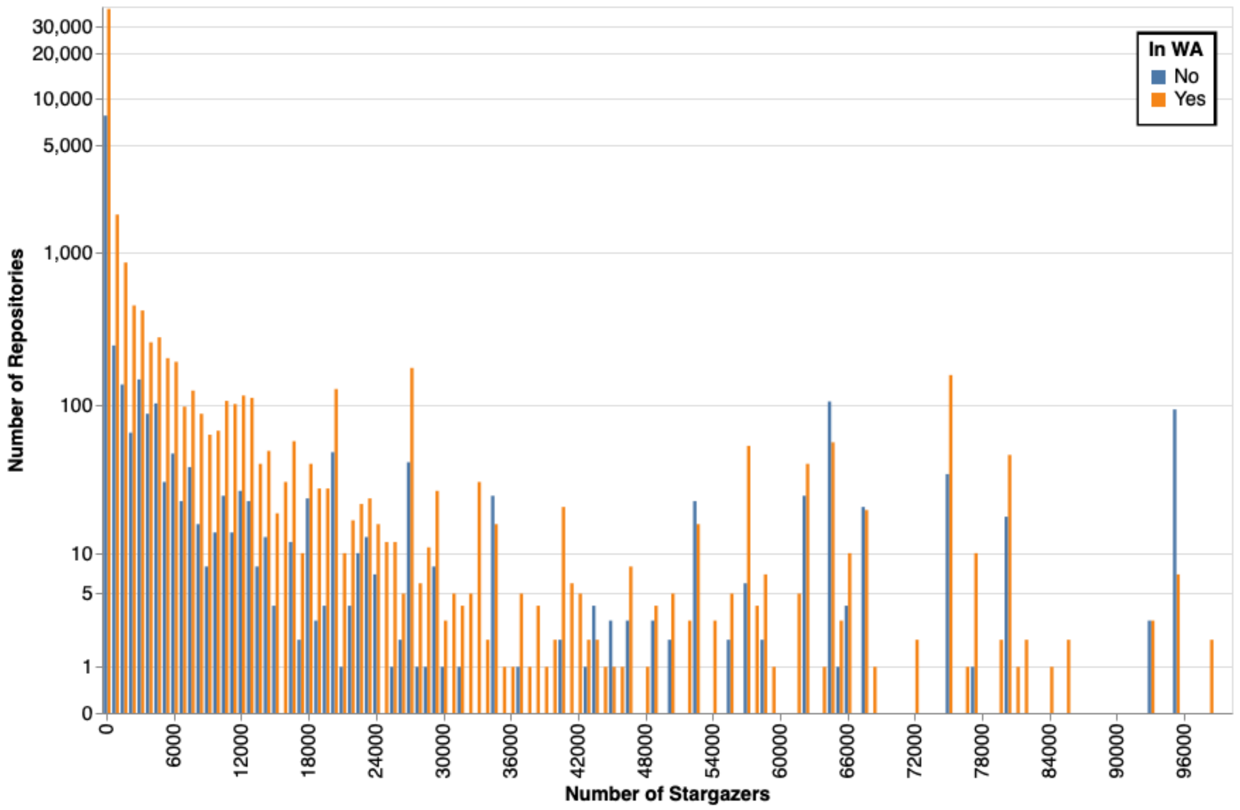


Fig. 37. The distribution of stargazers for repositories archived and not archived in Web archives.

7.3 DISCUSSION

For GitHub repositories in Software Heritage, there is a stronger correlation between the popularity of the repository and its archival than for Web archives. This may be caused by the higher portion of GitHub URIs available in Web archives (79.56%) than in Software Heritage (63.15%) that we found in Chapter 5. If Web archives have mementos for more GitHub repositories than Software Heritage, the Web archives are more likely to have mementos for less popular GitHub repositories. This may be due to differences in crawling patterns, crawling resources, and/or manual user submissions. However, there is a statistically significant difference between the engagement metrics for GitHub repositories that have and have not been archived in Web archives and Software Heritage. Therefore, smaller, less popular repositories are at increased risk of being vulnerable and, eventually, unrecoverable. This points to the need for researchers to submit their scholarly GHP URIs to archives for long term preservation especially if they are less popular.

We focused our analysis on GitHub repositories. Less popular GHP repositories like GitLab, Bitbucket, and SourceForge may have a different correlation between popularity and archival depending on the method of archival. For example, researchers who use SourceForge may be more likely to self-archive their repository in Web archives and/or Software Heritage which would reduce the impact of popularity on archival. Repository popularity may be more strongly impacted by incidental archiving, or allowing the archive crawlers to identify URIs, than by self-archiving.

7.4 SUMMARY

In this chapter, we addressed the fourth and final research question: Does popularity as measured by user engagement affect the likelihood of a GitHub repository being archived? We found that the GitHub repositories that have been captured by Software Heritage have significantly higher engagement metrics from a statistical perspective. We found the same correlation between engagement metrics and presence in Web archives. Therefore, the popularity of the GitHub repository does positively effect the likelihood of a Github repository to be archived for both Software Heritage and Web archives.

CHAPTER 8

FUTURE WORK AND CONCLUSIONS

Reproducibility is a foundational principle of scientific research and it is contingent on the availability of the original methodology, including software products, and data. The renewed emphasis on reproducibility by scholarly institutions and publishers reflects the importance of making software products and datasets available for the long-term. We discussed the current archival efforts to preserve software products including Web archives and Software Heritage. We also discussed the importance of archiving the ephemera that surrounds the software products hosted in GHPs

To understand the importance of archiving software products, we studied the prevalence of URIs to four Git Hosting Platforms: GitHub, GitLab, SourceForge, and Bitbucket. We extracted 253,590 GHP URIs from 2.64 million publications in the arXiv and PMC corpora. We found that the prevalance of URIs to the Web at large and GHP URIs has steadily increased since 2007 with one in five arXiv publications published in 2021 containing at least one GHP URI. Of all GHP URIs, 93.26% were GitHub URIs. Additionally, 33% of publications that contain a GitHub URI contain more than one GitHub URI which indicates a greater importance placed on the holdings of GitHub by authors.

Next, we analyzed the archival of the extracted GHP URIs and determined if they were rotten or active. We found that overall the Web archives had archived a larger portion of GHP URIs (79.15%) than Software Heritage (67.52%). However, Software Heritage captured GHP URIs that were not archived by Web archived. We found that 2,762 repository-level GHP URIs were rotten and 33.35% of those URIs are unrecoverable. Bitbucket had the highest percentage of rotten GHP URIs.

Scholars host software products in a variety of hosting platforms outside of the four GHPs that we previously studied. In order to identify other hosting platforms used by scholars, we used a machine learning classifier to identify OADS URIs outside of GitHub, GitLab, Bitbucket, and SourceForge. We identified almost 50,000 unique hostnames with 49,392 hostnames included in 0 to 50 non-GHP OADS URIs. Non-GHP OADS URIs were more prevalent than GHP URIs from 2007 to 2020. In 2020 and 2021, the number of non-GHP OADS URIs and GHP URIs were similar. Between the prevalence of non-GHP OADS URIs and the large number of unique hostnames, it is clear that a machine learning classifier is valuable tool for identifying OADS URIs for archival as manually enumerating hosting platforms would be insufficient.

Finally, we studied the relationship between popularity and archival for GitHub URIs. We found that there is a statistically significant relationship between the number of forks, subscribers, and stargazers and archival in both Web archives and Software Heritage. Less popular GitHub repositories are at an increased risk of becoming unrecoverable and, therefore, are at increased need of being archived.

Curators and archivists could use the extraction methods introduced in this work to identify potential software of interest for their collections. Using different methods, these URIs can then be used to seed the archiving process. For instance, these URIs could be used with the Memento Tracer framework[1] proposed by Klein et al. [55], which aims to strike a balance between scalability and quality for archiving scholarly code with its scholarly ephemera at scale. Memento Tracer allows users to create a heuristic called a trace, which can be used for for a class of Web publications. In testing the Memento Tracer framework, Klein et al. [55] was able to capture 100% of the expected URIs for 92.83% of the GitHub repositories in a given dataset. In future work, URIs derived using methods proposed in this study should be used to test the effectiveness of different archiving approaches at scale.

Our study focused on the arXiv and PMC corpora. Future work should investigate these trends across other corpora and other disciplines. Additionally, we looked at the presence of an archived copy of the repository, but did not investigate the quality of the copy. In some instances, the memento or capture may simply reflect that the URI is no longer available which would not be beneficial for reproducibility. In other cases, the capture may be incomplete in a way that negatively impacts reproducibility [56]. Future work should investigate the quality of the mementos and captures that are currently available. Additionally, future work should analyze the prevalence of content drift in GHPs compared to URIs to the Web at large.

Ideally, the archival of URIs to software products included in scholarly publications would be required by institutions and publishers to expand upon current open-access requirements. Future work should create a workflow that extracts software URIs and archives the software product and any surrounding ephemera upon submission to guarantee the long-term preservation of the software products used and created by authors.

In conclusion, software products hosted on the live Web are at risk of reference rot. Scholars and authors have a responsibility to archive the software products that they use and create to guarantee long-term reproducibility that benefits the scientific community. While there are current efforts to archive software, we have shown a need for increased emphasis on software preservation and the preservation of the ephemera that surrounds it.

[1]`http://tracer.mementoweb.org`

REFERENCES

[1] Committee on Reproducibility and Replicability in Science *et al. Reproducibility and Replicability in Science* (National Academies Press, 2019).

[2] Radford, A., Metz, L. & Chintala, S. Unsupervised representation learning with Deep Convolutional Generative Adversarial Networks. Tech. Rep. arXiv:1511.06434, arXiv (2016). URL `http://arxiv.org/abs/1511.06434`.

[3] Tabassi, E., Chugh, T., Deb, D. & Jain, A. K. Altered fingerprints: Detection and localization. Tech. Rep. arXiv:2106.02188, arXiv (2018). URL `http://arxiv.org/abs/1805.00911`.

[4] Lane, T. J., Schwantes, C. R., Beauchamp, K. A. & Pande, V. S. Efficient inference of protein structural ensembles. Tech. Rep. arXiv:1408:0255, arXiv (2014). URL `http://arxiv.org/abs/1408.0255`.

[5] Chacon, S. & Straub, B. *Pro Git* (Apress, 2014), second edn.

[6] Hickey, J. The Wayback Machine: Fighting Digital Extinction in New Ways (2019). URL `https://blog.archive.org/2019/10/18/the-wayback-machine-fighting-digital-extinction-in-new-ways/`.

[7] Reich, V. & Rosenthal, D. S. H. LOCKSS: A permanent web publishing and access system. *D-Lib Magazine* **7** (2001).

[8] Reich, V. CLOCKSS—It takes a community. *The Serials Librarian* **54**, 135–139 (2008).

[9] Fenton, E. G. An Overview of Portico: An Electronic Archiving Service. *Serials Review* **32**, 81–86 (2006).

[10] Bailey, J., Grotke, A., McCain, E., Moffatt, C. & Taylor, N. Web Archiving in the United States: A 2016 Survey. `http://ndsa.org/documents/WebArchivingintheUnitedStates_A2016Survey.pdf` (2017).

[11] Costa, M., Gomes, D. & Silva, M. J. The evolution of web archiving. *International Journal on Digital Libraries* **18**, 191–205 (2017).

[12] Gomes, D., Miranda, J. & Costa, M. A survey on web archiving initiatives. 408—420 (2011).

[13] Van de Sompel, H., Nelson, M. L. & Sanderson, R. HTTP Framework for Time-Based Access to Resource States - Memento - RFC 7089. `http://tools.ietf.org/html/rfc7089` (2013).

[14] Fielding, R. T. & Reschke, J. Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. RFC 7231 (2014).

[15] Alam, S. & Nelson, M. L. MemGator - a portable concurrent Memento aggregator. In *Proceedings of the 16th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL)*, 243–244 (2016).

[16] Software Heritage. Mission. `https://www.softwareheritage.org/mission/` (2023).

[17] Di Cosmo, R. & Zacchiroli, S. Software Heritage: Why and how to preserve software source code. In *Proceedings of the 14th International Conference on Digital Preservation (iPRES)*, 1–10 (2017). URL `https://hal.archives-ouvertes.fr/hal-01590958`.

[18] Software Heritage. Getting started with the Software Heritage API. `https://docs.softwareheritage.org/devel/getting-started/api.html` (2023).

[19] Kreymer, I. A New Phase for Webrecorder Project, Conifer and ReplayWeb.page. `https://webrecorder.net/2020/06/11/webrecorder-conifer-and-replayweb-page.html` (2020).

[20] Klein, M. *et al.* Scholarly context not found: One in five articles suffers from reference rot. *PLoS ONE* **9**, 1–39 (2014).

[21] Van de Sompel, H., Klein, M. & Shankar, H. Towards robust hyperlinks for web-based scholarly communication. In *Proceedings of the International Conference on Intelligent Computer Mathematics*, 12–25 (Springer International Publishing, 2014).

[22] Jones, S. M. *et al.* Scholarly context adrift: Three out of four URI references lead to changed content. *PLoS ONE* **11**, 1–32 (2016).

[23] Zittrain, J., Bowers, J. & Stanton, C. The paper of record meets an ephemeral web: An examination of linkrot and content drift within The New York Times. Tech. Rep., Social Science Research Network (2021).

[24] Agata, T., Miyata, Y., Ishita, E., Ikeuchi, A. & Ueda, S. Life span of web pages: A survey of 10 million pages collected in 2001. In *IEEE/ACM Joint Conference on Digital Libraries*, 463–464 (IEEE, 2014).

[25] Teixeira Da Silva, J. A. & Nazarovets, M. Archiving website-based references in academic papers: Problems caused by reference rot, potential solutions and limitations. *Learned Publishing* **36**, 477–487 (2023).

[26] Wattanakriengkrai, S. *et al.* GitHub repositories with links to academic papers: Public access, traceability, and evolution. *Journal of Systems and Software* **183** (2022).

[27] Färber, M. Analyzing the GitHub repositories of research papers. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*, 491–492 (ACM, 2020).

[28] Hasselbring, W., Carr, L., Hettrick, S., Packer, H. & Tiropanis, T. FAIR and open computer science research software. Tech. Rep. arXiv:1908.05986, arXiv (2019).

[29] Bhattarai, P., Ghassemi, M. & Alhanai, T. Open-source code repository attributes predict impact of computer science research. In *Proceedings of the 22nd ACM/IEEE Joint Conference on Digital Libraries*, 1–7 (2022).

[30] Pietri, A., Spinellis, D. & Zacchiroli, S. The Software Heritage graph dataset: Large-scale analysis of public software development history. In *Proceedings of the 17th International Conference on Mining Software Repositories*, 1–5 (2020).

[31] Bhattacharjee, A. *et al.* An Exploratory Study to Find Motives Behind Cross-platform Forks from Software Heritage Dataset. In *Proceedings of the 17th International Conference on Mining Software Repositories*, 11–15 (2020).

[32] Peters, I., Kraker, P., Lex, E., Gumpenberger, C. & Gorraiz, J. I. Zenodo in the Spotlight of Traditional and New Metrics. *Frontiers in Research Metrics and Analytics* **2**, 13 (2017).

[33] Foster, E. D. & Deardorff, A. Open Science Framework (OSF). *Journal of the Medical Library Association* **105** (2017).

[34] Milliken, G. Archiving the scholarly Git experience: An environmental scan. Tech. Rep., Open Science Framework (2021). URL `https://osf.io/ku24q/`.

[35] Salsabil, L. *et al.* A study of computational reproducibility using URLs linking to open access datasets and software. In *Companion Proceedings of the Web Conference 2022*, 784–788 (2022).

[36] Escamilla, E. *et al.* The rise of GitHub in scholarly publications. In *Proceedings of the International Conference on Theory and Practice of Digital Libraries (TPDL)*, 187–200 (2022).

[37] Fromme, A. arXiv hits 2M submissions (2022). URL `https://news.cornell.edu/stories/2022/01/arxiv-hits-2m-submissions`.

[38] Roberts, R. J. Pubmed central: The genbank of the published literature. *Proceedings of the National Academy of Sciences* **98**, 381–382 (2001).

[39] Hendricks, G., Tkaczyk, D., Lin, J. & Feeney, P. Crossref: The sustainable source of community-owned scholarly metadata. *Quantitative Science Studies* **1**, 414–427 (2020).

[40] Escamilla, E. Extract-URLs. URL `https://github.com/oduwsdl/Extract-URLs`.

[41] GitHub Docs. Managing a custom domain for your GitHub Pages site (2022). URL `https://docs.github.com/en/pages/configuring-a-custom-domain-for-your-github-pages-site/managing-a-custom-domain-for-your-github-pages-site`.

[42] Dhole, K. D. *et al.* NL-Augmenter: A framework for task-sensitive natural language augmentation. Tech. Rep. arXiv:2112.02721, arXiv (2021).

[43] Agol, E., Hernandez, D. M. & Langford, Z. A differentiable N-body code for transit timing and dynamical modeling. I. Algorithm and derivatives. Tech. Rep. arXiv:2106.02188, arXiv (2021).

[44] Truyen, E., Van Landuyt, D., Preuveneers, D., Lagaisse, B. & Joosen, W. A comprehensive feature comparison study of open-source container orchestration frameworks. Tech. Rep. arXiv:2002.02806, arXiv (2021).

[45] Kuo, T., Zavaleta Rojas, H. & Ohno-Machado, L. Comparison of blockchain platforms: a systematic review and healthcare examples. *Journal of the American Medical Informatics Association* **26**, 462–478 (2019).

[46] Kayani, M., Huang, W., Feng, R. & Chen, L. Genome-resolved metagenomics using environmental and clinical samples. *Briefings in Bioinformatics* **22** (2021).

[47] Chen, L. *et al.* The bioinformatics toolbox for circRNA discovery and analysis. *Briefings in Bioinformatics* **22**, 1706–1728 (2021).

[48] Yang, C. *et al.* A review of computational tools for generating metagenome-assembled genomes from metagenomic sequencing data. *Computational and Structural Biotechnology Journal* **19**, 6301–6314 (2021).

[49] McCown, F. & Nelson, M. L. A framework for describing web repositories. In *Proceedings of the 9th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL)*, 341–344 (2009).

[50] Bibliotheca Alexandrina. Internet Archive. `https://www.bibalex.org/en/project/details?documentid=283` (2023).

[51] Di Cosmo, R. Software Heritage is now open, please come in! (2016). URL `https://www.softwareheritage.org/2016/06/30/unveiling/`.

[52] Escamilla, E., Salsabil, L., Wu, J., Weigle, M. C. & Nelson, M. L. It's not just GitHub: Identifying data and software sources included in publications. In *Proceedings of the International Conference on Theory and Practice of Digital Libraries (TPDL)[Accepted for publication]* (2023).

[53] Mann, H. B. & Whitney, D. R. On a test of whether one of two random variables is stochastically larger than the other. *The Annals of Mathematical Statistics* **18**, 50–60 (1947).

[54] Cliff, N. Dominance statistics: Ordinal analyses to answer ordinal questions. *Psychological Bulletin* **114**, 494–509 (1993).

[55] Klein, M., Shankar, H., Balakireva, L. & Van de Sompel, H. The Memento Tracer Framework: Balancing quality and scalability for web archiving. In *Digital Libraries for Open Knowledge*, vol. 11799, 163–176 (Springer International Publishing, 2019).

[56] Brunelle, J. F., Kelly, M., SalahEldeen, H., Weigle, M. C. & Nelson, M. L. Not all mementos are created equal: Measuring the impact of missing resources. *International Journal on Digital Libraries* **16**, 283–301 (2015).

APPENDIX A

MEMGATOR API REQUEST AND RESPONSE

The `curl` request we used to get a JSON-formatted TimeMap from MemGator for `https://github.com/000justin000/gnn-residual-correlation`.

```
curl -s https://memgator.cs.odu.edu/timemap/json/https://github.com/000justin000/gnn-residual-correlation
```

The JSON response following the above request.

```
{
  "original_uri": "https://github.com/000justin000/gnn-residual-correlation",
  "self": "http://localhost:1208/timemap/json/https://github.com/000justin000/gnn-residual-correlation",
  "mementos": {
    "list": [
      {
        "datetime": "2020-06-19T15:23:57Z",
        "uri": "https://web.archive.org/web/20200619152357/https://github.com/000Justin000/gnn-residual-correlation"
      },
      {
        "datetime": "2020-10-25T22:28:50Z",
        "uri": "https://web.archive.org/web/20201025222850/https://github.com/000Justin000/gnn-residual-correlation"
      },
      {
        "datetime": "2022-04-25T14:02:36Z",
        "uri": "https://web.archive.org/web/20220425140236/https://github.com/000Justin000/gnn-residual-correlation"
      },
      {
        "datetime": "2022-10-14T10:40:43Z",
        "uri": "https://web.archive.org/web/20221014104043/http://github.com/000Justin000/gnn-residual-correlation"
      },
      {
        "datetime": "2023-03-21T00:26:35Z",
        "uri": "https://web.archive.org/web/20230321002635/https://github.com/000Justin000/gnn-residual-correlation"
      }
    ],
    "first": {
      "datetime": "2020-06-19T15:23:57Z",
      "uri": "https://web.archive.org/web/20200619152357/https://github.com/000Justin000/gnn-residual-correlation"
    },
    "last": {
      "datetime": "2023-03-21T00:26:35Z",
      "uri": "https://web.archive.org/web/20230321002635/https://github.com/000Justin000/gnn-residual-correlation"
    }
  },
  "timemap_uri": {
    "link_format": "http://localhost:1208/timemap/link/https://github.com/000justin000/gnn-residual-correlation",
    "json_format": "http://localhost:1208/timemap/json/https://github.com/000justin000/gnn-residual-correlation",
    "cdxj_format": "http://localhost:1208/timemap/cdxj/https://github.com/000justin000/gnn-residual-correlation"
  },
  "timegate_uri": "http://localhost:1208/timegate/https://github.com/000justin000/gnn-residual-correlation"
}
```

APPENDIX B

SOFTWARE HERITAGE API: REPOSITORY-LEVEL GHP URI

Using the Software Heritage API to request `https://github.com/aliasrobotics/RVD/`.

```
curl -H "Authorization: Bearer ${TOKEN}" -is https://archive.softwareher
↪  itage.org/api/1/origin/https://github.com/aliasrobotics/RVD/visits/
```

The response from the Software Heritage API to the above request.

```
HTTP/1.1 200 OK
Server: gunicorn/19.9.0
Content-Type: application/json
Vary: Accept,Cookie,Accept-Encoding
Allow: OPTIONS, HEAD, GET, OPTIONS
X-RateLimit-Limit: 1200
X-RateLimit-Remaining: 1197
X-RateLimit-Reset: 1675580195
X-Frame-Options: SAMEORIGIN
Via: 1.1 archive.softwareheritage.org
X-Varnish: 14266018
Age: 0
Via: 1.1 varnish (Varnish/6.1)
Strict-Transport-Security: max-age=15768000;
Accept-Ranges: bytes
Content-Length: 3844
Connection: keep-alive

[
  {
    "origin": "https://github.com/aliasrobotics/RVD",
    "visit": 9,
    "date": "2023-05-27T19:33:45.719423+00:00",
    "status": "full",
    "snapshot": "78fb33263eaa9e406925025b9243512d3407bac3",
    "type": "git",
    "metadata": {},
    "origin_visit_url": "https://archive.softwareheritage.org/api/1/origin/https://github.com/aliasrobotics/RVD/visit/9/",
    "snapshot_url": "https://archive.softwareheritage.org/api/1/snapshot/78fb33263eaa9e406925025b9243512d3407bac3/"
  },
  {
    "origin": "https://github.com/aliasrobotics/RVD",
    "visit": 8,
    "date": "2023-03-07T03:39:33.649247+00:00",
    "status": "full",
    "snapshot": "1b652bd11964ec9a4dc99424015340b41a72afe0",
    "type": "git",
    "metadata": {},
    "origin_visit_url": "https://archive.softwareheritage.org/api/1/origin/https://github.com/aliasrobotics/RVD/visit/8/",
    "snapshot_url": "https://archive.softwareheritage.org/api/1/snapshot/1b652bd11964ec9a4dc99424015340b41a72afe0/"
  },
  {
    "origin": "https://github.com/aliasrobotics/RVD",
    "visit": 7,
    "date": "2022-10-08T07:25:50.297834+00:00",
    "status": "full",
    "snapshot": "36bf33bcf64e080e2ae02ce988d694ae067efdd5",
    "type": "git",
    "metadata": {},
    "origin_visit_url": "https://archive.softwareheritage.org/api/1/origin/https://github.com/aliasrobotics/RVD/visit/7/",
    "snapshot_url": "https://archive.softwareheritage.org/api/1/snapshot/36bf33bcf64e080e2ae02ce988d694ae067efdd5/"
  },
  {
    "origin": "https://github.com/aliasrobotics/RVD",
    "visit": 6,
    "date": "2022-06-24T11:01:23.566922+00:00",
    "status": "full",
    "snapshot": "8b2f151a0ed288c666bd31e24a1d86e865a8f552",
    "type": "git",
    "metadata": {},
    "origin_visit_url": "https://archive.softwareheritage.org/api/1/origin/https://github.com/aliasrobotics/RVD/visit/6/",
    "snapshot_url": "https://archive.softwareheritage.org/api/1/snapshot/8b2f151a0ed288c666bd31e24a1d86e865a8f552/"
  },
  {
    "origin": "https://github.com/aliasrobotics/RVD",
    "visit": 5,
    "date": "2021-12-25T22:25:35.488503+00:00",
    "status": "full",
    "snapshot": "8b3aee3d9d55fc3d3fcbe906f329bc4c867ccff3",
    "type": "git",
    "metadata": {},
    "origin_visit_url": "https://archive.softwareheritage.org/api/1/origin/https://github.com/aliasrobotics/RVD/visit/5/",
    "snapshot_url": "https://archive.softwareheritage.org/api/1/snapshot/8b3aee3d9d55fc3d3fcbe906f329bc4c867ccff3/"
  },
  {
    "origin": "https://github.com/aliasrobotics/RVD",
    "visit": 4,
    "date": "2021-12-15T12:03:09.854107+00:00",
    "status": "full",
    "snapshot": "a73d4e6813be4c5c23ac57bc05acada0a37fd3c8",
    "type": "git",
    "metadata": {},
    "origin_visit_url": "https://archive.softwareheritage.org/api/1/origin/https://github.com/aliasrobotics/RVD/visit/4/",
    "snapshot_url": "https://archive.softwareheritage.org/api/1/snapshot/a73d4e6813be4c5c23ac57bc05acada0a37fd3c8/"
  },
  {
    "origin": "https://github.com/aliasrobotics/RVD",
    "visit": 3,
    "date": "2021-09-24T18:13:58.729338+00:00",
    "status": "full",
    "snapshot": "70664dd4a3d1a2bf0e9831ae345b96c545baac42",
    "type": "git",
    "metadata": {},
    "origin_visit_url": "https://archive.softwareheritage.org/api/1/origin/https://github.com/aliasrobotics/RVD/visit/3/",
    "snapshot_url": "https://archive.softwareheritage.org/api/1/snapshot/70664dd4a3d1a2bf0e9831ae345b96c545baac42/"
  },
  {
    "origin": "https://github.com/aliasrobotics/RVD",
    "visit": 2,
    "date": "2021-06-06T22:52:40.299241+00:00",
    "status": "full",
    "snapshot": "73b474e1a2cd9b17162f732edf5fa112cc921cbe",
    "type": "git",
    "metadata": {},
    "origin_visit_url": "https://archive.softwareheritage.org/api/1/origin/https://github.com/aliasrobotics/RVD/visit/2/",
    "snapshot_url": "https://archive.softwareheritage.org/api/1/snapshot/73b474e1a2cd9b17162f732edf5fa112cc921cbe/"
  },
  {
    "origin": "https://github.com/aliasrobotics/RVD",
    "visit": 1,
    "date": "2020-03-23T00:30:31.956121+00:00",
    "status": "full",
    "snapshot": "2f56fae02bcc42aa0b156e57555b3c71c27488fe",
    "type": "git",
    "metadata": {},
    "origin_visit_url": "https://archive.softwareheritage.org/api/1/origin/https://github.com/aliasrobotics/RVD/visit/1/",
    "snapshot_url": "https://archive.softwareheritage.org/api/1/snapshot/2f56fae02bcc42aa0b156e57555b3c71c27488fe/"
  }
```

```
]
```

APPENDIX C

SOFTWARE HERITAGE API: DEEP LINK GHP URI

Using the Software Heritage API to request https://github.com/aliasrobotics/RVD/blob/master/rvd_tools/database/schema.py.

```
curl -H "Authorization: Bearer ${TOKEN}" -is
 ↪ https://archive.softwareheritage.org/api/1/origin/https://github.com⌋
 ↪ /aliasrobotics/RVD/blob/master/rvd_tools/database/schema.py/visits/
```

The response from the Software Heritage API to the above request.

```
HTTP/1.1 404 Not Found
Content-Type: application/json
Vary: Accept,Cookie
Allow: OPTIONS, HEAD, GET, OPTIONS
X-RateLimit-Limit: 1200
X-RateLimit-Remaining: 1199
X-RateLimit-Reset: 1675579730
X-Frame-Options: SAMEORIGIN
Content-Length: 143
Via: 1.1 archive.softwareheritage.org
X-Varnish: 12635913
Age: 0
Via: 1.1 varnish (Varnish/6.1)
Strict-Transport-Security: max-age=15768000;
Connection: keep-alive

{"exception":"NotFoundExc","reason":"Origin
with url https://github.com/aliasrobotics/
RVD/blob/master/rvd_tools/database/schema.py
not found!"}
```

APPENDIX D

REGULAR EXPRESSIONS TO EXTRACT SOURCEFORGE URIS

The regular expressions used to capture SourceForge URIs.

```
sf_sitemap = ['sourceforge.net/create$', 'sourceforge.net/create/.*',
 ↪  'sourceforge.net/about$', 'sourceforge.net/about/.*',
 ↪  'sourceforge.net/top$', 'sourceforge.net/top/.*',
 ↪  'sourceforge.net/user/newsletters$',
 ↪  'sourceforge.net/user/newsletters/.*',
 ↪  'sourceforge.net/user/registration$',
 ↪  'sourceforge.net/user/registation/.*',
 ↪  'sourceforge.net/user/registration_business$',
 ↪  'sourceforge.net/user/registration_business/.*',
 ↪  'sourceforge.net/software/vendors$',
 ↪  'sourceforge.net/software/vendors/.*',
 ↪  'sourceforge.net/software/reviews$',
 ↪  'sourceforge.net/software/reviews/.*', 'sourceforge.net/p/forge$',
 ↪  'sourceforge.net/p/forge/.*', 'sourceforge.net/p/add_project$',
 ↪  'sourceforge.net/p/add_project/.*', 'sourceforge.net/auth$',
 ↪  'sourceforge.net/auth/.*', 'sourceforge.net/directory$',
 ↪  'sourceforge.net/directory/.*', 'sourceforge.net/software/?',
 ↪  'sourceforge.net/blog$', 'sourceforge.net/blog/.*',
 ↪  'sourceforge.net/about$', 'sourceforge.net/about/.*']

sf = re.search(r"(sourceforge.net)", url)
if sf is not None:
  if re.search(r"(?=("+'|'.join(sf_sitemap)+r"))", url) is not None:
    # URL included in the SourceForge sitemap
  else:
    # SourceForge repository URL
```

APPENDIX E

REGULAR EXPRESSIONS TO EXTRACT GITLAB URIS

The regular expressions used to capture GitLab URIs.

```
gl_sitemap = ['gitlab.com/users/sign_in$',
 ↪  'gitlab.com/users/sign_in/.*', 'gitlab.com/users/sign_up$',
 ↪  'gitlab.com/users/sign_up/.*', 'gitlab.com/explore$',
 ↪  'gitlab.com/explore/.*', 'gitlab.com/help$', 'gitlab.com/help/.*']

gl = re.search(r"(gitlab.com|gitlab.io)", url)
if gl is not None:
    if re.match(r'io,gitlab', s):
      # URL to GitLab.io
    elif not re.match(r"^(https?:\/\/w{0,3}.?gitlab.com\/.+)", url) or
     ↪  re.search(r"(?=("+'|'.join(gl_sitemap)+r"))", url) is not None:
      # URL included in GitLab sitemap and old GitLab pages URL formats
    else:
      # GitLab repository URL
```

APPENDIX F

REGULAR EXPRESSIONS TO EXTRACT BITBUCKET URIS

The regular expressions used to capture Bitbucket URIs.

```
bb_sitemap = ['bitbucket.org/product$', 'bitbucket.org/product/.*', 'bitbucket.org/blog$', 'bitbucket.org/blog/.*']

bb = re.search(r"(bitbucket.org|bitbucket.io)", url)
if bb is not None:
    # is it a link to a repo?
    if not re.match(r"^https?:\/\/(w{0,3}.?bitbucket.org\/.+|.*@bitbucket.org\/.+)", url) or re.search(r"(?=("+'|'.join(bb_sitemap)+r"))", url) is not None:
        # is it a link to Bitbucket pages?
        if re.match(r"^https?:\/\/((?!www).*.?bitbucket.org|.*bitbucket.io)", url):
            # URL to Bitbucket pages
        else:
            # Non-repository Bitbucket URL
    else:
      # Bitbucket repository URL
```

APPENDIX G

REGULAR EXPRESSIONS TO EXTRACT GITHUB URIS

The regular expressions used to capture GitHub URIs.

```
gh_sitemap = ['github.com/join\?', 'github.com/login',
 ↪  'github.com/pricing$', 'github.com/pricing/.*'
 ↪  'github.com/git-guides$', 'github.com/git-guides/.*',
 ↪  'github.com/team$', 'github.com/team/.*',
 ↪  'github.com/marketplace$', 'github.com/marketplace/.*',
 ↪  'github.com/enterprise$', 'github.com/enterprise/.*',
 ↪  'github.com/features$', 'github.com/features/.*',
 ↪  'github.com/readme$', 'github.com/readme/.*', 'github.com/about$',
 ↪  'github.com/about/.*', 'github.com/learn$', 'github.com/learn/.*']

gh = re.search(r"(github.com|github.io)", url)
if gh is not None:
  if re.match(r'com,github,gist', url):
    # URL to Gist
  elif re.match(r'org,archive,web\)\/save\/', url):
    # URL to Internet Archive
  elif re.match(r'org,archive,web\)\/web\/', url):
    # URL to Internet Archive
  elif re.match(r'io,github', url):
    # URL to GitHub.io
  elif not re.match(r"^(https?:\/\/w{0,3}.?github.com\/.+)", url) or
   ↪  re.search(r"(?=("+'|'.join(gh_sitemap)+r"))", url) is not None:
    # URL included in GitHub sitemap and old GitHub pages URL formats
  else:
    # GitHub repository URL
```

APPENDIX H

GITHUB API REQUEST AND RESPONSE

The `curl` request we used to get a JSON-formatted response from the GitHub API for `https://github.com/tensorflow/tensorflow`.

```
curl -L -H "Authorization: Bearer ${TOKEN}" -H "X-GitHub-Api-Version: 2022-11-28" -H "Accept: application/vnd.github+json" -is "https://api.github.com/repos/tensorflow/tensorflow"
```

The JSON response following the above request.

```
  {
  "id": 45717250,
  "node_id": "MDEwOlJlcG9zaXRvcnk0NTcxNzI1MA==",
  "name": "tensorflow",
  "full_name": "tensorflow/tensorflow",
  "private": false,
  "owner": {
    "login": "tensorflow",
    "id": 15658638,
    "node_id": "MDEyOk9yZ2FuaXphdGlvbjE1NjU4NjM4",
    "avatar_url": "https://avatars.githubusercontent.com/u/15658638?v=4",
    "gravatar_id": "",
    "url": "https://api.github.com/users/tensorflow",
    "html_url": "https://github.com/tensorflow",
    "followers_url": "https://api.github.com/users/tensorflow/followers",
    "following_url": "https://api.github.com/users/tensorflow/following{/other_user}",
    "gists_url": "https://api.github.com/users/tensorflow/gists{/gist_id}",
    "starred_url": "https://api.github.com/users/tensorflow/starred{/owner}{/repo}",
    "subscriptions_url": "https://api.github.com/users/tensorflow/subscriptions",
    "organizations_url": "https://api.github.com/users/tensorflow/orgs",
    "repos_url": "https://api.github.com/users/tensorflow/repos",
    "events_url": "https://api.github.com/users/tensorflow/events{/privacy}",
    "received_events_url": "https://api.github.com/users/tensorflow/received_events",
    "type": "Organization",
    "site_admin": false
  },
  "html_url": "https://github.com/tensorflow/tensorflow",
  "description": "An Open Source Machine Learning Framework for Everyone",
  "fork": false,
  "url": "https://api.github.com/repos/tensorflow/tensorflow",
  "forks_url": "https://api.github.com/repos/tensorflow/tensorflow/forks",
  "keys_url": "https://api.github.com/repos/tensorflow/tensorflow/keys{/key_id}",
  "collaborators_url": "https://api.github.com/repos/tensorflow/tensorflow/collaborators{/collaborator}",
  "teams_url": "https://api.github.com/repos/tensorflow/tensorflow/teams",
  "hooks_url": "https://api.github.com/repos/tensorflow/tensorflow/hooks",
  "issue_events_url": "https://api.github.com/repos/tensorflow/tensorflow/issues/events{/number}",
  "events_url": "https://api.github.com/repos/tensorflow/tensorflow/events",
  "assignees_url": "https://api.github.com/repos/tensorflow/tensorflow/assignees{/user}",
  "branches_url": "https://api.github.com/repos/tensorflow/tensorflow/branches{/branch}",
  "tags_url": "https://api.github.com/repos/tensorflow/tensorflow/tags",
  "blobs_url": "https://api.github.com/repos/tensorflow/tensorflow/git/blobs{/sha}",
  "git_tags_url": "https://api.github.com/repos/tensorflow/tensorflow/git/tags{/sha}",
  "git_refs_url": "https://api.github.com/repos/tensorflow/tensorflow/git/refs{/sha}",
  "trees_url": "https://api.github.com/repos/tensorflow/tensorflow/git/trees{/sha}",
  "statuses_url": "https://api.github.com/repos/tensorflow/tensorflow/statuses/{sha}",
  "languages_url": "https://api.github.com/repos/tensorflow/tensorflow/languages",
  "stargazers_url": "https://api.github.com/repos/tensorflow/tensorflow/stargazers",
  "contributors_url": "https://api.github.com/repos/tensorflow/tensorflow/contributors",
  "subscribers_url": "https://api.github.com/repos/tensorflow/tensorflow/subscribers",
  "subscription_url": "https://api.github.com/repos/tensorflow/tensorflow/subscription",
  "commits_url": "https://api.github.com/repos/tensorflow/tensorflow/commits{/sha}",
  "git_commits_url": "https://api.github.com/repos/tensorflow/tensorflow/git/commits{/sha}",
  "comments_url": "https://api.github.com/repos/tensorflow/tensorflow/comments{/number}",
  "issue_comment_url": "https://api.github.com/repos/tensorflow/tensorflow/issues/comments{/number}",
  "contents_url": "https://api.github.com/repos/tensorflow/tensorflow/contents/{+path}",
  "compare_url": "https://api.github.com/repos/tensorflow/tensorflow/compare/{base}...{head}",
  "merges_url": "https://api.github.com/repos/tensorflow/tensorflow/merges",
  "archive_url": "https://api.github.com/repos/tensorflow/tensorflow/{archive_format}{/ref}",
  "downloads_url": "https://api.github.com/repos/tensorflow/tensorflow/downloads",
  "issues_url": "https://api.github.com/repos/tensorflow/tensorflow/issues{/number}",
  "pulls_url": "https://api.github.com/repos/tensorflow/tensorflow/pulls{/number}",
  "milestones_url": "https://api.github.com/repos/tensorflow/tensorflow/milestones{/number}",
  "notifications_url": "https://api.github.com/repos/tensorflow/tensorflow/notifications{?since,all,participating}",
  "labels_url": "https://api.github.com/repos/tensorflow/tensorflow/labels{/name}",
  "releases_url": "https://api.github.com/repos/tensorflow/tensorflow/releases{/id}",
  "deployments_url": "https://api.github.com/repos/tensorflow/tensorflow/deployments",
  "created_at": "2015-11-07T01:19:20Z",
  "updated_at": "2023-02-11T18:08:11Z",
  "pushed_at": "2023-02-11T17:55:50Z",
  "git_url": "git://github.com/tensorflow/tensorflow.git",
  "ssh_url": "git@github.com:tensorflow/tensorflow.git",
  "clone_url": "https://github.com/tensorflow/tensorflow.git",
  "svn_url": "https://github.com/tensorflow/tensorflow",
  "homepage": "https://tensorflow.org",
  "size": 1047133,
  "stargazers_count": 171097,
  "watchers_count": 171097,
  "language": "C++",
  "has_issues": true,
  "has_projects": true,
  "has_downloads": true,
  "has_wiki": false,
  "has_pages": false,
  "has_discussions": false,
  "forks_count": 87766,
  "mirror_url": null,
  "archived": false,
  "disabled": false,
  "open_issues_count": 2305,
  "license": {
    "key": "apache-2.0",
    "name": "Apache License 2.0",
    "spdx_id": "Apache-2.0",
    "url": "https://api.github.com/licenses/apache-2.0",
    "node_id": "MDc6TGljZW5zZTI="
  },
  "allow_forking": true,
  "is_template": false,
  "web_commit_signoff_required": false,
  "topics": [
    "deep-learning",
    "deep-neural-networks",
    "distributed",
    "machine-learning",
    "ml",
    "neural-network",
    "python",
    "tensorflow"
  ],
  "visibility": "public",
  "forks": 87766,
  "open_issues": 2305,
  "watchers": 171097,
  "default_branch": "master",
  "permissions": {
    "admin": false,
    "maintain": false,
    "push": false,
    "triage": false,
    "pull": true
  },
  "temp_clone_token": "",
  "organization": {
    "login": "tensorflow",
    "id": 15658638,
    "node_id": "MDEyOk9yZ2FuaXphdGlvbjE1NjU4NjM4",
    "avatar_url": "https://avatars.githubusercontent.com/u/15658638?v=4",
    "gravatar_id": "",
    "url": "https://api.github.com/users/tensorflow",
    "html_url": "https://github.com/tensorflow",
    "followers_url": "https://api.github.com/users/tensorflow/followers",
    "following_url": "https://api.github.com/users/tensorflow/following{/other_user}",
    "gists_url": "https://api.github.com/users/tensorflow/gists{/gist_id}",
    "starred_url": "https://api.github.com/users/tensorflow/starred{/owner}{/repo}",
    "subscriptions_url": "https://api.github.com/users/tensorflow/subscriptions",
    "organizations_url": "https://api.github.com/users/tensorflow/orgs",
    "repos_url": "https://api.github.com/users/tensorflow/repos",
    "events_url": "https://api.github.com/users/tensorflow/events{/privacy}",
    "received_events_url": "https://api.github.com/users/tensorflow/received_events",
    "type": "Organization",
    "site_admin": false
  },
  "network_count": 87766,
  "subscribers_count": 7776
}
```

VITA

Emily Escamilla
Department of Computer Science
Old Dominion University
Norfolk, VA 23529

EDUCATION

2017-2021, B.S. in Computer Science, Liberty University, Lynchburg, VA.
2021-2023, M.S. in Computer Science, Old Dominion University, Norfolk, VA.

PROFESSIONAL EXPERIENCE

2021-Present, Graduate Research Assistant, Old Dominion University.
2017-2021, Software Developer Intern, MITRE Corporation, Hampton, VA.

CONFERENCE PRESENTATIONS

1. "The Rise of GitHub in Scholarly Publications" presented at the Theory and Practice of Digital Libraries (TPDL) 2022 in Padua, Italy.
2. "What if GitHub Disappeared Tomorrow?" presented at the International Internet Preservation Consortium (IIPC) 2023 in Hilversum, The Netherlands
3. "Institutional Web Archiving Initiatives to Support Digital Scholarship" presented at the International Internet Preservation Consortium (IIPC) 2023 in Hilversum, The Netherlands

PUBLICATIONS

A complete list is available at `https://elescamilla.github.io/publications/`