CODE CONTRIBUTION AND SCIENTIFIC AUTHORSHIP

Eva Maxfield Brown

University of Washington Information School

evamxb@uw.edu

Isaac Slaughter

University of Washington Information School

Shahan Ali Memon

University of Washington Information School

Nicholas Weber

University of Washington Information School

nmweber@uw.edu

February 27, 2025

ABSTRACT

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur eget porta erat. Morbi consectetur est vel gravida pretium. Suspendisse ut dui eu ante cursus gravida non sed sem. Nullam sapien tellus, commodo id velit id, eleifend volutpat quam. Phasellus mauris velit, dapibus finibus elementum vel, pulvinar non tellus. Nunc pellentesque pretium diam, quis maximus dolor faucibus id. Nunc convallis sodales ante, ut ullamcorper est egestas vitae. Nam sit amet enim ultrices, ultrices elit pulvinar, volutpat risus.

1 Introduction

Software has become integral to contemporary scientific research (Edwards et al. 2013; Mayernik et al. 2017). From analysis scripts to infrastructure for data collection (Hasselbring et al. 2024), specialized tools are now essential components of modern research practices. Today, research software serves multiple critical functions: enabling reproducible experiments (Krafczyk et al. 2019; Trisovic et al. 2021), providing methodological documentation (Ram 2013), and increasingly appearing alongside publications as a research artifact (Cao et al. 2023; Trujillo, Hébert-Dufresne, and Bagrow 2022). Software's expanding role has changed how science is conducted, with many fields now depending on specialized tools to advance knowledge.

Despite the growing dependence on software, a significant recognition gap exists between software development and traditional scientific outputs. Software contributors frequently find their work relegated to acknowledgments sections rather than warranting authorship (Philippe et al. 2019), creating a disconnect between the value software provides and the credit its developers receive. This misalignment has consequences for scientific careers, as the lack of formal credit can significantly impact promotion and advancement within research organizations (Carver et al. 2022; Biagioli and Galison 2014). The result is a system that increasingly relies on software while undervaluing those who create it, potentially discouraging talented developers from pursuing scientific careers and affecting the sustainability of research software development (Muna et al. 2016).

The scientific community has attempted to address this recognition gap through various initiatives. The Contributor Roles Taxonomy (CRediT) has expanded authorship criteria to include specialized contribution roles (Brand et al. 2015), while researchers have used similar taxonomies to investigate labor distribution within teams, including specialist and generalist roles (Larivière, Pontille, and Sugimoto 2020; Larivière et al. 2016; Sauermann and Haeussler 2017; Li, Zhang, and Larivière 2023; Lu et al. 2019). Despite these efforts, limitations persist. Current frameworks remain centered on traditional author lists, perpetuating historical biases and relying heavily on self-reporting without external verification (Haeussler and Sauermann 2013; Gøtzsche et al. 2007; Ni et al. 2021). The scientific community

continues to work on developing appropriate systems for software citation and credit attribution that adequately reflect these contributions (Merow et al. 2023; Westner et al. 2024; Katz et al. 2020).

The emergence of public code repositories alongside published research presents an opportunity to address this problem. Source code repositories typically employ version control systems that maintain records of who contributes what code and when, providing a window into the patterns of software development in scientific research. Using the transaction histories from source code repositories, we developed a predictive model that enables systematic matching between scientific article authors and source code developer accounts - creating a dataset that connects formal authorship with code contributions across a corpus of scientific work.

By applying our predictive model across 138596 paired research articles and repositories, we identify several patterns in the relationship between code contributions and scientific recognition. Our analysis reveals that nearly 30% of articles have non-author code-contributors - individuals who helped create the software but received no formal authorship credit. We find that code-contributing authors are associated with modest increases in article-level impact metrics (~5.2% increase in citations per code-contributing author), though these effects become statistically non-significant when controlling for domain, article type, and open access status. First authors emerge as significantly more likely to be code contributors compared to other positions across all conditions tested. We also document a negative relationship between coding frequency and scholarly impact: authors who contribute code more frequently show progressively lower h-indices compared to their non-coding peers, a pattern that persists when controlling for publication count, and author's most common author position, domain, and article type.

The primary contributions of this article are: (1) a predictive model for matching authors with developer account information that addresses challenges in identity resolution across platforms; (2) a dataset of linked authors and developers for 138596 article-repository pairs, providing a resource for analyzing scientific software development patterns; and (3) preliminary analyses that reveal insights into the software development dynamics of research teams, including patterns of recognition, impact, and career implications for code contributors. These contributions provide evidence for the ongoing discussions about scientific recognition systems and raise questions about aligning institutional incentives with the spectrum of contributions that drive modern scientific progress.

The remainder of this paper proceeds as follows. We first provide relevant background on scientific software development and recognition, introducing the specific hypotheses that guide our investigation. Next, we detail our data and methods, describing how we created a dataset of linked article-repository pairs, trained and evaluated our predictive model for entity matching, and applied our model across each article-repository pair. We then present our preliminary analysis of the data, focusing first on article/team-level dynamics before moving to individual-level patterns, formally accepting or rejecting each hypothesis based on our findings. We conclude with a discussion of the results, limitations of our work, and areas for future improvement.

2 Background

2.1 Research Team Composition and Scientific Impact

Scientific recognition systems have, except for foundational methodological shifts, favored experimental and theoretical contributions more than their methodological counterparts, with experimental and theoretical articles receiving higher citation rates (Aksnes 2006; Liu, Zhang, and Li 2023; Chen et al. 2024). However, the growing importance of computational methods in science has transformed how research is conducted across all types of scientific work. Modern scientific endeavors increasingly depend on sophisticated computational approaches, whether for processing large-scale experimental data, running complex simulations, or developing new methodological tools (Jin et al. 2015; Hampton et al. 2013; Edwards et al. 2013; Mayernik et al. 2017; Hasselbring et al. 2024).

The computational evolution in scientific practice intersects with established findings about team dynamics in both research and software development. Prior research has shown that larger and more diverse teams typically produce higher-impact scientific work (Franceschet and Costantini 2010; Larivière et al. 2014), while in software engineering, larger development teams tend to create more reliable software with fewer defects (Wyss, De Carli, and Davidson 2023). These parallel findings suggest that team size may be particularly important in scientific software development, where both technical robustness and scientific innovation are crucial.

The unique characteristics of scientific software development - including the implementation of novel algorithms, requirements for deep domain knowledge, and an increased emphasis on reproducibility (Muna et al. 2016; Howison and Herbsleb 2013) - make team composition especially relevant. Larger development teams may enhance scientific impact through multiple mechanisms: they can produce more robust and generalizable software tools for methodological contributions, while also enabling more sophisticated computational analyses and larger-scale data processing for experimental work. Given these patterns in team dynamics, software development practices, and the computational transformation of scientific work, we propose:

H1: The number of code contributors to a research article will be positively associated with the article's citation count.

2.2 Author Roles and Technical Contributions

Author positions in scientific publications signal specific roles and responsibilities, a relationship extensively studied through contribution role taxonomies like CRediT (Larivière et al. 2016). These studies reveal that first authors and corresponding authors, while occasionally the same individual (Chinchilla-Rodrguez et al. 2022), take on distinct sets of responsibilities. Analyses of contribution patterns consistently show that software development, data analysis, and visualization tasks typically fall to first authors (Larivière et al. 2016; Júnior et al. 2016; Larivière, Pontille, and Sugimoto 2020; Sauermann and Haeussler 2017). Meanwhile, corresponding authors, whether or not they are also first authors, often maintain responsibility for research artifacts' long-term sustainability and reuse, which we believe may include the maintenance and documentation of software tools.

Source code repository contribution records provide a unique method to verify these established patterns of contribution. Given prior findings about the distribution of technical responsibilities within research teams, we expect these repository records to reflect similar patterns of engagement with software development:

H2a: First authors have higher rates of code contribution compared to authors in other positions.

H2b: Corresponding authors have higher rates of code contribution compared to non-corresponding authors.

2.3 Code Contribution and Individual Scientific Impact

Despite the increasingly central role of software in science, researchers who develop scientific software face persistent challenges in receiving formal scientific recognition for their contributions. Prior work has shown that software developers in research settings are often relegated to acknowledgment sections rather than receiving authorship credit, even when their technical contributions are fundamental to the research (Carver et al. 2022; Philippe et al. 2019).

The challenge of recognition is compounded by inconsistent practices in software citation. While efforts have been made to standardize software citation, actual citation practices remain highly variable across fields and journals (Lamprecht et al. 2020; Katz et al. 2020; Smith, Katz, and Niemeyer 2016). This variability creates particular challenges for researchers who maintain and update existing software packages. While creating entirely new software may lead to dedicated publications and citations, the ongoing work of maintaining, debugging, and extending existing software - often crucial for scientific progress - typically generates less visible scientific credit (Howison and Herbsleb 2011, 2013).

These structural challenges in how software contributions are recognized and cited suggest a potential misalignment between technical contributions and traditional scientific impact metrics. When researchers dedicate significant time to software development and maintenance, their contributions may not be fully captured by conventional bibliometric measures, regardless of the software's importance to the field. Whether through attribution practices that favor acknowledgments over authorship, or citation patterns that undervalue maintenance work, multiple mechanisms could lead to lower traditional impact metrics for active code contributors. Based on these patterns in software recognition and citation, we hypothesize:

H3: The frequency of a researcher's code contributions will be negatively associated with their h-index.

3 Data and Methods

3.1 Linking Scientific Articles and Source Code Repositories

Modern scientific research increasingly requires the public sharing of research code, creating unique opportunities to study the relationship between scientific authorship and software development. Many journals and platforms now require or recommend code and data sharing (Stodden, Guo, and Ma 2013; Sharma et al. 2024), creating traceable links between publications and code that enable systematic study of both article-repository and author-developer relationships (Hata et al. 2021; Kelley and Garijo 2021; Stankovski and Garijo 2024; Milewicz, Pinto, and Rodeghero 2019).

Our data collection process leverages multiple sources of linked scientific articles and code repositories to ensure broad coverage of multiple different domains and article types. Our dataset combines article-source-code-repository pairs from:

- PLOS: Traditional research articles
- JOSS and SoftwareX: Specialized software-focused publications
- Papers with Code (ArXiv): preprints

To reduce the complexity of dataset processing and enrichment, we filter out any article-source-code-repository pairs which store code somewhere other than GitHub. While this decision prioritizes processing simplicity, we acknowledge that while GitHub is the predominate host of scientific software (Cao et al. 2023; Escamilla et al. 2022), it is also stored and shared on other platforms, which should be investigated as a part of future research (Trujillo, Hébert-Dufresne, and Bagrow 2022).

Through integration of multiple data sources, we extract detailed information about both the scientific and software development aspects of each project. We utilize the Semantic Scholar API for DOI resolution to ensure that we find the latest version of each article, which is particularly important for working with preprints as they may have been published in a journal since their inclusion in the Papers with Code dataset. We then utilize the OpenAlex API to gather publication metadata (open access status, domain, publication date), author details (name, author position, corresponding author status), and article- and individual-level metrics (citation counts, FWCI, h-index). The GitHub API provides similar information for source code repositories, including repository metadata (name, description, languages, creation date), contributor details (username, name, email), and repository-level metrics (star count, fork count, issue count).

Semantic Scholar DOI Resolution Statistics:

- Overall DOI resolution rate: 56.3%
- PLOS DOI resolution rate: 2.1%
- JOSS DOI resolution rate: 4.0%
- SoftwareX DOI resolution rate: 0.0%
- Papers with Code DOI resolution rate: 49.2%

Taken together, we form one of the largest and most comprehensively annotated collections of paired scientific articles and associated source-code repositories. In total, we collect and enrich data for 163292 article-repository pairs.

3.2 A Predictive Model for Matching Article Authors and Source Code Contributors

3.2.1 Annotated Dataset Creation

The development of an accurate author-developer matching model requires high-quality labeled training data that captures the complexity of real-world identity matching. Entity matching between authors and developers is non-trivial due to multiple forms of name variation and incomplete information. These variations can include differences in formatting (e.g., "J. Doe" vs "Jane Doe"), institutional versus personal email addresses, and incomplete or outdated information.

We developed an annotation process to create a robust training dataset while maximizing efficiency and accuracy. We focused our annotation efforts on JOSS articles to increase positive match density, as these software-focused publications typically have higher overlap between authors and developers. For each JOSS author, we generated three random pairings with developer accounts from the article's associated repository. From the full set of generated pairs, we randomly sampled 3,000 for annotation which two independent annotators then labeled as either a match or non-match. After completing all annotations, we systematically resolved any disagreements between the annotators through discussion and additional verification.

The resulting annotated dataset provides a comprehensive foundation for training our predictive model while highlighting common patterns in author-developer identity matching. After resolving all annotated pairs, our final dataset contains:

1. Match Distribution:
   - 451 (15.0%) positive matches
   - 2548 (85.0%) negative matches
2. Unique Individuals:
   - 2027 unique authors
   - 2733 unique developer accounts
3. Developer Profile Completeness:
   - 2191 (80.2%) accounts have associated names
   - 839 (30.7%) accounts have associated emails

3.2.2 Training and Evaluation

Our training and evaluation methodology begins with careful dataset preparation to prevent data leakage between training and test sets. To ensure complete separation of both authors and developers, we randomly selected 10% of

unique authors and 10% of unique developers, designating any pairs containing these selected entities for the test set. Due to the combinatorial nature of our author-developer pairs, this entity-based splitting strategy resulted in 2442 (81.4%) pairs for training and 557 (18.6%) pairs for testing.

For our predictive model, we evaluate three transformer-based architectures that have demonstrated strong performance in entity matching tasks: DeBERTa-v3-base (He, Gao, and Chen 2021; He et al. 2021), mBERT (bert-base-multilingual-cased) (Devlin et al. 2018), and DistilBERT (Sanh et al. 2019). While BERT-based architectures have been widely studied, they continue to achieve state-of-the-art results across various natural language processing tasks, particularly in scenarios requiring precise entity matching and relationship identification (Tran et al. 2024; Yu et al. 2024; Jeong and Kim 2022).

We conducted systematic evaluation of these base models across different combinations of developer-account features, ranging from using only the username to incorporating full profile information (username, display name, and email address). All models were fine-tuned using the Adam optimizer with a learning rate of 1e-05, batch sizes of 8 for both training and evaluation, and a linear learning rate scheduler. Given the size of our dataset and the binary nature of our classification task, models were trained for a single epoch to prevent overfitting.

Model performance was assessed using standard binary classification metrics, with particular emphasis on the F1 score for positive (matching) pairs due to the inherent class imbalance in author-developer matching. This evaluation framework allows us to directly compare model architectures and feature combinations while accounting for both precision and recall in identifying correct matches.

Our comprehensive model evaluation revealed that fine-tuning DeBERTa-v3-base (He, Gao, and Chen 2021) with developer username and display name as input features produces optimal performance for author-developer matching. This model configuration achieves a binary F1 score of 0.944, with accuracy of 0.984, precision of 0.938, and recall of 0.95. A complete comparison of model architectures and feature combinations can be found in Table 4.

Analysis of each model's performance revealed that including developer display names has the largest positive impact on model performance compared to username alone. While this effect might be partially attributed to the higher availability of display names in our dataset compared to email addresses, the performance improvement is notable. We additionally observe that mBERT's performance was comparable with DeBERTa's while additionally using the developer email address as an additional input feature, but selected the DeBERTa configuration for its relative simplicity and more recent and comprehensive pretraining. DeBERTa's consistent strong performance across various feature combinations, combined with its more extensive pretraining dataset, suggests better generalization potential for future applications.

To facilitate the reuse of our work, we have made our trained model and supporting code publicly available. Complete fine-tuning, evaluation, and inference code is available as the Python package: sci-soft-models, and the fine-tuned model has been deployed to HuggingFace (evamxb/dev-author-em-clf).

3.3 Linking Authors and GitHub Developer Accounts

Our trained entity-matching model enables comprehensive identification of author-developer relationships across all possible author and developer-account combinations within each article-repository pair. This broad application accounts for the complex realities of scientific software development practices, particularly the common occurrence of researchers maintaining multiple developer accounts across different projects or institutions, and account transitions as researchers move between roles.

While our model demonstrates strong performance, we acknowledge certain limitations in our approach. Notably, the model's performance can be affected by shorter names (both usernames and display names) where less textual information is available for matching. Additionally, while organization accounts (such as research lab accounts used for project management) present a potential challenge for accurate matching, our filtering mechanisms applied before analysis help minimize their impact in modeling.

The resulting dataset represents, to our knowledge, the first large-scale collection of linked article-repository and author-developer-account pairs, particularly in the physical sciences. Specifically, our dataset contains 138596 article-repository pairs, with 295806 distinct authors and 152170 distinct developer accounts. From these distinct entities, we identify 108754 annotated author-developer pairs. A detailed breakdown of these counts by data source, domain, document type, and open access status is available in Table 1.

Table 1: Counts of Article-Repository Pairs, Authors, and Developers by Data Sources, Domains, Document Types, and Access Status.

Category	Subset	Article-Repository Pairs	Authors	Developers
By Domain	Physical Sciences	116597	240536	130601
	Social Sciences	8839	29239	14023
	Life Sciences	7727	31613	12123
	Health Sciences	5176	26022	7277
By Document Type	preprint	72177	170301	87311
	research article	63528	173183	78935
	software article	2891	9294	12868
By Access Status	Open	132856	286874	147831
	Closed	5740	23668	9352
By Data Source	pwc	129615	262889	134926
	plos	6090	30233	8784
	joss	2336	7105	11362
	softwarex	555	2244	1628
Total		**138596**	**295806**	**152170**

4 Preliminary Analysis of Code Contributor Authorship and Development Dynamics of Research Teams

4.1 Software Development Dynamics Within Research Teams

Understanding the composition and dynamics of software development teams provides essential context for analyzing how code contributions relate to scientific recognition and impact. To ensure reliable analysis, we focus on a subset of our data that includes only article-repository pairs with at least one article citation, repository commit activity that at the latest falls within 90 days of publication, and research teams of typical size (removing those with fewer than 3 authors and less than 12 authors, the 97th percentile for research team size). In addition, we filter out any author-developer pairs which have a predictive model confidence of less than 0.97.

Within this filtered dataset, we categorized individuals into three groups: code-contributing authors (CC-A) who both authored papers and contributed code to associated repositories, non-code-contributing authors (NCC-A) who authored papers but showed no evidence of code contributions, and code-contributing non-authors (CC-NA) who contributed code but received no authorship recognition. This categorization revealed that papers in our dataset typically have 4.9 ś 1.9 total authors, with 1.0 ś 0.7 code-contributing authors and 3.9 ś 2.0 non-code-contributing authors. Beyond the author list, papers averaged 0.5 ś 1.7 code-contributing non-authors. Table 2 provides a detailed breakdown of these distributions by domain, article type, and open access status.

Perhaps most striking is our finding that 6586 papers (28.9%) have at least one code contributor who did not receive authorship recognition. Within this substantial subset of papers, we found an average of 1.6 ś 2.9 unrecognized code contributors per paper. The presence of only one code-contributing author per paper, on average, aligns with previous research by Larivière, Pontille, and Sugimoto (2020) showing that technical tasks like data curation, formal analysis, visualization, and software development typically fall to first authors. However, our finding that over a quarter of papers have unrecognized code contributors suggests a more complex dynamic between software development and authorship recognition.

Table 2: Mean and Standard Deviation of Non-Code-Contributing Authors (NCC-A), Code-Contributing Authors (CC-A), and Code-Contributing Non-Authors (CC-NA) Research Team Members by Domain, Article Type, and Open Access Status. Only includes research teams from article-repository pairs with a most recent commit no later than 90 days after publication and excludes research teams which are in the top 3% of total author sizes.

Control	Subset	Total Authors	NCC-A	CC-A	CC-NA
OA Status	Closed	5.1 ś 1.9	4.1 ś 1.9	1.0 ś 0.7	0.6 ś 2.1
	Open	4.9 ś 1.9	3.9 ś 2.0	1.0 ś 0.7	0.4 ś 1.7
Domain	Health Sciences	6.1 ś 2.5	5.1 ś 2.6	0.9 ś 0.6	0.4 ś 1.2
	Life Sciences	5.2 ś 2.1	4.2 ś 2.2	1.0 ś 0.7	0.4 ś 1.2
	Physical Sciences	4.8 ś 1.8	3.8 ś 1.9	1.0 ś 0.7	0.5 ś 1.8
	Social Sciences	4.5 ś 1.7	3.5 ś 1.8	1.1 ś 0.7	0.3 ś 1.1
Article Type	preprint	4.8 ś 1.8	3.8 ś 1.9	1.0 ś 0.7	0.6 ś 2.2
	research article	4.9 ś 1.9	3.9 ś 2.0	1.0 ś 0.7	0.4 ś 1.6
	software article	4.7 ś 1.9	3.2 ś 1.9	1.5 ś 1.4	1.0 ś 1.1
Overall		**4.9 ś 1.9**	**3.9 ś 2.0**	**1.0 ś 0.7**	**0.5 ś 1.7**

When examining these patterns over time and across different team sizes (Figure 1), we found that both the number of code-contributing authors and unrecognized contributors has remained relatively stable. This stability suggests that while the exclusion of code contributors from authorship isn't worsening, it represents a persistent feature of scientific software development rather than a historical artifact or transition period in research practices.

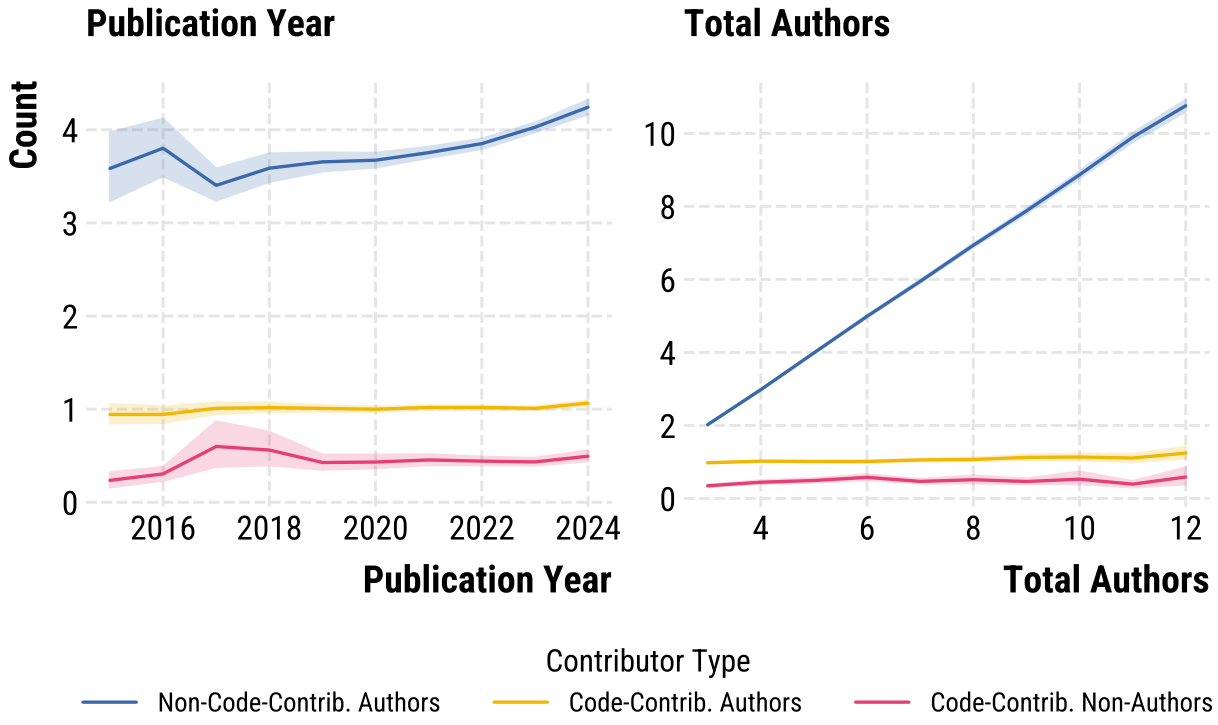

Figure 1: Mean number of Non-Code-Contributing Authors, Code-Contributing Authors, and Code-Contributing Non-Authors by Publication Year and by Total Number of Authors. Only includes article-repository pairs with a most recent commit no later than 90 days after publication and excludes research-teams which are in the top 3% of total author sizes for publication years with 50 or more articles.

4.1.1 Modeling Article Citations

Building upon previous work examining the effects of team size and team diversity on scientific impact and software quality (see Section 2), we investigate how the number of code contributors within a research team may be associated with an article's research impact. We hypothesize that more code contributors may signal greater technical complexity

in research, which may be associated with higher citation counts as the community builds upon more technically sophisticated works.

Using our filtered dataset of article-repository pairs, we conducted multiple regression analyses to examine these relationships while controlling for various factors. Without controlling for domain, open access, or article type differences (Table 5), we found a modest positive association between the number of code-contributing authors and article citations, with each code-contributing author being associated with a 5.2% increase in article citations ($p < 0.001$).

When controlling for article type (Table 8), we observed divergent patterns between preprints and research articles. For preprints, each code-contributing non-author was associated with a statistically significant 3.1% decrease in citations ($p < 0.005$). In contrast, research articles showed more positive associations: we found a significant positive relationship between code-contributing authors and citations ($p < 0.001$), though we cannot estimate the precise magnitude due to the non-significant main effect in the model. Additionally, each code-contributing non-author was associated with a 0.7% increase in expected citations for research articles ($p < 0.001$).

Overall, while we find statistically significant associations between code contributions and citation counts, these effects are relatively modest in magnitude.

4.2 Characteristics of Scientific Code Contributors

4.2.1 Author Positions of Code Contributing Authors

Building upon previous work examining the relationship between authorship position and research contributions, we investigate how author position may relate to code contribution patterns. We hypothesize that first authors, who traditionally contribute the bulk of intellectual and experimental work, would be most likely to contribute code to a project, while middle and last authors, who often provide oversight and guidance, would be less likely to contribute code.

To analyze these patterns within our previously filtered dataset of article-repository pairs, we first conducted chi-square tests of independence between author position and code contribution status. These tests revealed significant associations both overall and when controlling for research domain, article type, and open access status (all $p < 0.01$). Following these significant results, detailed post-hoc binomial tests (Table 9) revealed clear position-based differences: 67.4% of first authors contributed code to their projects, compared to only 8.5% of middle authors and 8.7% of last authors. These differences remained statistically significant across all tested scenarios, regardless of research domain, article type, or open access status.

These patterns strongly align with traditional scientific authorship conventions, where first authors typically take primary responsibility for both intellectual and technical aspects of the research, while middle and last authors more commonly provide oversight and guidance. The consistency of these findings across different subsets of our data suggests a deeply embedded relationship between author position and technical contributions in scientific software development.

4.2.2 Corresponding Status of Code Contributing Authors

Building upon our analysis of author position, we next examine how corresponding author status relates to code contribution patterns. We hypothesize that corresponding authors, who traditionally maintain research artifacts and serve as primary points of contact, would be more likely to contribute code compared to non-corresponding authors, as this role often involves responsibility for project resources and materials.

To analyze these relationships within our filtered dataset of article-repository pairs, we conducted chi-square tests of independence between corresponding author status and code contribution status. Surprisingly, these tests revealed patterns contrary to our initial hypothesis. Both corresponding and non-corresponding authors were significantly less likely to be code contributors than would be expected by chance ($p < 0.01$), with only 29.5% of corresponding authors and 19.9% of non-corresponding authors contributing code. Detailed post-hoc binomial tests (Table 10) revealed this pattern holds true across nearly all conditions, with only two notable exceptions: corresponding authors in software-focused articles and those in closed-access publications showed no significant difference in their likelihood to contribute code.

These findings challenge conventional assumptions about the relationship between corresponding authorship and technical contributions. While corresponding authors are traditionally responsible for maintaining research artifacts, our results suggest this responsibility may not typically extend to direct engagement with software development.

4.2.3 Modeling Author H-Index

Table 3: Counts of Researcher Coding Status Used in H5

Control	Subset	Any Coding	Majority Coding	Always Coding	Total
Freq. Author Pos.	First	1689	4952	3322	11444
	Middle	12184	3327	701	31911
	Last	2413	568	191	10188
Freq. Domain	Health Sciences	346	202	87	1507
	Life Sciences	369	241	132	1440
	Physical Sciences	15286	8176	3815	49294
	Social Sciences	285	228	180	1302
Freq. Article Type	Preprint	9572	4963	2214	28948
	Research Article	6669	3765	1871	24217
	Software Article	45	119	129	378

Building upon previous work examining career implications for researchers who prioritize software development Section 2, we investigated how varying levels of code contribution relate to scholarly impact through h-index metrics. To ensure a robust analysis, we applied several key data filtering steps. We only included researchers with at least three publications in our dataset, removed those with more than three developer account associations, and used each researcher's most common (or most recent) domain, article type, and author position, with ties broken by the most recent occurrence. We removed h-index outliers by excluding researchers below the bottom 3rd and above the top 97th percentiles. Finally, we removed any author-developer-account pairs which had a predictive model confidence of less than 0.97.

We categorized researchers' coding contributions into mutually exclusive groups: non-coders (no code contributions), any coding (code contribution in less than half of article-repository pairs), majority coding (code contribution in at least half, but not all, article-repository pairs), and always coding (code contribution in every article-repository pair).

Our analysis revealed a consistent and statistically significant negative relationship between code contribution frequency and h-index across multiple analytical controls. In our initial uncontrolled analysis (Table 11), we observed increasingly negative h-index effects as coding frequency increased. Compared to non-coding authors, researchers were associated with progressively lower h-indices: occasional code contributors showed a ~27.3% lower h-index ($p < 0.001$), majority code contributors demonstrated a ~53.5% lower h-index ($p < 0.001$), and always coding authors exhibited a ~62.1% lower h-index ($p < 0.001$).

When controlling for author position (Table 12), we found a general pattern of reduced h-indices with increased code contribution, with one notable exception. Occasional coding first authors were associated with a ~14.9% higher h-index ($p < 0.001$), while always coding first authors saw a ~21.6% reduction compared to non-coding first authors ($p < 0.001$). For middle and last authors, the pattern was more consistently negative. Middle authors who occasionally coded showed a ~26.6% lower h-index ($p < 0.001$), and those always coding demonstrated a ~52.9% lower h-index ($p < 0.001$). Similarly, last authors who occasionally coded experienced a ~13.1% lower h-index ($p < 0.001$), with always coding authors showing a ~45.7% lower h-index ($p < 0.001$).

When controlling for research domain (Table 13), majority coding scientists showed significant h-index reductions across all domains. Health sciences researchers saw the most dramatic reduction at ~76.5% ($p < 0.001$), followed by physical sciences at ~52.6% ($p < 0.001$), social sciences at ~51.4% ($p < 0.001$), and life sciences at ~47.1% ($p < 0.001$).

Analyzing by common article type (Table 14) revealed similar patterns. For authors primarily publishing preprints, the h-index reductions were substantial: ~25.6% for occasional coding, ~53.5% for majority coding, and ~62.9% for always coding authors. Authors primarily publishing software articles showed slightly different but still significant reductions: ~33.1% for majority coding and ~33.0% for always coding authors.

Taken as a whole, these findings indicate that the more frequently an author contributes code, the lower their h-index is likely to be relative to their peers, with the notable exception of first authors who occasionally contribute code. What makes these results particularly striking is that each of our models includes publication count as an input feature, suggesting that these h-index reductions persist even when accounting for total research output.

5 Discussion

Our analysis reveals significant disparities in the recognition of software contributions to scientific research, with nearly 30% of articles having non-author code-contributors who received no formal authorship credit. This persistent pattern suggests a systemic disconnect between software development and scientific recognition systems, reflecting challenges in how scientific contributions are valued and credited. This exclusion reflects what Shapin (1989) observed about scientific authority—the selective attribution of technical work as either genuine knowledge or mere skill significantly impacts who receives formal recognition. These findings further support previous research by Philippe et al. (2019) and Carver et al. (2022) documenting the frequent relegation of software contributors to either acknowledgment sections or receiving no credit at all, rather than authorship positions, despite the increasingly central role of software in scientific inquiry. The stability of this pattern over time indicates this is not a transitional phenomenon but rather an embedded feature of scientific software development, raising questions about scientific labor and how technical contributions are integrated into reward structures.

The distribution of code contributions across author positions provides context to the hierarchical organization of scientific work. First authors emerge as significantly more likely to contribute code with 66.8% of all first authors in our dataset contributing code. Middle and last authors meanwhile were statistically significantly less likely to contribute code with only 9.1% of middle authors and 9.8% of last authors acting as code contributing members of the research team. Corresponding authors were similarly less likely than expected to be code contributors as we found that within our dataset, corresponding authors were code contributors 32.6% of the time. These patterns align with traditional scientific labor distribution where first authors typically handle technical aspects of research while middle and last authors are likely specialist contributors or provide guidance and oversight (Larivière, Pontille, and Sugimoto 2020; Sauermann and Haeussler 2017). However, our initial hypothesis that corresponding authors would also be more likely to contribute code due to their common resposibility for long-term maintenance of research artifacts' was not supported by our data. This finding suggests a potential strict division between project management responsibilities and direct technical engagement with software development.

The modest citation advantage associated with code-contributing authors (4.5% increase in citations per code-contributing author) stands in contrast with the significant negative relationship between coding frequency and an individual's scholarly impact (h-index). This misalignment between technical contributions and scientific recognition creates an asymetrical relationship in which software development may enhance research impact but potentially penalizes individual careers. The progressive reduction in h-index as coding frequency increases indicates a cumulative disadvantage for frequent code contributors. This pattern persists even when controlling for publication count, suggesting issues in how software contributions are valued relative to other scientific outputs. These findings echo concerns raised by Muna et al. (2016) about the sustainability of research software development and highlight how current reward structures may discourage talented developers from pursuing scientific careers.

Software development represents a form of scholarly labor that has become increasingly essential to modern research yet remains incompletely integrated into formal recognition systems. Similar to the high proportion of articles with authors who made data curation contributions towards research observed by Larivière, Pontille, and Sugimoto (2020), our finding that a quarter of papers have unacknowledged code contributors highlights a labor role that is simultaneously common and undervalued. The prevalence of code contributions across domains demonstrates the importance of this work to contemporary research, yet the persistent exclusion of contributors from authorship suggests that software development continues to be classified as technical support rather than intellectual contribution. This classification may reflect disciplinary traditions that privilege certain forms of scholarly production, despite the growing recognition that software itself represents a legitimate research output (Katz et al. 2020). The tension between software's importance and contributors' recognition status raises questions about how we define, value, and reward different forms of scientific labor in an increasingly computational research landscape.

5.1 Limitations

Our data collection approach introduces several methodological constraints that should be considered when interpreting these results. By focusing exclusively on GitHub repositories, we likely miss contributions stored on alternative platforms such as GitLab, Bitbucket, or institutional repositories, potentially skewing our understanding of contribution patterns. As Trujillo, Hébert-Dufresne, and Bagrow (2022), Cao et al. (2023), and Escamilla et al. (2022) have all noted, while GitHub is the predominate host of scientific software, significant portions of research code exist on other platforms. Additionally, our reliance on public repositories means we cannot account for private repositories or code that was never publicly shared, potentially underrepresenting sensitive research areas or proprietary methods.

Our predictive modeling approach for matching authors with developer accounts presents additional limitations in how we operationalize the relationship between code contributions and authorship. The model's performance can be affected by shorter names where less textual information is available for matching, potentially creating biases against

researchers from cultures with shorter naming conventions. Organization accounts used for project management pose particular challenges for accurate matching, and while we implemented filtering mechanisms to minimize their impact, some misclassifications may persist. Furthermore, our approach may not capture all code contributors if multiple individuals developed code but only one uploaded it to a repository, creating attribution artifacts that may systematically underrepresent certain types of contributors, particularly junior researchers or technical staff who may not have direct repository access.

Our analytical approach required substantial data filtering to ensure reliable results, which introduces potential selection biases in our sample. By focusing on article-repository pairs with commit activity no later than 90 days past the date of article publication and at least 3 authors and less than 12 authors, we may have systematically excluded certain types of research projects, particularly those with extended development timelines or unusually large collaborations. Our categorization of coding status (non-coder, any coding, majority coding, always coding) necessarily simplifies complex contribution patterns and does not account for the quality, complexity, or significance of code contributions. Additionally, our reliance on OpenAlex metadata introduces certain limitations to our analysis. While OpenAlex provides good overall coverage, it lags behind proprietary databases in indexing references and citations which may affect our citation-based analyses and the completeness of author metadata used in our study (Alperin et al. 2024).

5.2 Future Work

Future technical improvements may enhance our understanding of the relationship between software development and scientific recognition systems. Expanding analysis beyond GitHub to include other code hosting platforms would provide a more comprehensive understanding of scientific software development practices across different domains and institutional contexts. More sophisticated entity matching techniques could improve author-developer account identification, particularly for cases with limited information or common names. Developing more nuanced measures of code contribution quality and significance beyond binary contribution identification would better capture the true impact of technical contributions to research. These methodological advances would enable more precise tracking of how code contributions translate—or fail to translate—into formal scientific recognition, providing clearer evidence for policy interventions.

Our findings point to several directions for future research on the changing nature of scientific labor and recognition. Longitudinal studies tracking how code contribution patterns affect career trajectories would provide valuable insights into the long-term impacts of the observed h-index disparities and whether these effects vary across career stages. Comparative analyses across different scientific domains could reveal discipline-specific norms and practices around software recognition, potentially identifying models that more equitably credit technical contributions. Qualitative studies examining how research teams make authorship decisions regarding code contributors would complement our quantitative findings by illuminating the social and organizational factors that influence recognition practices. Additionally, to better understand corresponding authors' role in maintaining research artifacts, future work could remove the 90-day post-publication commit activity filter to examine long-term sustainability actions, though this approach would need to address the introduction of contributors unrelated to the original paper.

The persistent underrecognition of software contributions despite their growing importance suggests a need for structural interventions in how we conceptualize and reward scientific work. Building upon efforts like CRediT (Brand et al. 2015), future work should investigate potential policy changes to better align institutional incentives with the diverse spectrum of contributions that drive modern scientific progress. However, as the example of CRediT demonstrates, even well-intentioned taxonomies may reproduce existing hierarchies or create new forms of inequality if they fail to address underlying power dynamics in scientific communities. The challenge is not merely technical but social: how to create recognition systems that simultaneously support innovation, ensure appropriate credit, maintain research integrity, and foster equitable participation in an increasingly computational scientific enterprise.

6 References

Aksnes, Dag W. 2006. "Citation Rates and Perceptions of Scientific Contribution." *J. Assoc. Inf. Sci. Technol.* 57: 169–85.

Alperin, Juan Pablo, Jason Portenoy, Kyle Demes, Vincent Larivière, and Stefanie Haustein. 2024. "An Analysis of the Suitability of OpenAlex for Bibliometric Analyses." *arXiv Preprint arXiv:2404.17663*.

Biagioli, Mario, and Peter Galison. 2014. *Scientific Authorship: Credit and Intellectual Property in Science*. Routledge.

Brand, Amy, Liz Allen, Micah Altman, Marjorie Hlava, and Jo Scott. 2015. "Beyond Authorship: Attribution, Contribution, Collaboration, and Credit." *Learned Publishing* 28 (2).

Cao, Hancheng, Jesse Dodge, Kyle Lo, Daniel A. McFarland, and Lucy Lu Wang. 2023. "The Rise of Open Science: Tracking the Evolution and Perceived Value of Data and Methods Link-Sharing Practices." *ArXiv* abs/2310.03193.

Carver, Jeffrey C., Nic Weber, Karthik Ram, Sandra Gesing, and Daniel S. Katz. 2022. "A Survey of the State of the Practice for Research Software in the United States." *PeerJ Computer Science* 8.

Chen, Liyue, Jie-lan Ding, Donghuan Song, and Zihao Qu. 2024. "Exploring Scientific Contributions Through Citation Context and Division of Labor." *ArXiv* abs/2410.13133.

Chinchilla-Rodrguez, Zaida, Rodrigo Costas, Vicent Lariviére, Nicolás Robinson-Garca, and Cassidy R Sugimoto. 2022. "The Relationship Between Corresponding Authorship and Author Position." In *Proceedings of the 26th International Conference on Science, Technology and Innovation Indicators (STI 2022)*, 7–9.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding." *CoRR* abs/1810.04805. http://arxiv.org/abs/1810.04805.

Edwards, Paul N, Steven J Jackson, Melissa K Chalmers, Geoffrey C Bowker, Christine L Borgman, David Ribes, Matt Burton, and Scout Calvert. 2013. "Knowledge Infrastructures: Intellectual Frameworks and Research Challenges."

Escamilla, Emily, Martin Klein, Talya Cooper, Vicky Rampin, Michele C. Weigle, and Michael L. Nelson. 2022. "The Rise of GitHub in Scholarly Publications." In *Linking Theory and Practice of Digital Libraries*, edited by Gianmaria Silvello, Oscar Corcho, Paolo Manghi, Giorgio Maria Di Nunzio, Koraljka Golub, Nicola Ferro, and Antonella Poggi, 187–200. Cham: Springer International Publishing.

Franceschet, Massimo, and Antonio Costantini. 2010. "The Effect of Scholar Collaboration on Impact and Quality of Academic Papers." *J. Informetrics* 4: 540–53.

Gøtzsche, Peter C, Asbjørn Hróbjartsson, Helle Krogh Johansen, Mette T Haahr, Douglas G Altman, and An-Wen Chan. 2007. "Ghost Authorship in Industry-Initiated Randomised Trials." *PLoS Medicine* 4 (1): e19.

Haeussler, Carolin, and Henry Sauermann. 2013. "Credit Where Credit Is Due? The Impact of Project Contributions and Social Factors on Authorship and Inventorship." *Research Policy* 42 (3): 688–703. https://doi.org/10.1016/j.respol.2012.09.009.

Hampton, Stephanie E, Carly A Strasser, Joshua J Tewksbury, Wendy K Gram, Amber E Budden, Archer L Batcheller, Clifford S Duke, and John H Porter. 2013. "Big Data and the Future of Ecology." *Frontiers in Ecology and the Environment* 11 (3): 156–62.

Hasselbring, Wilhelm, Stephan Druskat, Jan Bernoth, Philine Betker, Michael Felderer, Stephan Ferenz, Anna-Lena Lamprecht, Jan Linxweiler, and Bernhard Rumpe. 2024. "Toward Research Software Categories." https://arxiv.org/abs/2404.14364.

Hata, Hideaki, Jin L. C. Guo, Raula Gaikovina Kula, and Christoph Treude. 2021. "Science-Software Linkage: The Challenges of Traceability Between Scientific Knowledge and Software Artifacts." *ArXiv* abs/2104.05891.

He, Pengcheng, Jianfeng Gao, and Weizhu Chen. 2021. "DeBERTaV3: Improving DeBERTa Using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing." https://arxiv.org/abs/2111.09543.

He, Pengcheng, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. "DEBERTA: DECODING-ENHANCED BERT WITH DISENTANGLED ATTENTION." In *International Conference on Learning Representations*. https://openreview.net/forum?id=XPZIaotutsD.

Howison, James, and James D. Herbsleb. 2011. "Scientific Software Production: Incentives and Collaboration." In *Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work*, 513–22. CSCW '11. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/1958824.1958904.

———. 2013. "Incentives and Integration in Scientific Software Production." *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*.

Jeong, Yuna, and Eunhui Kim. 2022. "Scideberta: Learning Deberta for Science Technology Documents and Fine-Tuning Information Extraction Tasks." *IEEE Access* 10: 60805–13.

Jin, Xiaolong, Benjamin W Wah, Xueqi Cheng, and Yuanzhuo Wang. 2015. "Significance and Challenges of Big Data Research." *Big Data Research* 2 (2): 59–64.

Júnior, Edilson Anselmo Corrêa, Filipi Nascimento Silva, Luciano da Fontoura Costa, and Diego Raphael Amancio. 2016. "Patterns of Authors Contribution in Scientific Manuscripts." *J. Informetrics* 11: 498–510.

Katz, Daniel S., Neil P. Chue Hong, Tim Clark, August Muench, Shelley Stall, Daina R. Bouquin, Matthew Cannon, et al. 2020. "Recognizing the Value of Software: A Software Citation Guide." *F1000Research* 9.

Kelley, Aidan, and Daniel Garijo. 2021. "A Framework for Creating Knowledge Graphs of Scientific Software Metadata." *Quantitative Science Studies* 2: 1423–46.

Krafczyk, Matthew, August Shi, Adhithya Bhaskar, Darko Marinov, and Victoria Stodden. 2019. "Scientific Tests and Continuous Integration Strategies to Enhance Reproducibility in the Scientific Software Context." In *Proceedings of the 2nd International Workshop on Practical Reproducible Evaluation of Computer Systems*, 23–28. P-RECS '19. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3322790.3330595.

Lamprecht, Anna-Lena, Leyla J. García, Mateusz Kuzak, Carlos Martinez, Ricardo Arcila, Eva Martin Del Pico, Victoria Dominguez Del Angel, et al. 2020. "Towards FAIR Principles for Research Software." *Data Sci.* 3: 37–59.

Larivière, Vincent, Nadine Desrochers, Benoît Macaluso, Philippe Mongeon, Adèle Paul-Hus, and Cassidy R Sugimoto. 2016. "Contributorship and Division of Labor in Knowledge Production." *Social Studies of Science* 46 (3): 417–35. https://doi.org/10.1177/0306312716650046.

Larivière, Vincent, Yves Gingras, Cassidy R. Sugimoto, and Andrew Tsou. 2014. "Team Size Matters: Collaboration and Scientific Impact Since 1900." *Journal of the Association for Information Science and Technology* 66.

Larivière, Vincent, David Pontille, and Cassidy R. Sugimoto. 2020. "Investigating the Division of Scientific Labor Using the Contributor Roles Taxonomy (CRediT)." *Quantitative Science Studies*, 1–18.

Li, Kai, Chenwei Zhang, and Vincent Larivière. 2023. "Are Research Contributions Assigned Differently Under the Two Contributorship Classification Systems in PLoS ONE?" *ArXiv* abs/2310.11687.

Liu, Xin, Chengjing Zhang, and Jiang Li. 2023. "Conceptual and Technical Work: Who Will Disrupt Science?" *Journal of Informetrics* 17 (3): 101432. https://doi.org/10.1016/j.joi.2023.101432.

Lu, Chao, Yingyi Zhang, Yong-Yeol Ahn, Ying Ding, Chenwei Zhang, and Dandan Ma. 2019. "Cocontributorship Network and Division of Labor in Individual Scientific Collaborations." *Journal of the Association for Information Science and Technology* 71: 1162–78.

Mayernik, Matthew S, David L Hart, Keith E Maull, and Nicholas M Weber. 2017. "Assessing and Tracing the Outcomes and Impact of Research Infrastructures." *Journal of the Association for Information Science and Technology* 68 (6): 1341–59.

Merow, Cory, Brad L. Boyle, Brian J. Enquist, Xiao Feng, Jamie M. Kass, Brian Salvin Maitner, Brian McGill, et al. 2023. "Better Incentives Are Needed to Reward Academic Software Development." *Nature Ecology & Evolution* 7: 626–27.

Milewicz, Reed, Gustavo Pinto, and Paige Rodeghero. 2019. "Characterizing the Roles of Contributors in Open-Source Scientific Software Projects." In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, 421–32. https://doi.org/10.1109/MSR.2019.00069.

Muna, Demitri, Michael Alexander, Alice Allen, Richard Ashley, Daniel Asmus, Ruyman Azzollini, Michele Bannister, et al. 2016. "The Astropy Problem." https://arxiv.org/abs/1610.03159.

Ni, Chaoqun, Elise Smith, Haimiao Yuan, Vincent Larivière, and Cassidy R. Sugimoto. 2021. "The Gendered Nature of Authorship." *Science Advances* 7 (36): eabe4639. https://doi.org/10.1126/sciadv.abe4639.

Philippe, Olivier, Martin Hammitzsch, Stephan Janosch, Anelda van der Walt, Ben van Werkhoven, Simon Hettrick, Daniel S. Katz, et al. 2019. "Softwaresaved/International-Survey: Public Release for 2018 Results." Zenodo. https://doi.org/10.5281/zenodo.2585783.

Ram, Karthik. 2013. "Git Can Facilitate Greater Reproducibility and Increased Transparency in Science." *Source Code for Biology and Medicine* 8: 1–8.

Sanh, Victor, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. "DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter." *ArXiv* abs/1910.01108.

Sauermann, Henry, and Carolin Haeussler. 2017. "Authorship and Contribution Disclosures." *Science Advances* 3 (11): e1700404. https://doi.org/10.1126/sciadv.1700404.

Shapin, Steven. 1989. "The Invisible Technician." *American Scientist* 77 (6): 554–63. http://www.jstor.org/stable/27856006.

Sharma, Nitesh Kumar, Ram Ayyala, Dhrithi Deshpande, Yesha Patel, Viorel Munteanu, Dumitru Ciorba, Viorel Bostan, et al. 2024. "Analytical Code Sharing Practices in Biomedical Research." *PeerJ Computer Science* 10: e2066.

Smith, Arfon M., Daniel S. Katz, and Kyle E. Niemeyer. 2016. "Software Citation Principles." *PeerJ Comput. Sci.* 2: e86.

Stankovski, Aleksandar, and Daniel Garijo. 2024. "RepoFromPaper: An Approach to Extract Software Code Implementations from Scientific Publications." In *NSLP*.

Stodden, Victoria, Peixuan Guo, and Zhaokun Ma. 2013. "Toward Reproducible Computational Research: An Empirical Analysis of Data and Code Policy Adoption by Journals." *PloS One* 8 (6): e67111.

Tran, Hanh Thi Hong, Nishan Chatterjee, Senja Pollak, and Antoine Doucet. 2024. "DeBERTa Beats Behemoths: A Comparative Analysis of Fine-Tuning, Prompting, and PEFT Approaches on LegalLensNER." In *Proceedings of the Natural Legal Language Processing Workshop 2024*, 371–80.

Trisovic, Ana, Matthew K. Lau, Thomas Pasquier, and Mercè Crosas. 2021. "A Large-Scale Study on Research Code Quality and Execution." *Scientific Data* 9.

Trujillo, Milo Z, Laurent Hébert-Dufresne, and James Bagrow. 2022. "The Penumbra of Open Source: Projects Outside of Centralized Platforms Are Longer Maintained, More Academic and More Collaborative." *EPJ Data Science* 11 (1): 31.

Westner, Britta U., Daniel R. McCloy, Eric Larson, Alexandre Gramfort, Daniel S. Katz, Arfon M. Smith, and invited co-signees. 2024. "Cycling on the Freeway: The Perilous State of Open Source Neuroscience Software." *ArXiv* abs/2403.19394.

Wyss, Elizabeth, Lorenzo De Carli, and Drew Davidson. 2023. "(Nothing but) Many Eyes Make All Bugs Shallow." In *Proceedings of the 2023 Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses*, 53–63. SCORED '23. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3605770.3625216.

Yu, Liqiang, Bo Liu, Qunwei Lin, Xinyu Zhao, and Chang Che. 2024. "Semantic Similarity Matching for Patent Documents Using Ensemble BERT-Related Model and Novel Text Processing Method." *arXiv Preprint arXiv:2401.06782*.

7 Appendix

7.1 Extended Modeling and Analysis Results and Supporting Tables

7.1.1 Full Comparison of Models and Optional Features for Author-Developer-Account Matching

Table 4: Comparison of Models for Author-Developer-Account Matching

Optional Feats.	Model	Accuracy	Precision	Recall	F1
name	deberta	0.984	0.938	0.950	0.944
name, email	bert-multilingual	0.984	0.938	0.950	0.944
name, email	deberta	0.982	0.907	0.975	0.940
name	bert-multilingual	0.982	0.938	0.938	0.938
name	distilbert	0.978	0.936	0.912	0.924
name, email	distilbert	0.978	0.936	0.912	0.924
email	deberta	0.957	0.859	0.838	0.848
email	bert-multilingual	0.950	0.894	0.738	0.808
n/a	deberta	0.946	0.847	0.762	0.803
n/a	bert-multilingual	0.941	0.862	0.700	0.772
n/a	distilbert	0.856	0.000	0.000	0.000
email	distilbert	0.856	0.000	0.000	0.000

7.1.2 Linear Models for Software Development Dynamics Within Research Teams

Table 5: Article Citations by Code Contributorship of Research Team

Variable	coef	P>|z|	[0.025	0.975]
const ***	**0.98**	**0.00**	**0.93**	**1.03**
Total Authors ***	**0.07**	**0.00**	**0.06**	**0.08**
Code-Contrib. Authors ***	**0.05**	**0.00**	**0.03**	**0.07**
Code-Contrib. Non-Authors	-0.00	0.65	-0.01	0.01
Years Since Publication ***	**0.39**	**0.00**	**0.38**	**0.40**

Table 6: Article Citations by Code Contributorship of Research Team Controlled by Open Access Status

Variable	coef	P>|z|	[0.025	0.975]
const ***	**0.61**	**0.00**	**0.49**	**0.73**
Total Authors ***	**0.07**	**0.00**	**0.06**	**0.08**
Code-Contrib. Authors	0.05	0.26	-0.04	0.14
Code-Contrib. Non-Authors	0.00	0.94	-0.03	0.03
Years Since Publication ***	**0.38**	**0.00**	**0.37**	**0.39**
Is Open Access ***	**0.41**	**0.00**	**0.29**	**0.53**
Code-Contrib. Authors * Is Open Access	-0.00	0.98	-0.09	0.09
Code-Contrib. Non-Authors * Is Open Access	-0.00	0.87	-0.03	0.03

Table 7: Article Citations by Code Contributorship of Research Team Controlled by Domain

Variable	coef	P>|z|	[0.025	0.975]
const ***	**0.89**	**0.00**	**0.76**	**1.02**
Total Authors ***	**0.07**	**0.00**	**0.06**	**0.08**
Code-Contrib. Authors	0.02	0.66	-0.08	0.12
Code-Contrib. Non-Authors	0.01	0.62	-0.04	0.06
Years Since Publication ***	**0.40**	**0.00**	**0.39**	**0.41**
Domain Life Sciences ***	**-0.20**	**0.01**	**-0.35**	**-0.06**
Domain Physical Sciences ***	**0.13**	**0.04**	**0.00**	**0.25**
Domain Social Sciences ***	**-0.19**	**0.02**	**-0.35**	**-0.03**
Code-Contrib. Authors * Domain Life Sciences	0.07	0.30	-0.06	0.19
Code-Contrib. Authors * Domain Physical Sciences	0.02	0.74	-0.09	0.12
Code-Contrib. Authors * Domain Social Sciences	0.11	0.09	-0.02	0.24
Code-Contrib. Non-Authors * Domain Life Sciences	-0.04	0.26	-0.11	0.03
Code-Contrib. Non-Authors * Domain Physical Sciences	-0.02	0.55	-0.07	0.04
Code-Contrib. Non-Authors * Domain Social Sciences	-0.03	0.34	-0.10	0.04

Table 8: Article Citations by Code Contributorship of Research Team Controlled by Article Type

Variable	coef	P>|z|	[0.025	0.975]
const ***	**0.51**	**0.00**	**0.43**	**0.59**
Total Authors ***	**0.07**	**0.00**	**0.06**	**0.08**
Code-Contrib. Authors	-0.02	0.49	-0.07	0.03
Code-Contrib. Non-Authors ***	**-0.03**	**0.00**	**-0.05**	**-0.01**
Years Since Publication ***	**0.40**	**0.00**	**0.39**	**0.41**
Article Type Research Article ***	**0.49**	**0.00**	**0.41**	**0.56**
Article Type Software Article ***	**-0.46**	**0.00**	**-0.72**	**-0.21**
Code-Contrib. Authors * Article Type Research Article ***	**0.09**	**0.00**	**0.04**	**0.15**
Code-Contrib. Authors * Article Type Software Article	-0.07	0.30	-0.20	0.06
Code-Contrib. Non-Authors * Article Type Research Article ***	**0.04**	**0.00**	**0.02**	**0.06**
Code-Contrib. Non-Authors * Article Type Software Article	0.08	0.28	-0.07	0.22

7.1.3 Post-Hoc Tests for Coding vs Non-Coding Authors by Position

Table 9: Counts of Code-Contributing Authors ('Coding') as well as Total Authors by Position and Bonferroni Corrected p-values from Post-Hoc Binomial Tests

Control	Subset	Position	Coding	Total	p
Domain	**Health Sciences**	First	1535	2404	0.000***
		Middle	510	11513	0.000***
		Last	228	2352	0.000***
	Life Sciences	First	2538	3890	0.000***
		Middle	815	11854	0.000***
		Last	469	3778	0.000***
	Physical Sciences	First	28426	41983	0.000***
		Middle	10022	111735	0.000***
		Last	3303	40407	0.000***
	Social Sciences	First	2766	4044	0.000***
		Middle	989	9251	0.000***
		Last	397	3861	0.000***
Article Type	**Preprint**	First	13146	19523	0.000***
		Middle	4872	52925	0.000***
		Last	1432	18598	0.000***
	Research Article	First	21595	32081	0.000***
		Middle	7034	89991	0.000***
		Last	2801	31188	0.000***
	Software Article	First	524	717	0.000***
		Middle	430	1437	0.000***
		Last	164	612	0.000***
Open Access Status	**Closed Access**	First	2587	3742	0.000***
		Middle	861	10965	0.000***
		Last	266	3642	0.000***
	Open Access	First	32678	48579	0.000***
		Middle	11475	133388	0.000***
		Last	4131	46756	0.000***
Overall	**Overall**	First	35265	52321	0.000***
		Middle	12336	144353	0.000***
		Last	4397	50398	0.000***

7.1.4 Post-Hoc Tests for Coding vs Non-Coding Authors by Corresponding Status

Table 10: Counts of Code-Contributing Authors ('Coding') as well as Total Authors by Corresponding Status and Bonferroni Corrected p-values from Post-Hoc Binomial Tests

Control	Subset	Is Corresponding	Coding	Total	p
Domain	**Life Sciences**	Corresponding	1720	8014	0.000***
		Not Corresponding	2102	11508	0.000***
	Physical Sciences	Corresponding	5146	12483	0.000***
		Not Corresponding	36605	181642	0.000***
	Social Sciences	Corresponding	789	2455	0.000***
		Not Corresponding	3363	14701	0.000***
Article Type	**Preprint**	Corresponding	760	1036	0.000***
		Not Corresponding	18690	90010	0.000***
	Research Article	Corresponding	7545	27339	0.000***
		Not Corresponding	23885	125921	0.000***
	Software Article	Corresponding	206	438	0.464
		Not Corresponding	912	2328	0.000***
Open Access Status	**Closed Access**	Corresponding	250	468	0.304
		Not Corresponding	3464	17881	0.000***
	Open Access	Corresponding	8261	28345	0.000***
		Not Corresponding	40023	200378	0.000***
Overall	**Overall**	Corresponding	8511	28813	0.000***
		Not Corresponding	43487	218259	0.000***

7.1.5 Linear Models for Characterizing Code-Contributing Author H-Index

Table 11: Code-Contributing Authors H-Index by Coding Status

Variable	coef	P>|z|	[0.025	0.975]
const ***	**3.19**	**0.00**	**3.18**	**3.20**
Works Count ***	**0.00**	**0.00**	**0.00**	**0.00**
Any Coding ***	**-0.32**	**0.00**	**-0.33**	**-0.30**
Majority Coding ***	**-0.77**	**0.00**	**-0.79**	**-0.74**
Always Coding ***	**-0.97**	**0.00**	**-1.02**	**-0.92**

Table 12: Researcher H-Index by Coding Status Controlled by Most Freq. Author Position

Variable	coef	P>|z|	[0.025	0.975]
const ***	**2.38**	**0.00**	**2.31**	**2.44**
Works Count ***	**0.00**	**0.00**	**0.00**	**0.00**
Any Coding ***	**0.14**	**0.00**	**0.05**	**0.22**
Majority Coding	-0.07	0.08	-0.14	0.01
Always Coding ***	**-0.24**	**0.00**	**-0.33**	**-0.16**
Common Author Position Last ***	**1.05**	**0.00**	**0.98**	**1.11**
Common Author Position Middle ***	**0.74**	**0.00**	**0.68**	**0.81**
Any Coding * Common Author Position Last ***	**-0.28**	**0.00**	**-0.37**	**-0.19**
Any Coding * Common Author Position Middle ***	**-0.45**	**0.00**	**-0.53**	**-0.36**
Majority Coding * Common Author Position Last ***	**-0.35**	**0.00**	**-0.45**	**-0.26**
Majority Coding * Common Author Position Middle ***	**-0.61**	**0.00**	**-0.69**	**-0.52**
Always Coding * Common Author Position Last ***	**-0.37**	**0.00**	**-0.51**	**-0.22**
Always Coding * Common Author Position Middle ***	**-0.51**	**0.00**	**-0.64**	**-0.38**

Table 13: Researcher H-Index by Coding Status Controlled by Most Freq. Domain

Variable	coef	P>|z|	[0.025	0.975]
const ***	**3.32**	**0.00**	**3.29**	**3.36**
Works Count ***	**0.00**	**0.00**	**0.00**	**0.00**
Any Coding ***	**-0.39**	**0.00**	**-0.47**	**-0.30**
Majority Coding ***	**-1.44**	**0.00**	**-1.64**	**-1.24**
Always Coding ***	**-1.22**	**0.00**	**-1.59**	**-0.86**
Common Domain Life Sciences ***	**0.10**	**0.00**	**0.05**	**0.15**
Common Domain Physical Sciences ***	**-0.14**	**0.00**	**-0.18**	**-0.11**
Common Domain Social Sciences ***	**-0.16**	**0.00**	**-0.22**	**-0.10**
Any Coding * Common Domain Life Sciences	0.10	0.09	-0.02	0.21
Any Coding * Common Domain Physical Sciences	0.07	0.12	-0.02	0.16
Any Coding * Common Domain Social Sciences	-0.00	1.00	-0.14	0.14
Majority Coding * Common Domain Life Sciences ***	**0.80**	**0.00**	**0.57**	**1.03**
Majority Coding * Common Domain Physical Sciences ***	**0.69**	**0.00**	**0.49**	**0.89**
Majority Coding * Common Domain Social Sciences ***	**0.74**	**0.00**	**0.48**	**1.00**
Always Coding * Common Domain Life Sciences	0.31	0.14	-0.10	0.73
Always Coding * Common Domain Physical Sciences	0.25	0.17	-0.11	0.62
Always Coding * Common Domain Social Sciences	0.32	0.14	-0.11	0.75

Table 14: Researcher H-Index by Coding Status Controlled by Most Freq. Article Type

Variable	coef	P>|z|	[0.025	0.975]
const ***	**3.10**	**0.00**	**3.09**	**3.11**
Works Count ***	**0.00**	**0.00**	**0.00**	**0.00**
Any Coding ***	**-0.30**	**0.00**	**-0.32**	**-0.27**
Majority Coding ***	**-0.77**	**0.00**	**-0.81**	**-0.73**
Always Coding ***	**-0.99**	**0.00**	**-1.07**	**-0.92**
Common Article Type Research Article ***	**0.18**	**0.00**	**0.17**	**0.20**
Common Article Type Software Article ***	**0.22**	**0.00**	**0.12**	**0.33**
Any Coding * Common Article Type Research Article	-0.02	0.15	-0.05	0.01
Any Coding * Common Article Type Software Article	0.19	0.06	-0.01	0.39
Majority Coding * Common Article Type Research Article	0.01	0.80	-0.05	0.06
Majority Coding * Common Article Type Software Article ***	**0.36**	**0.00**	**0.19**	**0.54**
Always Coding * Common Article Type Research Article	0.00	0.93	-0.10	0.10
Always Coding * Common Article Type Software Article ***	**0.37**	**0.00**	**0.15**	**0.58**

7.2 Analysis of Project Duration and Percentage Code-Contributors Who Are Authors

In our pre-registered analysis plan (https://osf.io/fc74m), we originally hypothesized about the relationship between project duration and authorship recognition. Specifically, we posited that sustained technical engagement and scientific recognition might be meaningfully related, with longer project durations potentially leading to higher rates of code-contributor authorship. We saw repository histories as providing a unique opportunity to examine this relationship, leading us to hypothesize that projects with longer commit durations would be associated with higher percentages of developers receiving authorship recognition (pre-registered as H2).

However, our analysis found no evidence to support this hypothesis. When examining the relationship between a repository's commit duration and the percentage of developers who receive authorship recognition, we found no significant correlation (r = -0.00, p = n.s.). This suggests that the length of time a project has been in development has no meaningful relationship with the proportion of developers who are recognized as authors.

We ultimately decided to move this analysis to the appendix for two key methodological reasons. First, our approach of using repository-level commit duration as a proxy for individual contribution patterns proved too coarse-grained. A more precise analysis would need to examine individual-level contribution durations and patterns rather than overall project length. Second, our method did not account for the varying levels of contribution that different developers make to a repository. Simply correlating overall project duration with authorship rates fails to capture the nuanced ways that sustained, meaningful technical contributions might influence authorship decisions.

These limitations suggest potential directions for future work that could more rigorously examine the relationship between long-term technical engagement and scientific recognition. Such work might benefit from more granular analysis of individual contribution patterns, perhaps incorporating measures of contribution significance and sustainability rather than just temporal duration.