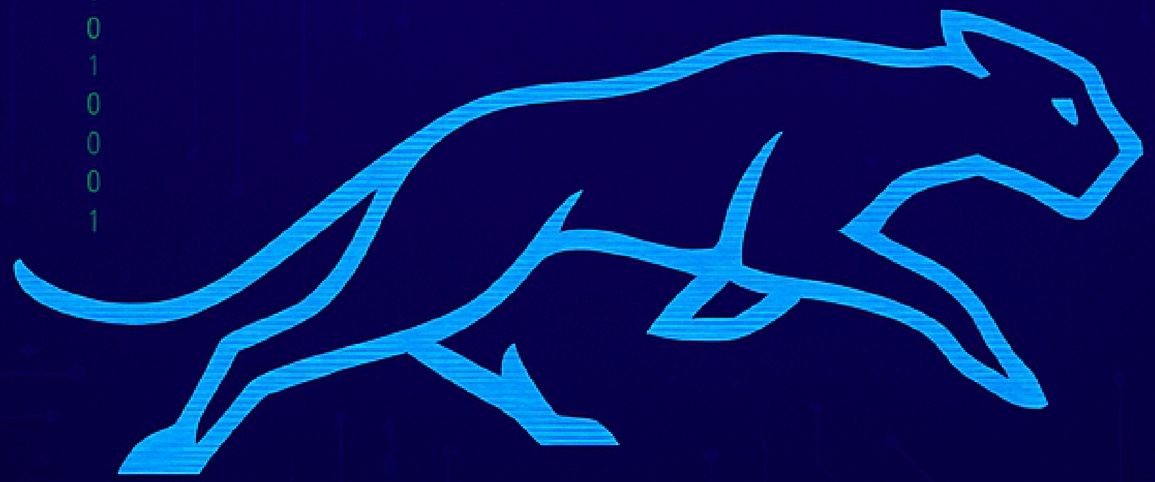
CODE RUNNERS



By: Iyana, Holy, Aaliyah, Arghavan, and Copernic

Workflow

Day 1: Project Kickoff & Criteria Definition

- Team selected papers, assigned roles, and defined reproducibility criteria.

Day 2: Scope Expansion & Automation Setup

- Expanded to multi-paper study (ICSE 2023 & SC24).
- Built 100-point scorecard.
- Developed semi-automated scoring with ChatGPT & Manus AI.
- GitHub repo setup with key files.

Day 3: PDF Processing & Scoring Improvements

- Implemented PDF download & text extraction.
- Identified scoring bugs (PDFs not analyzed).
- Improved accuracy with hybrid (manual + auto) scoring.
- Started Streamlit/Flask portal development.

Day 4: Portal & Poster Progress

- Continued portal build for score visualization.
- Began Gateways 2025 poster draft.

Day 5: Poster & Presentation Finalization

- Completed poster, polished portal, and prepped for demo.

Kickoff

Reproducibility Criteria

For each paper, we assess:

- Paper Accessibility
  - Status: [Accessible/Partially Accessible/Not Accessible]
  - Notes: How easy is it to access and interpret the work?
- Code Availability and Usability
  - Status: [Shared and Runnable / Shared with Issues / Not Shared]
  - Notes: Where is it hosted (e.g., GitHub)? Does it work out of the box?
- Data Accessibility
  - Status: [Findable and Usable / Requires Effort / Not Usable]
  - Notes: Formats, licensing, preprocessing steps
- First Code Run Attempt
  - Result: [Success / Partial Success / Failure]
  - Notes: Error messages, manual fixes, missing steps

Github Link

https://github.com/SGX3CodeRunners/RealWorldBugs

Artifact Setup

Fetching HTML Content:
Downloads the raw HTML of a target website using the requests library, ensuring a local copy for processing..

Extracting Paper Data
Parses the downloaded HTML to extract structured information (ID, Title, URLs) about research papers, typically from embedded JavaScript arrays, using regular expressions and JSON parsing.

Downloading PDFs:
Navigates to artifact/DOI URLs, identifies direct PDF links (handling redirects and relative paths), and downloads the PDF files, saving them locally with safe filenames.

Scorecard Draft

```
Found 30 PDF files to analyze.

Analyzing: logan2024megammap.pdf
An error occurred while analyzing pdf/logar
--- Automated Scorecard Summary ---
Paper Availability: Score=1, Notes: PDF was
Availability of Code and Software: Score=2,
Availability of Datasets: Score=1, Notes: F
Computer Requirements: Score=0, Notes:
GPU Requirements: Score=0, Notes:
Documentation Quality: Score=0, Notes:
Ease of Setup: Score=0, Notes:
Reproducibility of Results: Score=0, Notes:
Overall Rating: Score=N/A, Notes: Manual re

Analyzing: Responsibility_in_Context_On_App
An error occurred while analyzing pdf/Respc
--- Automated Scorecard Summary ---
Paper Availability: Score=1, Notes: PDF was
Availability of Code and Software: Score=1,
Availability of Datasets: Score=1, Notes: F
Computer Requirements: Score=0, Notes:
GPU Requirements: Score=0, Notes:
Documentation Quality: Score=0, Notes:
Ease of Setup: Score=0, Notes:
Reproducibility of Results: Score=0, Notes:
Overall Rating: Score=N/A, Notes: Manual r
```

Matrices

Paper Availability

Availability of Datasets

Computer Requirements

GPU Requirements

Documentation Quality

Ease of Setup

Reproducibility of Results

Overall Rating

- **How to score:**
- **4: Fully open access, no barriers.**
- **3: Mostly accessible (e.g., free account required).**
- **2: Paywalled but accessible via institution.**
- **1: Not accessible or access is unclear.**

Reproducibility of Code-Understanding Large Language Models: A 217-Paper Analysis of ICSE '23 & SC '24

Iyana Jones, Copernic Mensah, Aaliyah Lockett, Holy Agyei

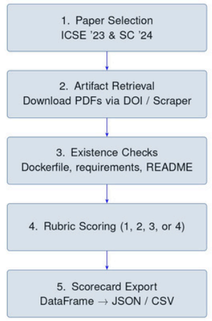
High Performance Computing and Gateways 2025

An Important Problem

Reproducibility is essential to scientific progress: without it, results cannot be validated, built upon, or trusted [6]. In artificial intelligence and high-performance computing, complex software stacks mean that missing containerization, incomplete documentation, or unclear environment specifications can easily undermine workflow credibility and slow down collaboration.

Methodology

To evaluate the reproducibility of HPC/AI research papers in computer science and data science, a structured scorecard was used to assess the content of technical papers, the quality of the documentation and the reproducibility of the environment. This framework is informed by recent reproducibility benchmarks and HPC/AI research challenges [2].



Select papers, fetch contents, check reproducibility files, score via rubric, then export the scorecard.

We employ a detailed, nine-criterion scorecard in which each paper is assessed on a 0–4 scale (0 = not scored) and annotated with free-text notes [5].

1. Paper Availability: Is the paper open access?
2. Code & Software Availability: Is the source hosted publicly, versioned and documented?
3. Dataset Availability: Are datasets clearly shared?
4. Hardware Requirements: Are specifications unreasonable or ambiguous
5. GPU Requirements: Are accelerator needs stated and justified?
6. Documentation Quality: Are README, install guides, and API docs complete?
7. Ease of Setup: Are containers like Docker provided?
8. Reproducibility of Results: Can key metrics be reproduced simply?

These 8 sub scores feed into an overall rating (Excellent, Good, Fair, Poor) alongside a summary note. By combining quantitative scores with contextual notes, our rubric delivers both a transparent metric and the qualitative insight needed to drive improvements across high performance computing and artificial intelligence research.

Results

Analyzing 217 ICSE '23 and SC '24 LLM-for-code papers on a 0–15 reproducibility scale reveals a distinct results: 118 papers (62%) achieved a score of 4 or 15, signaling support via containers, dependency manifests, documentation, or community channels, while 16 papers (8%) scored 0, lacking any reproducibility contents. The remaining 55 papers (29%) scored 12, typically missing only non-essential items like CI/CD badges or discussion forums.

The mean score is 13.60, the median and upper quartile both sit at 15, confirming that best practices are prevalent but not universal. To raise the baseline, we recommend shipping standard Docker templates, explicit dependency manifests, mandatory README setup/validation sections, and active issue-tracking channels.

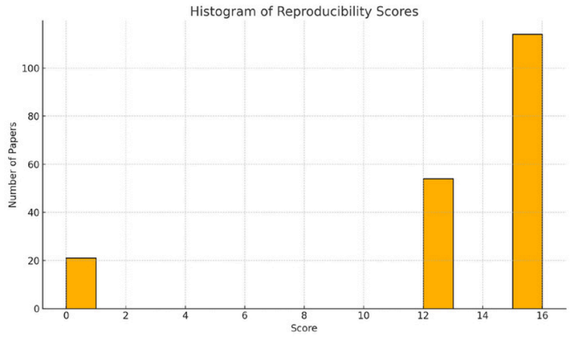


Fig. 1: 0 -> 1 score, 12 - > 3 score, 15 -> 4 score

Future efforts could ensure that every software release is accompanied by a detailed execution environment. Attaching a Docker or Singularity image, ensuring that all dependencies, library versions, and operating-system configurations are included in the code repository will smoothen reproducibility process for collaborators.

Additionally, researchers can provide readable dependency specifications and automated validation workflows. A clearly documented requirements.txt or environment.yml should enumerate each library and its version. Publishing these CI configurations demonstrates that the experiments have been successfully rerun in a standardized context and provides an objective check on reproducibility.

Finally, a reproducibility criteria could be formalized within peer-review and post-publication processes. Conferences and journals can adopt these criteria that score container availability and dependency manifest completeness before acceptance. This transmutes reproducibility within high performance computing research and artificial intelligence into a reality instead of the known 'crisis'.

Cont.

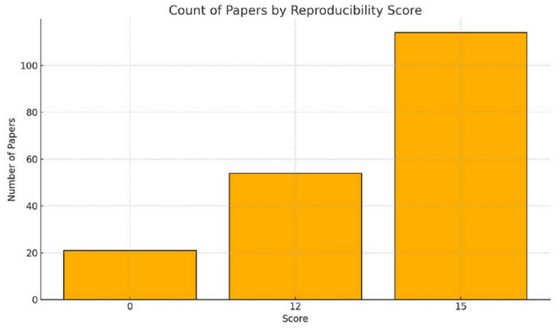


Fig. 2: Distinctive Results

Remarks

Our study demonstrates that while a majority of code understanding large language model papers[4][1] now include essential artifacts, a significant minority still omit container recipes, dependency manifests, or validation pipelines. Elevating reproducibility from an optional bonus to a baseline requirement will require both technical infrastructure and cultural change [3] (artifact evaluation at submission, public scorecards). By publishing our portal and scorecards, we aim to provide transparency, invite corrections and updates, and track progress over time.

Acknowledgments

We gratefully acknowledge the SGX3 ADMI Hackathon organizers and mentors for their guidance and support throughout this project. None of this would have been made possible without the collaborative spirit and resources of the ADMI community.

References

[1] Proceedings of the 45th International Conference on Software Engineering (ICSE 2023). Melbourne, Australia, 2023.

[2] Benjamin A. Antunes and David R. C. Hill. "Reproducibility, Replicability and Repeatability: A Survey of Reproducible Research with a Focus on High Performance Computing". In: *Computer Science Review* (2024), p. 100655.

[3] Association for Computing Machinery (ACM). *Artifact Review and Badging – Version 1.0.* https://www.acm.org/publications/policies/artifact-review-badging. 2017.

[4] Super Computing Conference. *Reproducibility Initiative – SC23.* https://sc23.supercomputing.org/program/papers/reproducibility-initiative/. SC Conference, 2023.

[5] Google. *Google Codelabs.* https://codelabs.developers.google.com/. 2015.

[6] National Academies of Sciences, Engineering, and Medicine. *Reproducibility and Replicability in Science.* Washington, DC: National Academies Press (US), 2019.

WEBSITE live at → https://website-portal-1.onrender.com/

- **Interactive Reproducibility Dashboard:**

Visualizes and compares the reproducibility of LLM research papers from ICSE 2023 and SC24, using a robust 100-point scoring framework.

- **Clean, Intuitive UI:**

Features concise filtering, instant paper lookup, and a detailed scorecard for each paper—making complex reproducibility data easy to explore and present.

- **Fully Open & Deployable:**

Built with Streamlit, the portal is open source, cloud-deployable, and ready for team collaboration or public sharing.