Artisan: Automated Artifact Evaluation with Agents

ANONYMOUS AUTHOR(S)

Artifact evaluation has proven to be an invaluable resource in ensuring the reproduction of software artifacts accompanying research. But, artifact evaluation practiced in its current form is notoriously labor-intensive. To mitigate this, there has been a recent effort to benchmark the performance of LLM agents on reproducing research results. We find several limitations with the existing benchmarks: (1) Existing benchmarks largely formulate the problem as an output prediction problem, which undermines its credibility. (2) Due to such problem formulation, they have limited usefulness: Contrary to human artifact evaluation which adds confidence that the result of papers are reproducible, exercise of benchmarks do not confer similar confidence. (3) They are limited in terms of domain of computer science (ML, NLP) and programming languages (Python, R, Stata).

To mitigate this problem, we present SEAE-Bench, an automated artifact evaluation benchmark in the domain of software engineering. SEAE-Bench has the following advantages: (1) It measures LLM agents' ability to generate a code that reproduces the particular claim of the paper; not only the output but the code behind should be submitted. (2) It thus has two applications: A script submitted by LLM agents can be used as a partial reproduction scripit that reproduces the particular claim of the paper. Also, in the case that there is a transcription or rounding error in the paper itself, it acts as a credible evidence that this is so. (3) It covers diverse software engineering techniques and programming languages.

Informed by the existing agents' struggle with the SEAE-Bench, we also develop Artisan, an agentic approach for the automated artifact evalutaion. Artisan is equipped with many specialized tools to deal with domain-specific problems of artifact evaluation. Our evaluation shows that while most LLM agents struggle with SEAE-Bench, Artisan is the most effective LLM agent by producing TODO partial reproduction scripts, outperforming the baseline by TODO. In addition, during our development of SEAE-Bench, we have uncovered TODO inconsistencies between the paper and the artifact, which demonstrates the strength of our approach in benchmark construction.

CCS Concepts: • **Software and its engineering → Software testing and debugging**.

Additional Key Words and Phrases: Artifact Evaluation, Benchmarks, LLM agents

ACM Reference Format:
Anonymous Author(s). 2025. Artisan: Automated Artifact Evaluation with Agents. 1, 1 (September 2025), 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

This is a sample citation [1].

In summary, this paper contributes the following:

- SERE-Bench, a novel benchmark that evaluates the capabilities of AI agents to reproduce Software Engineering papers using script submission.
- Artisan, an LLM-based approach which incorporates several tools to aid the reproduction of software engineering research artifacts.
- Empirical evaluation which shows that (1) SERE-Bench is challenging for the current state of the art LLM agents and (2) Artisan effectively addresses the common pitfalls faced by LLM agents.

ACM XXXX-XXXX/2025/9-ART
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 Benchmark

2.1 Overview

Table 1. Comparison of prior benchmarks and our benchmark.

	SUPER	CORE-Bench	ReproBench	SERE-Bench (Ours)
Domain	ML, NLP	Computer Science, Social Science, Medicine	Social Science	Software Engineering
Language	Python	Python, R	Python, R, Julia, Stata, MATLAB, Julia, Multiple Languages[1]	Python, R, Julia, C, C++, C#, Go, JS, TS, Perl, Java, Scala, Rust, Ocaml
#Paper	45	90[2]	112	40
#Task	45[3]	270	112	162
Time Limit	30 minutes	45 minutes	2 hours	8 hours
Input	Pre-processed and curated	Pre-processed and curated	Realistic	Realistic
Judging	Accuracy of outputs	Accuracy of outputs	Accuracy of reproducibility scores	Script reproducing the exact numbers
Discovered inconsistensies	0	0	0	3+

Note. [1] As labeled "Multiple Languages" in the source figure. *Note.* [2] 37 for Computer Science *Note.* [3] In terms of Expert set, They additionally provide 152 masked sets and 604 Auto sets.

The notable difference between our work and previous work is that we are script-based. Why is it important? It is immediately useful. It acts as a standalone script that is push-button runnable which reproduces the particular table. It is supported by a concrete execution. What's a good word to capture this? Realistic? Execution-based? Execution-proof? Evidence-accompanying? Comparison with existing work in that they formulate reproduction as a prediction problem but we as a code generation problem? It (loosely) implies that this can be used to correct mistakes made in the paper. Previous works have not pointed out any errors in the paper. We don't need separate ground truth. Numbers in the paper is the ground truth. Less prone to data combination? Or it doesn't matter if it's contaminated because it's useful? What's the argument for this?

2.2 Paper Selection

- Top SE (ICSE, FSE, ASE, ISSTA) with Available and Reusable badges 2024: 114 - Find artifact url - Packaged using Docker: 77 - Exclude Commercial LLM API use Github token (non-public): 67 - Exclude GPU use: 46 - ess than 8 hours per task (when they mention it): 40

2.3 Task Selection

We want our benchmark to be - No cherry-picking - Extensive (100 300) but not too expensive and long-running to run. - Manual validation is desired but should not take too much time.

There are few different options: Run full task set without modification (162) Manually validate all task set (162) to filter out unreproducible and keep the partial table Random sample to smaller size (25 / 50)

I argue for the second option (manually validate all task sets). It is not too much work (1 hour per task) to check whether it is sensible to reproduce the table by running artisan. We have about 127 tables to manually validate and we can sensibly validate them by the end of the ISSTA submission Makes our benchmark higher quality. Less underestimation of agent capability If we manually validate all without bias, no cherry-picking. Is using artisan as a time-saving measurement already a bias?

Randomly sample and manually validate and time budget with time budget.

3 Approach

4 Evaluation

5 Related work

References

[1] Doehyun Baek, Jakob Getz, Yusung Sim, Daniel Lehmann, Ben Titzer, Sukyoung Ryu, and Michael Pradel. 2024. Wasm-R3: Record-Reduce-Replay for Realistic and Standalone WebAssembly Benchmarks. In *Proceedings of the ACM on Programming Languages: Object-Oriented Programming, Systems, Languages & Applications (OOPSLA '24).*