Artisan: Automated Artifact Evaluation with Agents

ANONYMOUS AUTHOR(S)

Artifact evaluation has beenproven 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-Benchhas 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 maintenance tools**.

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), 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Since the pilot of the artifact evaluation at ESEC/FSE 2011, artifact evaluation has become a standard at a software engineering and programming language communities [7]. One of the main goals of artifact evaluation is reproducibility, which is defined as obtainment of the similar experimental outcome with the same experimental setup by the different team [1] Reproducibility is one of the stated goals of artifact evaluation in the Calls for Artifacts and generally regarded as one of the most important goal of the artifact evaluation by the participants. [4]. In addition, there has been studies reporting that artifact evaluation leads to more available, maintained, and documented artifacts [9].

Despite these benefits, artifact evaluation practiced in its current form is notoriously labor-intensive. There has been report that an individual artifact evaluation could take from eight to

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

Table 1. Comparison of prior benchmarks and our benchmark. ML stands for Machine Learning. NLP stands for Natural Language Processing. CS stands for Computer Science. SS stands for Social Science. Med stands for Medicine. SE stands for Software Engineering.

	SUPER [2]	**CORE-Bench** [8]	**Repro-Bench** [6]	**SEAE-Bench (Ours)**
Domain	ML, NLP	CS, SS, Med	SS	SE
#Paper	45	90[1]	112	35
#Task	45[2]	270	112	162
#Programming Languages	1	2	6	10
#Inconsistensies Reported	0	0	0	10
Time Limit	30 minutes	45 minutes	2 hours	8 hours
Judging	Accuracy of outputs	Accuracy of outputs	Accuracy of reproducibility scores	#Script that reproduces the claim

Note. [1] 37 for CS *Note.* [2] In terms of Expert set, They additionally provide 152 masked sets and 604 Auto sets.

ten hours, which is more than the usual time for the paper review [3]. There has been mixed evidence whether papers that underwent artifact evaluation has more visibility, with some studies suggestiing higher visibility [5] and some studies finding no correlation [9]. Considering the significant manual effort, it is unclear whether the benefit for individual researchers as to do artifact evalutaion outweight the cost as both the authors and the users of the artifact.

To mitigate this, there has been a recent effort to benchmark the performance of LLM agents on reproducing research results. Table 1 summarizes these recent efforts. SUPER is a benchmark that evaluates the capability of LLM agents in reproducing results from ML and NLP papers [2]. CORE-Bench, while having similar overall goal, differs by targeting broader domain (Social Science, Medicine) and including vision-language tasks [8]. Repro-Bench focuses on reproduction of social science research, and makes the problem setup more realistic [6].

However, We find several limitations with the existing benchmarks: (1) Existing benchmarks largely formulate the problem as an output prediction problem, which undermines its credibility. SUPER and CORE-Bench only measures the accuracy of the reproduction outcome by the agents; how they obtained the result is not part of the evaluation. This disagrees with the goals of the artifact evaluation as practiced in the software engineering community; it is also potentially more prone to data leakages. Repro-Bench measures the accuracy of the reproducibility scores from one to four compared to the ground truth. While artifact evaluation badge (similar to the reproducibility score used by Repro-Bench) is one of the important side-effect of the artifact evaluation process which signals the quality of the artifact, it is not the main goal of the artifact evaluation. (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. Existing benchmarks thus can be used to measure LLM agents' performance on the formulated task, but they can not augment or replace the current artifact evaluation. (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-Benchhas 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. SEAE-Benchalso distinguishes various fast-path and short-path that LLM agents can take to reproduce the results, which is a crucial distinction not yet highlighted. (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. In the case that the outcome of the submitted script differs from that of the paper, it acts as a credible evidence that paper actually the one with a transcription or rounding error. (3) It covers diverse software engineering techniques and programming languages. It covers diverse software engineering problems (empirical study, fuzzing, static analysis) and diverse real-world Programming Languages (C, C++, Go, JS, Java, Scala, Rust, Ocaml and many more).

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.

In summary, this paper contributes the following:

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

2 Background

2.1 Artifact Evaluation

3 Benchmark

3.1 Overview

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?

3.2 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

3.3 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.

4 Approach

5 Evaluation

6 Discussion

6.1 Threats To Validity

6.2 Limitations

7 Related work

References

[1] Association for Computing Machinery. 2020. *Artifact Review and Badging — Current.* https://www.acm.org/publications/policies/artifact-review-and-badging-current Version 1.1.

[2] Ben Bogin, Kejuan Yang, Shashank Gupta, Kyle Richardson, Erin Bransom, Peter Clark, Ashish Sabharwal, and Tushar Khot. 2024. SUPER: Evaluating Agents on Setting Up and Executing Tasks from Research Repositories. *ArXiv* abs/2409.07440 (2024). https://api.semanticscholar.org/CorpusID:272593197

[3] Ben Hermann, Fabio Massacci, Eric Bodden, and Antonino Sabetta. 2022. What Has Artifact Evaluation Ever Done for Us? *IEEE Security & Privacy* 20 (2022), 96–99. https://api.semanticscholar.org/CorpusID:252224062

[4] Ben Hermann, Stefan Winter, and Janet Siegmund. 2020. Community expectations for research artifacts and evaluation processes. *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (2020). https://api.semanticscholar.org/CorpusID:222270323

[5] Robert Heumüller, Sebastian Nielebock, Jacob Krüger, and Frank Ortmeier. 2020. Publish or perish, but do not forget your software artifacts. *Empirical Software Engineering* 25 (2020), 4585 – 4616. https://api.semanticscholar.org/CorpusID:220070385

[6] Chuxuan Hu, Liyun Zhang, Yeji Lim, Aum Wadhwani, Austin Peters, and Daniel Kang. 2025. REPRO-Bench: Can Agentic AI Systems Assess the Reproducibility of Social Science Research?. In *Annual Meeting of the Association for Computational Linguistics*. https://api.semanticscholar.org/CorpusID:280045482

[7] Shriram Krishnamurthi and Jan Vitek. 2015. The real software crisis. *Commun. ACM* 58 (2015), 34 – 36. https://api.semanticscholar.org/CorpusID:1371568

[8] Zachary S. Siegel, Sayash Kapoor, Nitya Nagdir, Benedikt Stroebl, and Arvind Narayanan. 2024. CORE-Bench: Fostering the Credibility of Published Research Through a Computational Reproducibility Agent Benchmark. *Trans. Mach. Learn. Res.* 2024 (2024). https://api.semanticscholar.org/CorpusID:272694423

[9] Stefan Winter, Christopher Steven Timperley, Ben Hermann, Jürgen Cito, Jonathan Bell, Michael C Hilton, and Dirk Beyer. 2022. A retrospective study of one decade of artifact evaluations. *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (2022). https://api.semanticscholar.org/CorpusID:253421604