Performance Degradation of Codestral and DeepSeek R1 on LiveCodeBench Under Time Contamination

Assignee Research

June 1, 2026

Abstract

This report synthesises findings from 11 peer-reviewed papers addressing the following research question: How does the performance degradation of Codestral and DeepSeek R1 on LiveCodeBench compare when evaluated on time-contaminated versus contamination-free coding problems. Large Language Models (LLMs) applied to code-related applications have emerged as a prominent field, attracting significant interest from both academia and industry. However, as new and improved LLMs are developed, existing evaluation benchmarks (e.g., HumanEval, MBPP) are no. 16 claims were extracted from source literature; 0 were independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 4.0/10. This report is a machine-generated literature synthesis and does not constitute original research.

1 Introduction

This paper examines: LiveCodeBench: Holistic and Contamination Free Evaluation of Large Language Models for Code. Research question: How does the performance degradation of Codestral and DeepSeek R1 on LiveCodeBench compare when evaluated on time-contaminated versus contamination-free coding problems?.

2 Methodology

Systematic literature search across multiple databases yielded 11 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 4.0/10.

3 Results

11 papers retrieved. 16 claims extracted; 0 independently verified. Quality review score: 4.0/10.

4 Limitations

This report is a machine-generated literature synthesis and does not constitute original research. Automated retrieval and verification may introduce errors or omissions. Review scores reflect automated assessment, not human peer review. Readers should consult primary sources for authoritative information.

5 Extracted Claims

Claim	Verified	Confidence
LiveCodeBench curates problems from three coding competition websites: LeetCode, AtCoder, and CodeForces.	×	0.14
The problems consist of a natural language problem statement along with example input-output examples.	×	0.03
The goal is to write a program that passes a set of hidden tests.	×	0.01
Thousands of participants participate, solving these problems thus ensuring that the problems are vetted for clarity and	×	0.02
HTML scrapers were written for each of the websites to collect problems and the corresponding metadata.	×	0.01
Problems with images are excluded to ensure quality and consistency.	×	0.02
Problems that are not suitable for grading by input-output examples, such as those that accept multiple correct answers	×	0.02
For each problem, tuples of natural language problem statement P, test cases T, and ground truth solution S are collecte	×	0.03
The contest date D is associated to mark the release date of each problem.	×	0.01
The release date allows measuring the performance of LLMs over different time windows by filtering problems based on whe	×	0.05
A UI has been developed that allows comparing models on problems released during different time windows.	×	0.04
Tests are crucial for assessing the correctness of the generated outputs and are used in all four scenarios.	×	0.03
Tests available on platform websites are collected whenever possible and used for the benchmark.	×	0.02
If tests are not available, a LLM (GPT-4-Turbo) is used to generate tests for the problems.	×	0.03
Instead of generating inputs directly using the LLM, input generators are constructed that sample inputs based on the pr	×	0.05
A small fraction of failing tests are collected from the platform for the more recent problems allowing more directed ad	×	0.01

References

- http://arxiv.org/abs/2403.07974v2
- http://arxiv.org/abs/2501.18576v1
- http://arxiv.org/abs/2504.00016v1