# Open-Source vs. Proprietary Language Models on Coding Benchmarks V9

Assignee Research

June 6, 2026

**Abstract**

This report synthesises findings from 14 peer-reviewed papers addressing the following research question: What is the comparative performance of open-source language models versus proprietary models on coding benchmarks v9. 13 claims were extracted from source literature; 1 was independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 3.4/10. This report is a machine-generated literature synthesis and does not constitute original research.

## 1 Introduction

This paper examines: Assessing Small Language Models for Code Generation: An Empirical Study with Benchmarks. Research question: What is the comparative performance of open-source language models versus proprietary models on coding benchmarks v9.

## 2 Methodology

Systematic literature search across multiple databases yielded 14 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 3.4/10.

## 3 Results

14 papers retrieved. 13 claims extracted; 1 independently verified. Quality review score: 3.4/10.

## 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 |
|---|---|---|
| Existing studies mostly focus on general natural-language processing tasks, leaving the effectiveness of SLMs for specia | × | 0.06 |
| The absence of cross-programming-language evaluations limits understanding of whether SLM performance and efficiency var | × | 0.08 |
| Prior work does not systematically examine task-specific trade-offs between code-generation quality and computational ef | × | 0.11 |
| The study focuses on three key aspects: (i) performance of SLMs on code generation, (ii) how they balance computational | ✓ | 0.17 |
| The research questions include RQ1: How do different small language models perform on code generation tasks across estab | × | 0.12 |
| The benchmarks used include HumanEval, MBPP, Mercury, CodeXGLUE (C2T), and HumanEvalPack. | × | 0.10 |
| The metrics used include pass@k and BLEU. | × | 0.01 |
| The models evaluated include PolyCoder 0.4B, InCoder 1.3B, Phi1 1.3B, DeepSeek-Coder 1.3B, OpenCodeInterpreter 1.3B, Yi- | × | 0.03 |
| The evaluation parameters include Temperature 0.2, Top-k 0 (disabled), Top-p 0.95, Max generation length (512 tokens def | × | 0.04 |
| The performance metrics for PolyCoder 0.4B on HumanEval are pass@1: 0.03, pass@2: 0.04, pass@5: 0.04, pass@10: 0.04. | × | 0.03 |
| The performance metrics for InCoder 1.3B on HumanEval are pass@1: 0.09, pass@2: 0.11, pass@5: 0.13, pass@10: 0.15. | × | 0.02 |
| The performance metrics for Phi1 1.3B on HumanEval are pass@1: 0.52, pass@2: 0.56, pass@5: 0.60, pass@10: 0.61. | × | 0.02 |
| The performance metrics for DeepSeek-Coder 1.3B on HumanEval are pass@1: 0.32, pass@2: 0.36, pass@5: 0.42, pass@10: 0.43 | × | 0.02 |

## References


- http://arxiv.org/abs/2507.03160v4
- http://arxiv.org/abs/2403.03788v1
- http://arxiv.org/abs/2409.11272v7