Scaling Inference Efficiency of Code Generation Models on Multilingual CodeMixBench

Assignee Research

May 30, 2026

Abstract

This report synthesises findings from 15 peer-reviewed papers addressing the following research question: How do different code generation models scale in inference efficiency when evaluated on multilingual programming benchmarks like CodeMixBench. Large Language Models (LLMs) have achieved remarkable success in code generation tasks, powering various applications like code completion, debugging, and programming assistance. However, existing benchmarks such as HumanEval, MBPP, and BigCodeBench primarily evaluate LLMs on. 10 claims were extracted from source literature; 1 was independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 4.2/10. This report is a machine-generated literature synthesis and does not constitute original research.

1 Introduction

This paper examines: CodeMixBench: Evaluating Large Language Models on Code Generation with Code-Mixed Prompts. Research question: How do different code generation models scale in inference efficiency when evaluated on multilingual programming benchmarks like CodeMixBench?.

2 Methodology

Systematic literature search across multiple databases yielded 15 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 4.2/10.

3 Results

15 papers retrieved. 10 claims extracted; 1 independently verified. Quality review score: 4.2/10.

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
CodeMixBench is constructed as an augmentation of BigCodeBench.	×	0.06
BigCodeBench comprises two splits: complete split and instruct split.	×	0.03
CodeMixBench supports three realistic code-mixing configurations: Hinglish (Hindi-English), Spanish-English, and Chinese	✓	0.21
Code-mixing in CodeMixBench is applied only to the prompt instructions and docstrings, leaving code unchanged.	×	0.06
The GAME score is an embedding-based metric designed for code-mixed evaluation that uses cosine similarity to compare se	×	0.06
Pass@k is the standard metric for evaluating functional correctness of generated code.	×	0.04
BLEU is poorly suited for evaluating code-mixed prompts due to lexical variability and absence of gold-standard referenc	×	0.11
CodeMixBench uses a two-stage augmentation strategy involving translation and controlled code-mixing.	×	0.07
The CMD parameter controls the proportion of embedded English tokens retained during code-mixing.	×	0.06
The implementation pipeline produces a structured dictionary mapping English $\rightarrow$ Matrix Language $\rightarrow$ Romanized Forms.	×	0.03

References

- http://arxiv.org/abs/2412.15453v1

- http://arxiv.org/abs/2505.05063v1
- http://arxiv.org/abs/2303.12869v1