Scaling Efficiency of RAG and Parametric Models in MBPP Code Generation Tasks

Assignee Research

June 2, 2026

Abstract

This report synthesises findings from 16 peer-reviewed papers addressing the following research question: How does the computational efficiency of RAG-based models scale with increasing code generation task complexity in the MBPP benchmark compared to parametric-only models measured in tokens per second. 14 claims were extracted from source literature; 0 were independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 3.3/10. This report is a machine-generated literature synthesis and does not constitute original research.

1 Introduction

This paper examines: EVOR: Evolving Retrieval for Code Generation. Research question: How does the computational efficiency of RAG-based models scale with increasing code generation task complexity in the MBPP benchmark compared to parametric-only models measured in tokens per second?.

2 Methodology

Systematic literature search across multiple databases yielded 16 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 3.3/10.

3 Results

16 papers retrieved. 14 claims extracted; 0 independently verified. Quality review score: 3.3/10.

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
With CodeLlama, the improvements of MPSC, ExeDec, and Reflexion on EVOR-BENCH are smaller than 2% on average compared to	×	0.07
The execution accuracy of MPSC, ExeDec, and Reflexion remains 0 on the Ring subset of EVOR-BENCH when using CodeLlama.	×	0.07
DocPrompting significantly surpasses MPSC, ExeDec, and Reflexion on EVOR-BENCH by explicitly using documentation.	×	0.05
EVOR achieves a 16.1% absolute gain over DocPrompting when using ChatGPT on EVOR-BENCH.	×	0.08
EVOR achieves a 16.2% absolute gain over DocPrompting when using CodeLlama on EVOR-BENCH.	×	0.08
DocPrompting uses documentation as a single retrieval source without evolution in both queries and knowledge.	×	0.13
The paper uses execution accuracy (pass@1) as the default metric for evaluation.	×	0.03
The Vanilla baseline generates outputs directly from LLMs based on the coding question without augmenting external knowl	×	0.05
MPSC prompts LLMs to generate diverse outputs from three perspectives: Solution, Specification, and Test case.	×	0.05
MPSC constructs a 3-partite graph and picks the optimal choice of solutions based on confidence scores.	×	0.03
ExeDec employs a subgoal model to predict the subgoal of the desired program state for the next part of the program.	×	0.03
ExeDec uses a synthesizer model to generate the corresponding subprogram to achieve the predicted subgoal.	×	0.02
Retrieval-augmented code generation introduces risks of biased or incorrect information being retrieved, which could pro	×	0.10
Retrieval-augmented code generation poses privacy and security concerns if sensitive code snippets are inadvertently inc	×	0.10

References

- http://arxiv.org/abs/2509.25716v1
- http://arxiv.org/abs/2402.12317v2
- http://arxiv.org/abs/2501.03569v1