What is the sensitivity of LLaMA 3.2 and Mistral to varying context window sizes when repairing code, as measu

Assignee Research

June 10, 2026

Abstract

Large language models have shown remarkable aptitude in code generation, but still struggle to perform complex tasks. Self-repair – in which the model debugs and repairs its own code – has recently become a popular way to boost performance in these settings. However, despite its increasing popularity, existing studies of self-repair have been limited in scope; in many settings, its efficacy thus remains poorly understood. In this paper, we analyze Code Llama, GPT-3.5 and GPT-4's ability to perform self-repair on problems taken from HumanEval and APPS. We find that when the cost of carrying o

1 Introduction

This paper examines: Is Self-Repair a Silver Bullet for Code Generation?. Research question: What is the sensitivity of LLaMA 3.2 and Mistral to varying context window sizes when repairing code, as measured by pass@k scores on CodeGen and GPT-4 benchmarks?.

2 Methodology

Systematic literature search across multiple databases yielded 13 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 3.8/10.

3 Results

13 papers retrieved. 14 claims extracted; 0 independently verified. Quality review score: 3.8/10.

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
Self-repair is evaluated for GPT-3.5, GPT-4, and CodeLlama-13b-instruct models.	×	0.11
Python programming challenges from APPS and HumanEval datasets are used for evaluation.	×	0.04
For APPS, experiments are conducted on a randomly chosen set of 300 tasks.	×	0.04
Self-repair is implemented using templated string concatenation with one-shot prompting.	×	0.06
Decoding temperature is set to 0.8 for all models based on preliminary experiments.	×	0.05
Self-repair is compared against a baseline without repair, which is i.i.d. sampling from the corresponding model.	×	0.08
Self-repair is not a silver bullet but improves with diverse initial samples.	×	0.13
Self-repair performance is evaluated by varying the number of initial i.i.d. base samples (np) and joint feedback, repai	×	0.07
Figures 4 and 3 show the results for CodeLlama and GPT-3.5 on HumanEval, and GPT-3.5 and GPT-4 on APPS, respectively.	×	0.07
Self-repair achieves the same pass rate as the baseline when the normalized mean pass rate is 1.	×	0.06
Self-repair involves four stages: code generation, code execution, feedback generation, and code repair.	×	0.12
Code generation involves generating np samples i.i.d. from a programming model MP given a specification ψ.	×	0.06
Code execution involves executing the np code samples against a test bed and collecting error messages if any sample fai	×	0.04
Feedback generation involves using a feedback model MF to produce detailed explanations of what went wrong for each wron	×	0.05

References

- `http://arxiv.org/abs/2310.06825v1`

- http://arxiv.org/abs/2306.09896v5
- http://arxiv.org/abs/2411.07586v1