Llama3 and Codestral Trade-offs in Adversarial Robust Vulnerability Scanning

Assignee Research

June 4, 2026

Abstract

This report synthesises findings from 2 peer-reviewed papers addressing the following research question: How do inference efficiency and detection accuracy trade-offs differ between Llama3 and Codestral when fine-tuned for adversarial robustness in C++ and Python vulnerability scanning. 11 claims were extracted from source literature; 11 were independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 8.3/10. This report is a machine-generated literature synthesis and does not constitute original research.

1 Introduction

This paper examines: Prompt-Driven Code Summarization: A Systematic Literature Review. Research question: How do inference efficiency and detection accuracy trade-offs differ between Llama3 and Codestral when fine-tuned for adversarial robustness in C++ and Python vulnerability scanning?.

2 Methodology

Systematic literature search across multiple databases yielded 2 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 8.3/10.

3 Results

2 papers retrieved. 11 claims extracted; 11 independently verified. Quality review score: 8.3/10.

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
Software documentation is essential for program comprehension, developer onboarding, code review, and long-term maintena	✓	0.30
Producing quality documentation manually is time-consuming and frequently yields incomplete or inconsistent results.	✓	0.27
Large language models (LLMs) offer a promising solution by automatically generating natural language descriptions from s	✓	0.30
LLMs help developers understand code more efficiently, facilitate maintenance, and support downstream activities such as	✓	0.22
The effectiveness of LLMs in documentation tasks critically depends on how they are prompted.	✓	0.22
Properly structured instructions can substantially improve model performance.	✓	0.21
Prompt engineering is a foundational technique in LLM-based software engineering.	✓	0.22
Approaches such as few-shot prompting, chain-of-thought reasoning, retrieval-augmented generation, and zero-shot learnin	✓	0.32
Current research on prompting strategies for code summarization remains fragmented.	✓	0.19
There is limited understanding of which prompting strategies work best, for which models, and under what conditions.	✓	0.22
Evaluation practices for code summarization vary widely, with most studies relying on overlap-based metrics that may not	✓	0.28

References

- https://doi.org/10.48550/arxiv.2407.08924
- https://openalex.org/W7155245368