Impact of Quantization on SLM Inference Throughput and Accuracy for Python CWE Detection Against CodeBERTa

Assignee Research

June 12, 2026

Abstract

Large Language Models (LLMs) have demonstrated significant capabilities in understanding and analyzing code for security vulnerabilities, such as Common Weakness Enumerations (CWEs). However, their reliance on cloud infrastructure and substantial computational requirements pose challenges for analyzing sensitive or proprietary codebases due to privacy concerns and inference costs. This work explores the potential of Small Language Models (SLMs) as a viable alternative for accurate, on-premise vulnerability detection. We investigated whether a 350-million parameter pre-trained code model (codeg

1 Introduction

This paper examines: Case Study: Fine-tuning Small Language Models for Accurate and Private CWE Detection in Python Code. Research question: What is the impact of quantization techniques on the inference throughput of SLMs for CWE detection in Python, and how does this compare to the accuracy loss when benchmarked against the CodeBERTa dataset?.

2 Methodology

Systematic literature search across multiple databases yielded 12 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 6.9/10.

3 Results

12 papers retrieved. 12 claims extracted; 12 independently verified. Quality review score: 6.9/10.

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
Farasat & Posegga [15] achieved an average accuracy of 98.6%, F1 score of 94.7%, precision of 96.2%, recall of 93.3%, an	✓	0.15
Bagheri et al. [31] reported an F1 score of 99% using a hybrid model combining self-attention and CNN (Conformer).	✓	0.18
Singh et al. [32] achieved an accuracy of 0.66, precision of 0.65, recall of 0.66, and F1 score of 0.64 for CWE predicti	✓	0.16
Dozono et al. [6] reported Python F1 scores for various LLMs: GPT-4o=0.80, GPT-4T=0.76, Gemini 1.5 Pro=0.75, CodeLlama-7	✓	0.32
Steenhoek et al. [34] evaluated LLMs on C/C++ and found a low balanced accuracy of 54.5%.	✓	0.17
Shestov et al. [35] achieved the best F1 score of 0.86 for binary classification using WizardCoder for JAVA CWE detectio	✓	0.23
Li et al. [36] reported a 5-6% improvement over the base model using LoRa and IA3 fine-tuning approaches for LLMs.	✓	0.20
Jiang et al. [37] achieved the best F1 score of 87% using the Llama 2-7b model with LoRa fine-tuning.	✓	0.21
The un-tuned codegen-mono model failed to detect a single CWE within any of the code snippets presented in a baseline ev	✓	0.31
After instruction-following fine-tuning, the codegen-mono model achieved an accuracy of 99%, precision of $\approx$98.08%, recal	✓	0.42
The MITRE Top 25 Most Dangerous Software Weaknesses list [30] was selected as the target for the detection model.	✓	0.21
Google's gemini-2.0-flash-thinking-exp-01-21 model was used via its API for generating synthetic data for the dataset.	✓	0.17

References

- http://arxiv.org/abs/2504.16584v1

- http://arxiv.org/abs/2506.22486v1
- http://arxiv.org/abs/2603.03203v3