Real-Time Vulnerability Classification Trade-offs in CodeT5 and State-of-the-Art Code Models

Assignee Research

June 1, 2026

Abstract

This report synthesises findings from 16 peer-reviewed papers addressing the following research question: How does the trade-off between real-time vulnerability classification accuracy and throughput compare between CodeT5 models and other state-of-the-art code language models (e.g., CodeGen, CodeGPT). Many ML-based approaches have been proposed to automatically detect, localize, and repair software vulnerabilities. While ML-based methods are more effective than program analysis-based vulnerability analysis tools, few have been integrated into modern IDEs, hindering practical. 15 claims were extracted from source literature; 0 were independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 2.0/10. This report is a machine-generated literature synthesis and does not constitute original research.

1 Introduction

This paper examines: AIBugHunter: A Practical Tool for Predicting, Classifying and Repairing Software Vulnerabilities. Research question: How does the trade-off between real-time vulnerability classification accuracy and throughput compare between CodeT5 models and other state-of-the-art code language models (e.g., CodeGen, CodeGPT) when deployed as Visual Studio Code extensions?.

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 2.0/10.

3 Results

16 papers retrieved. 15 claims extracted; 0 independently verified. Quality review score: 2.0/10.

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
The experiments were conducted using an NVIDIA GTX 3090 GPU with 24GB of memory.	×	0.02
The proposed approach achieved 74% multiclass accuracy for predicting CWE-IDs.	×	0.04
CodeBERT achieved 71% multiclass accuracy for predicting CWE-IDs.	×	0.02
BERT-base achieved 72% multiclass accuracy for predicting CWE-IDs.	×	0.03
The BoW+RF baseline achieved 52% multiclass accuracy for predicting CWE-IDs.	×	0.02
The Naive Bayes baseline achieved 51% multiclass accuracy for predicting CWE-IDs.	×	0.02
The proposed approach achieved 65% multiclass accuracy for predicting CWE-Types.	×	0.05
BERT-base achieved 59% multiclass accuracy for predicting CWE-Types.	×	0.03
CodeBERT achieved 50% multiclass accuracy for predicting CWE-Types.	×	0.02
The BoW+RF baseline achieved 27% multiclass accuracy for predicting CWE-Types.	×	0.03
The Naive Bayes baseline achieved 27% multiclass accuracy for predicting CWE-Types.	×	0.03
BERT-base is a BERT model pre-trained on natural language.	×	0.07
CodeBERT is a BERT model pre-trained on programming language.	×	0.07
The BoW+RF baseline uses bag of words as features together with a Random Forest model.	×	0.04
The BoW+NB baseline uses bag of words as features together with a Naive Bayes model.	×	0.02

References

- http://arxiv.org/abs/2311.03365v1
- http://arxiv.org/abs/2504.16584v1
- http://arxiv.org/abs/2305.16615v1