Pretraining Language Impact on CodeT5 Robustness to Software Version Shifts

Assignee Research

June 9, 2026

Abstract

This report synthesises findings from 11 peer-reviewed papers addressing the following research question: To what extent does pretraining on logic-based languages versus functional languages improve CodeT5's robustness against software version shifts in the PPTC-R benchmark. 9 claims were extracted from source literature; 0 were independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 3.7/10. This report is a machine-generated literature synthesis and does not constitute original research.

1 Introduction

This paper examines: Imandra CodeLogician: Neuro-Symbolic Reasoning for Precise Analysis of Software Logic. Research question: To what extent does pretraining on logic-based languages versus functional languages improve CodeT5's robustness against software version shifts in the PPTC-R benchmark?.

2 Methodology

Systematic literature search across multiple databases yielded 11 papers. Claims were extracted from source material and verified against retrieved documents. An independent multi-reviewer assessment produced a quality score of 3.7/10.

3 Results

11 papers retrieved. 9 claims extracted; 0 independently verified. Quality review score: 3.7/10.

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
Imandra CodeLogician uses an iterative refinement process for autoformalization to produce models admissible to ImandraX	×	0.07
Admitted models can be verified but may contain unresolved abstractions.	×	0.01
Executable models support full behavioral analysis and test generation.	×	0.03
CodeLogician enables rigorous reasoning about systems whose direct analysis in the source language would be impractical.	×	0.07
CodeLogician adopts a notion of correctness based on functional equivalence at the chosen abstraction level.	×	0.05
Two programs are considered equivalent if they produce the same observable outputs for the same inputs within the modele	×	0.02
Exhaustive behavioral analysis plays a central role in assessing model correctness.	×	0.03
CodeLogician provides a structured and auditable view of program behavior by enumerating all distinct behavioral regions	×	0.04
CodeLogician makes mismatches between intended and modeled semantics explicit and actionable.	×	0.03

References

- http://arxiv.org/abs/1402.0087v1
- http://arxiv.org/abs/2601.11840v2
- http://arxiv.org/abs/2209.02422v3