Taxonomy of AI Techniques for Solving Competition-Level Software Engineering Problems

Assignee Research

June 7, 2026

Abstract

This report synthesises findings from 16 peer-reviewed papers addressing the following research question: What techniques enable language models to solve competition-level software engineering problems v18. 14 claims were extracted from source literature; 1 was independently verified against retrieved documents. An automated multi-reviewer quality assessment produced a score of 4.5/10. This report is a machine-generated literature synthesis and does not constitute original research.

1 Introduction

This paper examines: A Comprehensive Survey of AI-Driven Advancements and Techniques in Automated Program Repair and Code Generation. Research question: What techniques enable language models to solve competition-level software engineering problems v18.

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 4.5/10.

3 Results

16 papers retrieved. 14 claims extracted; 1 independently verified. Quality review score: 4.5/10.

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 paper 'Report on Timing Side-Channel Mitigation via Automated Program Repair' targets the C programming language and	×	0.08
The paper 'Report on Greybox Fuzzing for Concurrency Testing' targets C and C++ programming languages and uses SCTBench	×	0.04
The paper 'Coevolution of Patches and Tests in Automated Program Repair' targets the Java programming language and uses	×	0.08
The paper 'ProveNFix: Temporal-Guided Program Repair' targets the OCaml programming language and uses a custom benchmark	×	0.05
The paper 'Program Repair by Fuzzing over Patch and Input Space' targets the C programming language and uses the VulnLoc	×	0.05
The paper 'LLM-guided Protocol Fuzzing' targets the C programming language and uses the ProFuzzBench benchmark.	×	0.03
The paper 'DEAR' targets the Java programming language and uses the Defects4J benchmark.	×	0.06
The paper 'Transfer' targets the Java programming language and uses the Defects4J benchmark.	×	0.06
The paper 'SPT Code' targets Java, Python, JavaScript, PHP, Go, and Ruby programming languages and uses BLEU, METEOR, an	×	0.05
The methodology includes Systematic Literature Review, Taxonomy Development, Comparative Analysis, and Trend Analysis an	×	0.04
Systematic Literature Review involves searching relevant papers, applying inclusion and exclusion criteria, and conducti	×	0.03
Taxonomy Development involves creating a classification of AI techniques, tools, and methods used for debugging, bug fix	✓	0.16
Comparative Analysis involves comparing selected papers using evaluation criteria such as performance and accuracy, and	×	0.03
Trend Analysis and Gap Identification aims to identify open challenges and research gaps in the field.	×	0.08

References

- http://arxiv.org/abs/2602.04449v1
- http://arxiv.org/abs/2505.23419v2
- http://arxiv.org/abs/2411.07586v1