UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

itee PhD
information technology
electrical engineering

Cristina Improta

Assessing and Enhancing Code Quality in the Era of AI-Developed Software

Tutor: Domenico Cotroneo

Cycle: XXXVIII

Year: 3rd

itee PhD
information technology
electrical engineering

Candidate's information

- **MSc degree** in Computer Engineering
- **Research group**: DEpendable and Secure Software Engineering and Real-Time systems (DESSERT)
- **PhD duration**: 1 November 2022 – 31 October 2025
- **Scholarship type**: UNINA
- **Periods abroad**: 11 days + 3 months in Università della Svizzera italiana (USI), Lugano, Switzerland (02/03/2024-12/03/2024 and 02/05/2024-31/07/2024), under the supervision of Prof. Gabriele Bavota.

Summary of study activities

PhD Courses:

- Strategic Orientation for STEM Research & Writing
- Using Deep Learning Properly
- Virtualization technologies and their applications
- IoT Data Analysis
- Innovation and Entrepreneurship
- AI Code Generation: Foundations, Evaluation, and Security

PhD Schools:

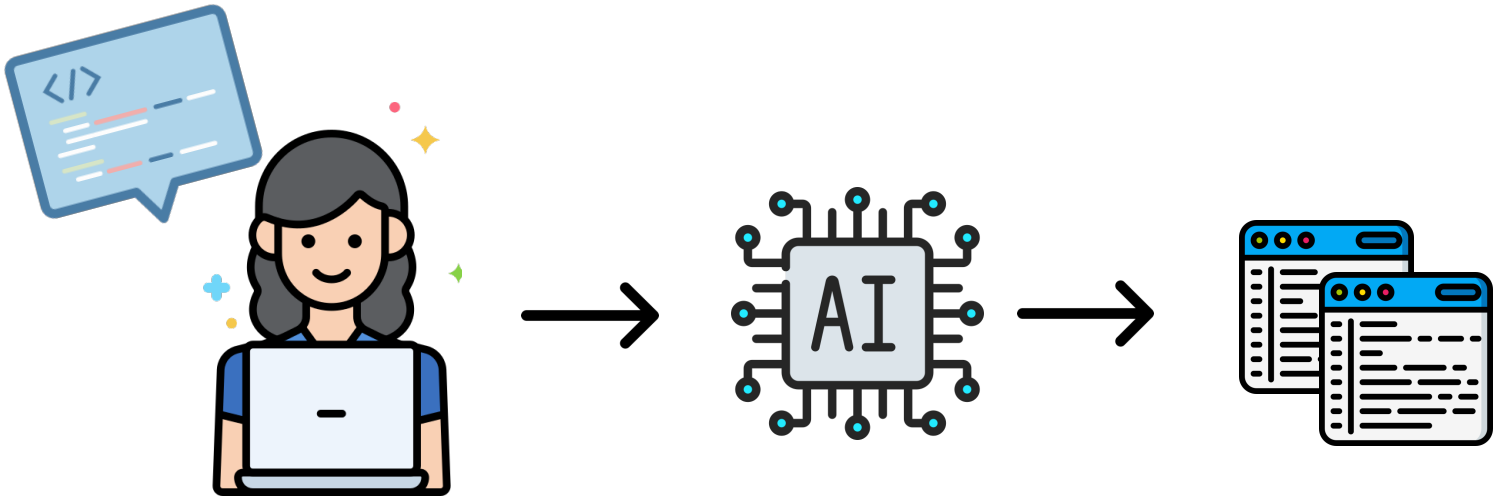
- 2023 Spring School on Transferable Skills
- 3rd International Software Engineering Summer School (SIESTA23)

Conferences / events attended

- 34th IEEE International Symposium on Software Reliability Engineering Conference (ISSRE 2023). October 9-12 2023, Florence, Italy. *Presenting author*.
- 32nd IEEE/ACM International Conference on Program Comprehension (ICPC24), 15-16 April, Lisbon, Portugal. *Presenting author*
- 46th International Conference on Software Engineering (ICSE24), 17-19 April, Lisbon, Portugal.
- 35th IEEE International Symposium on Software Reliability Engineering (ISSRE24). 28-31 October, Tsukuba, Japan. *Presenting author*
- 36th IEEE International Symposium on Software Reliability Engineering (ISSRE25), 21-24 October, Sao Paulo, Brazil. *Presenting author*

Research area(s)

AI-based code generation, with a focus on two main dimensions: (i) its application to **offensive security**, including the generation, validation, and robustness of AI-generated exploits; and (ii) the evaluation and improvement of **software quality** in AI-generated code



Research results

- A framework for assessing the syntactical and semantic correctness of AI-generated security exploits
- A data augmentation strategy to perturb NL code descriptions and improve model robustness
- A prompt-engineering solution to leverage additional contextual information to improve model performance

- A training data cleaning pipeline to improve quality of AI-generated code.
- A standardized evaluation methodology based on ODC and CWE taxonomies to ensure consistent quality assessment
- Large-scale comparisons of human vs. AI code to reveal differences in defects, vulnerabilities and complexity

PhD Thesis

Research products

[C1]	P. Liguori, C. Improta, S. De Vivo, R. Natella, B. Cukic, D. Cotroneo. *Can NMT Understand Me? Towards Perturbation-based Evaluation of NMT Models for Code Generation.* **IEEE/ACM 1st International Workshop on Natural Language-Based Software Engineering (NLBSE)**, Pittsburgh, PA, USA, May 2022, pp. 59-66, IEEE Computer Society.
[J1]	P. Liguori, C. Improta, R. Natella, B. Cukic, D. Cotroneo. *Who evaluates the evaluators? On automatic metrics for assessing AI-based offensive code generators.* **Expert Systems with Applications Journal (ESWA)**, vol. 225, pp. 120073, 2023.
[C2]	C. Improta. *Poisoning Programs by Un-Repairing Code: Security Concerns of AI-generated Code.* **1st IEEE International Workshop on Reliable and Secure AI for Software Engineering (ReSAISE23), ISSREW23**, Florence, Italy, Oct. 2023, pp. 128-131, IEEE.
[J2]	R. Natella, P. Liguori, C. Improta, B. Cukic, D. Cotroneo, *AI Code Generators for Security: Friend or Foe?,* **IEEE Security & Privacy**, vol. 22(5), pp. 73-81, 2024.
[C3]	D. Cotroneo, C. Improta, P. Liguori, R. Natella, *Vulnerabilities in AI Code Generators: Exploring Targeted Data Poisoning Attacks,* **32nd IEEE/ACM International Conference on Program Comprehension (ICPC24)** Lisbon, Portugal, Apr. 2024, pp. 280-292, IEEE Computer Society,
[J3]	D. Cotroneo, A. Foggia, C. Improta, P. Liguori, R. Natella, *Automating the correctness assessment of AI-generated code for security contexts,* **Journal of Systems and Software,** vol. 216, pp. 112113, 2024.

Research products

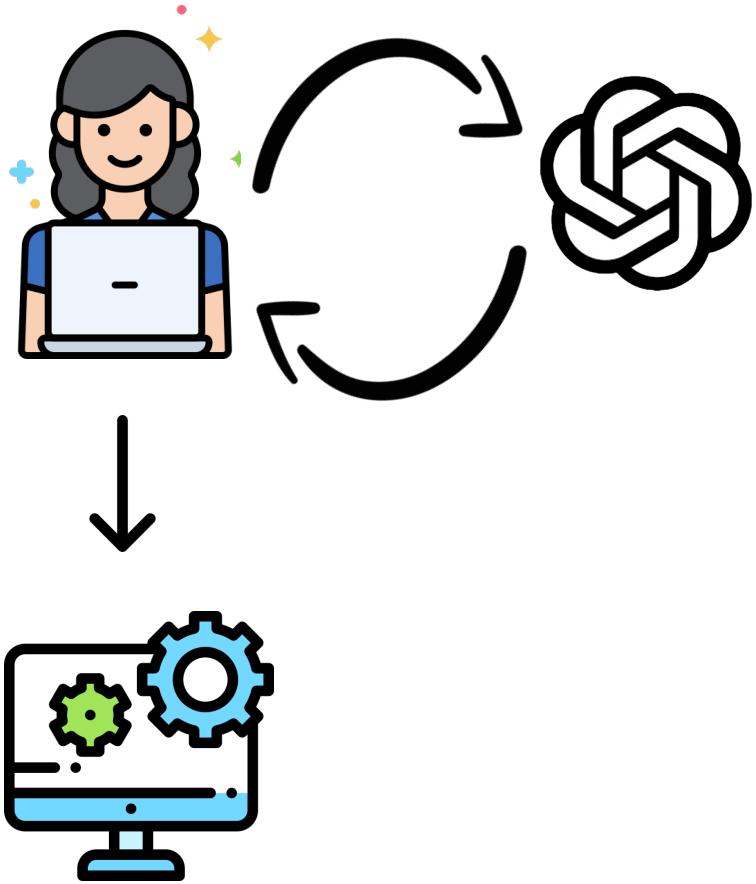
[J4]	C. Improta, P. Liguori, R. Natella, B. Cukic, D. Cotroneo, *Enhancing Robustness of AI Offensive Code Generators via Data Augmentation,* **Empirical Software Engineering (EMSE) Journal**, vol. 30(1), pp. 7, 2025.
[C4]	P. Liguori, C. Improta, R. Natella, B. Cukic, D. Cotroneo, *Enhancing AI-based Generation of Software Exploits with Contextual Information,* **35th IEEE International Symposium on Software Reliability Engineering (ISSRE24)** Tsukuba, Japan, Oct. 2024, pp. 180-191, IEEE
[C5]	C. Improta, R. Tufano, P. Liguori, D. Cotroneo, G. Bavota, *Quality In, Quality Out: Investigating Training Data's Role in AI Code Generation*, **33rd IEEE/ACM International Conference on Program Comprehension (ICPC25),** Ottawa, ON, Canada, Apr. 2025, pp. 454-465, IEEE Computer Society.
[C6]	D. Cotroneo, C. Improta, P. Liguori, *Human-Written vs. AI-Generated Code: A Large-Scale Study of Defects, Vulnerabilities, and Complexity,* **36th IEEE International Symposium on Software Reliability Engineering (ISSRE),** Sao Paulo, Brazil, Oct. 2025. Status: Accepted, to appear.
[C7]	C. Improta, *Detecting Stealthy Data Poisoning Attacks in AI Code Generators,* **3rd IEEE International Workshop on Reliable and Secure AI for Software Engineering (ReSAISE),** Sao Paulo, Brazil, Oct. 2025. Status: Accepted, to appear.
[J5]	C. Improta, P. Liguori, R. Natella, B. Cukic, & D. Cotroneo. *Reading between the Lines: Context-Aware AI-based Generation of Software Exploits* **Empirical Software Engineering.** Status: Under review after revision.

PhD thesis overview

- **Motivation**
  - AI-developed software is nowdays a reality, with developers' role is shifting from implementers to reviewers of AI-generated code
  - As this code is increasingly deployed in production, concerns arise about its reliability, security and overall quality
- **Problem statement**
  - AI code may pass tests but still contain serious quality issues. Current evaluations do not address root causes and lack standardization.
- **Objective**
  - Provide a standard methodology to assess and enhance the overall quality of AI-generated code
- **Methodology**
  - Assess the impact of training data quality on code generation
  - Enhance quality via training data curation
  - Standardize evaluation by mapping results to established taxonomies
  - Measure differences between AI and human code to inform author-aware evaluation

PhD thesis

AI-generated code is progressively more integrated into production software



Correctness ≠ Quality

```
def hash_(listing_id):
    """Return the hexMD5 digest for
       the given listing identifier."""

    data = str(listing_id).encode("utf-8")
    return hashlib.md5(data).hexdigest()
```

Security

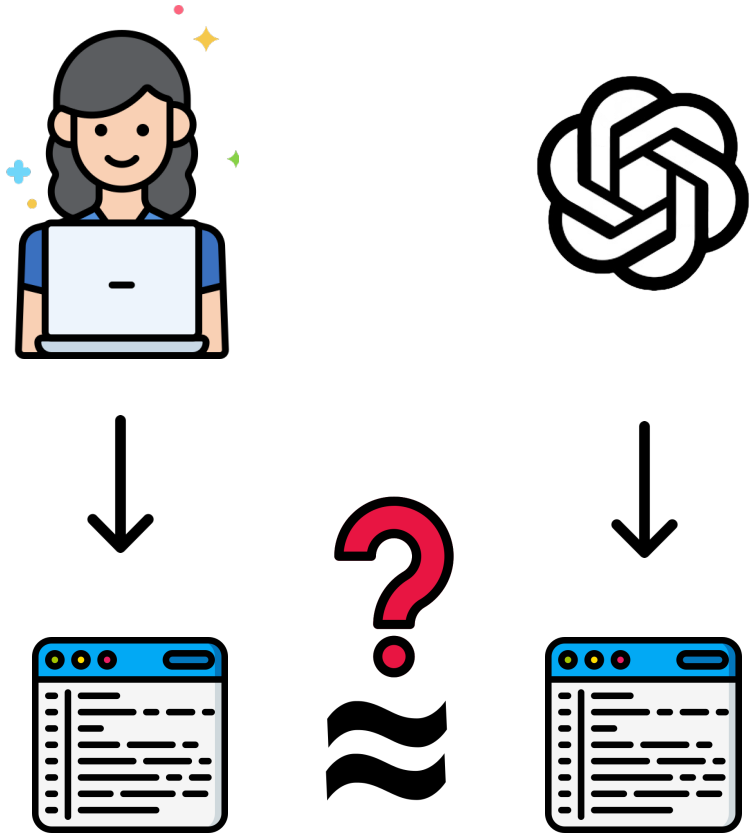
«CWE-327: Use of a Broken or Risky Cryptographic Algorithm»

What is the impact of AI coding assistants on Software Quality?

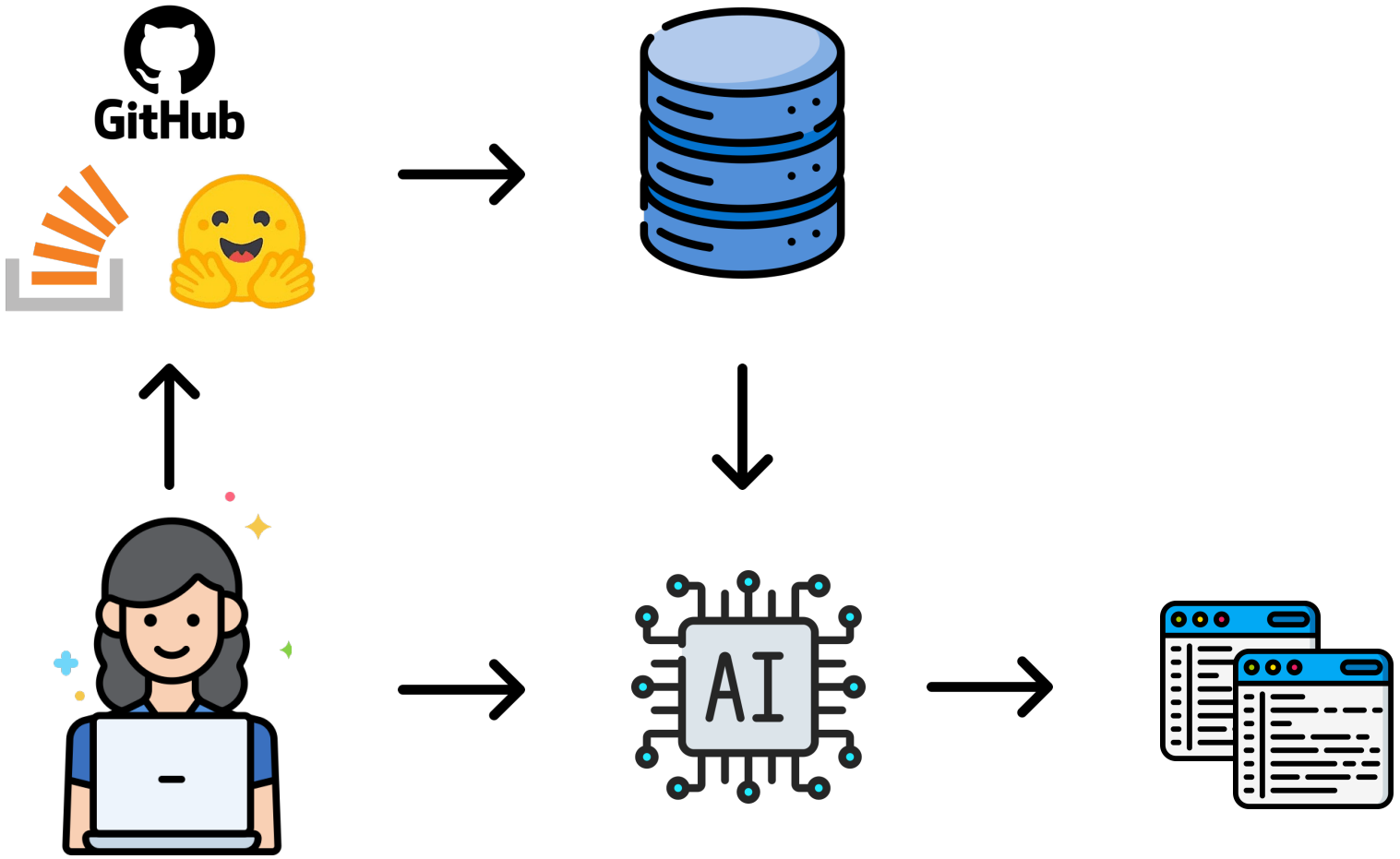
PhD thesis

Current literature

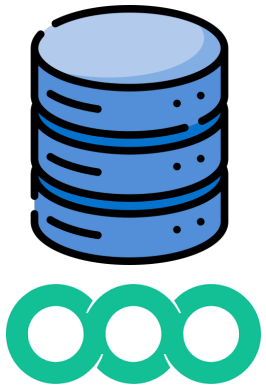
- Does not address the specific properties of AI-generated code wrt. human-written software
- Studies quality aspects in isolation (e.g., correctness, security) without assessing root causes
- Provides fragmented evaluations, with heterogeneous tools, metrics and languages



Root Cause: Training Data Quality



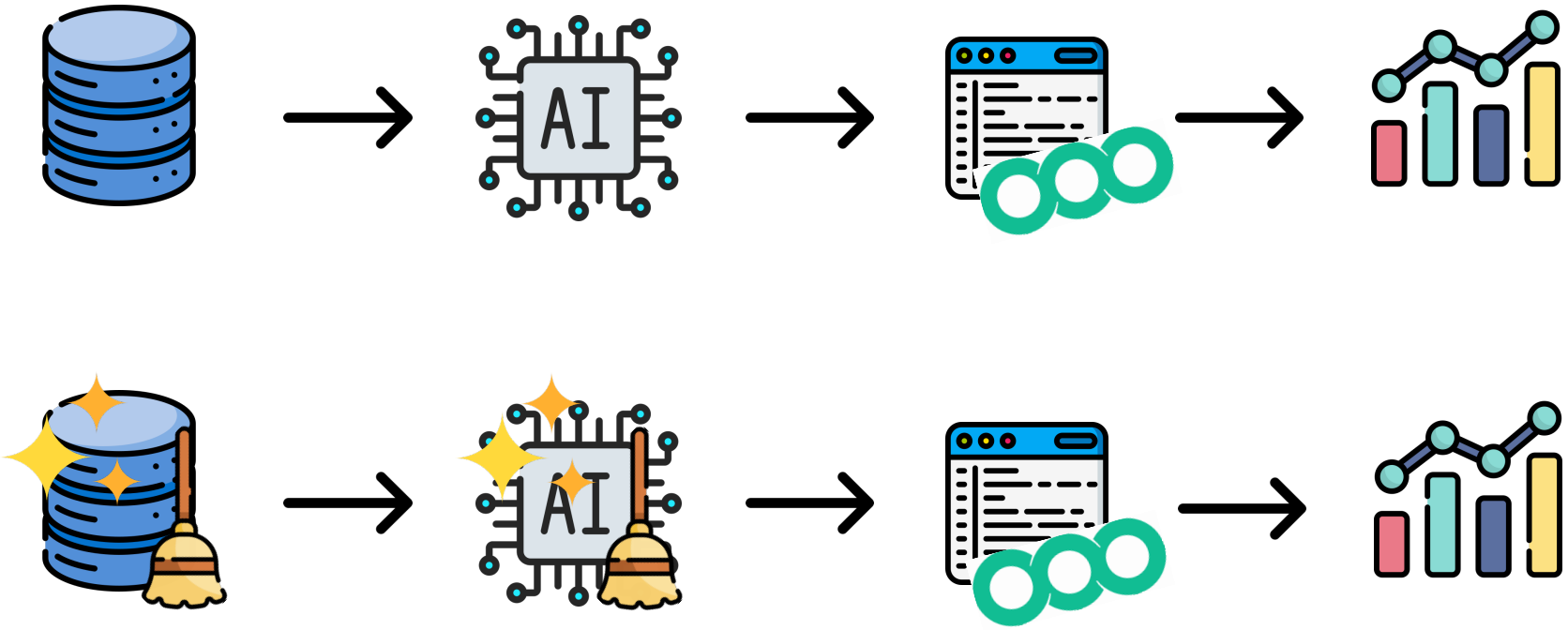
Root Cause: Training Data Quality



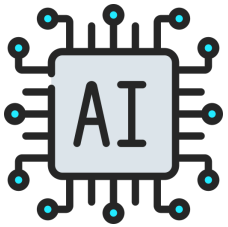
310k Quality Issues

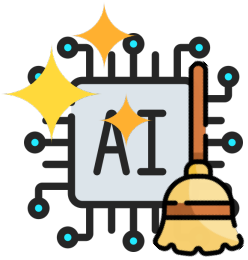
Best-Practice issue	53%
Security issue	20%
Maintainability issue	20%
Correctness issue	6%
Performance issue	<1%
Compatibility issue	<1%

Impact of Training Data on Generation



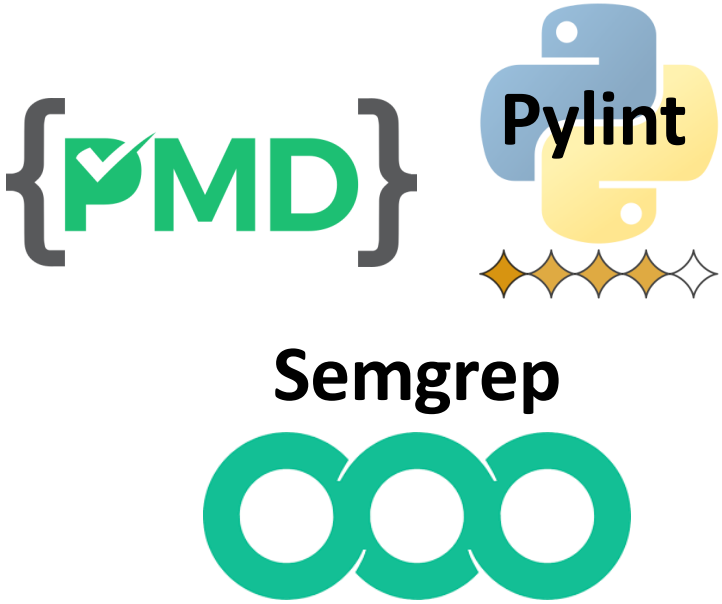
Data Curation to Improve Quality





Syntactically incorrect	7.13%	**6.68%**
Low-quality	5.85%	**2.16%**
Total # of issues	73,251	**22,466**

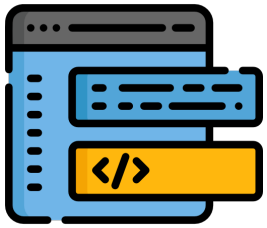
Standard Evaluation Methodology



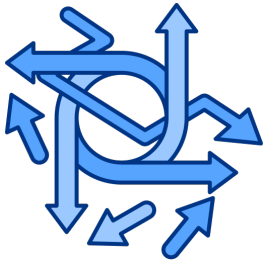
Heterogenous tools for quality assessment

Established defects and vulnerabilities taxonomies

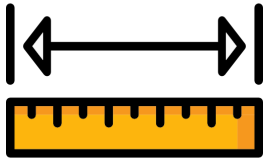
Standard Evaluation Methodology



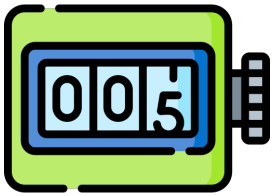
Number of Lines of Code (NLOC)



Cyclomatic Complexity Number (CCN)



Function Name Length (FNL)

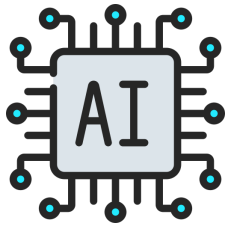


Token Count (TC)

Unique Tokens (UT)

Human vs. AI: Defects

Human developers write complex, expressive logic that introduces maintainability issues



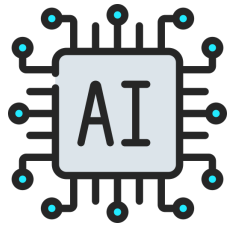
AI models generate shallow code that overuses hardcoded debugging and unused constructs

-4k

+22k

Human vs. AI: Vulnerabilities

Human developers' code is more secure except for ecception handling



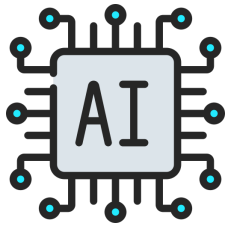
AI models generate high-severity issues such as injection flaws and information exposure

+5k

+18k

Human vs. AI: Complexity

Human developers' code is structurally more complex, both in terms of size and logical structure



AI models prioritize generating shorter code over producing highly structured or richly branched functions

-7 NLOC

-2 CCN

-8k UT