Manuel **Di Agostino**

Master's Student in Computer Science

Parma, 43121, Italy

manuel.diagostino@studenti.unipr.it | www.manueldiagostino.com | manueldiagostino | manueldiagostino | @manuel_diag

Education

University of Parma

Parma, Italy

Master's Degree in Computer Science (Expected)

Sep. 2024 - Jul. 2026

University of Parma

Parma, Italy

Bachelor's Degree in Computer Science

Sep. 2021 - Jul. 2024

Grade: 110/110 *cum laude*
Title: "Valutazione sperimentale sull'individuazione automatica di errori di programmazione nel codice generato da LLM"
Supervisor: Prof. Enea Zaffanella
Co-supervisor: Prof. Vincenzo Arceri
Keywords: *Static Analysis, AI, LLM, Software Verification and Validation*

Interests

My primary academic interests lie in Software Verification, particularly in Static Analysis by Abstract Interpretation. During my internship, I worked on integrating the PPLite library into the Mopsa static analyzer, further enhancing my practical experience and technical skills in this field. My bachelor's thesis focused on evaluating the quality of source code generated by Large Language Models (LLMs) through static analysis.

Additional experience

University of Parma

Parma, Italy

Tutor (Algorithms and Data Structures)

Feb. 2025 - Present

- Conceptual support: assist students in understanding fundamental algorithmic concepts, data structures, and their applications.
- Problem-solving sessions: guide students through exercises involving sorting, recursion, dynamic programming, and common data structures like stacks, queues, linked lists, trees, and graphs.
- Supplementary materials: provide additional learning materials and practical examples to reinforce comprehension (available at `https://github.com/unipr-org/tutorati/tree/main/ASD`).

University of Parma

Parma, Italy

Tutor (Logic and Discrete Structures)

Sep. 2024 - Present

- Academic support: provide assistance to first-year undergraduate students in understanding the theoretical concepts of logic and discrete mathematics.
- Practical exercises: guide students in solving problems related to set theory, algebraic structures, natural and structural induction, as well as propositional and predicate logic, helping them consolidate the necessary practical skills.
- Providing resources: offer supplementary materials and useful resources for further exploration of the topics covered in the course (available at `https://github.com/unipr-org/tutorati/tree/main/EdL`).

University of Parma

Parma, Italy

Internship

Oct. 2023 - Jan. 2024

- Examined the Mopsa static analyzer (`https://mopsa.lip6.fr/`) through a series of benchmarks (coreutils-benchmarks, juliet-benchmarks).
- Integrated the PPLite library (`https://github.com/ezaffanella/PPLite`) into the framework.
- Created a docker container to automate the installation (`manueldiagostino/mopsa:pplite`).

Tools and Software

Tree-sitter grammar for StrictDoc *From June 2025*

A Tree-sitter grammar for the StrictDoc language. This tool enables robust and efficient parsing of StrictDoc files, allowing for features like syntax highlighting, code navigation, and structural analysis in compatible text editors and IDEs. The grammar is designed to work with the Tree-sitter parsing framework, improving the developer experience when working with StrictDoc requirements files (`https://github.com/manueldiagostino/tree-sitter-strictdoc`).

Deep Neural Network Library *From May 2024*

A flexible C++ library for building, training, and using deep neural networks. Modular, easy-to-use, and high-performance. The library supports various neural network architectures, efficient training tools, and robust inference methods (`https://github.com/unipr-org/deep-neural-network`).

Grants

Research participant, "LLMs Meet Static Analysis: improving quality and reliability of AI-generated code"

ISCRA Project (class C), Cineca *From Jan. 2024*

Principal Investigator: Prof. Vincenzo Arceri. The goal of the project is to conduct an extensive quality and safety evaluation of the code generated with some of the most popular and open-source LLMs employing static analyzers, that can detect vulnerabilities and run-time errors statically, without executing the code. Once this information is available, it will be included in the code-generation task, to guide the LLM itself to produce a more precise and safe output, in which static analysis is somehow introduced in the pipeline of the code-generation task.

Conferences, workshops and school participations

2024 **Participant,** Lipari Summer School on Abstract Interpretation *Lipari, Italy*

2024 **Participant,** CSV 2024, 3rd Challenges of Software Verification Symposium *Venice, Italy*