LASSO PolyBench: Enhancing the LASSO Platform with Polyglot Programming Language Support for Holistic Benchmarking of Generative AI in Software Engineering Tasks

Team Project

Chair of Software Engineering / Marcus Kessel

Spring Semester 2024
Generative AI for Software Engineering Tasks

LLMs (chatbots) trained on massive amounts of open source code

- OpenAI
- ChatGPT
- Codex
- GitHub Copilot
- Tabnine
- StarCoder

... many more
Code & Test Generation Tasks

**Code Generation** (Program Synthesis)

**GitHub Copilot**

**Test Generation** (Inputs and Outputs)
Jack of all Trades, Master of None (?)

“… the robots are coming …”

“… they replace developers …”

“… hallucinating bullsh** …”

“… frees from boring tasks …”

/ r/ ProgrammerHumor
LASSO PolyBench

Replicate State-of-the-Art Benchmarks
Curated Coding Task Collections → HumanEval, MBPP, MultiPL-E etc.

Execution-based Program Analysis at Scale

Functionally correct? Quality of the code?
LASSO PolyBench

Replicate State-of-the-Art Benchmarks
Curated Coding Task Collections ➔ HumanEval, MBPP, MultiPL-E etc.

Problem
Limited to the analysis of Java code (problems)

Functionally correct?
Quality of the code?
Goal (1)

- LASSO is a leading edge software observatorium that allows advanced search and analysis techniques to be applied to “big code”. Among other things, this simplifies experimentation and the validation of tools and software engineering approaches.

- LASSO is currently limited to Java programming language support

- Goals that support the creation of LASSO PolyBench
  - add polyglot programming language support to the LASSO platform by starting with **Python language support**. This includes
    - crawling, parsing and indexing Python code
    - creating a test driver (LASSO Arena) to execute and analyze Python code of interest
  - demonstrate the **replication of benchmarks for Python coding problems** using LASSO’s concepts and data structures
    - setting up analysis pipelines in LASSO’s scripting language, LSL, to automate the experimentation process
  - demonstrate **cross-checking of code properties** (i.e., behaviour, quality etc.) across two programming languages (Java and Python)
Goal (2)

- Participants
  - 4-6 students

- Length
  - 6 months

- Prerequisites
  - Python and/or Java programming skills
  - Basic understanding of machine learning

- Language
  - English

- Organisation
  - Goals and timetable defined by agreement with the supervisor

- Applicable to MMDS: yes

- Online: By agreement

- Supervisor
  - Marcus Kessel
Open Positions – Student Assistants (HiWis)

- Be part of **AUTOR – Automated Test Oracle Recommendation**
  - Funding provided by the Ministerium für Wissenschaft, Forschung and Kunst Baden-Württemberg through *Research Seed Capital (RiSC)*

- A research project at the intersection of
  - *software engineering* (big code, mining software repositories, software testing),
  - *data science* (machine learning techniques, AI models …)

- Are you interested in … ?
  - Software testing and empirical software engineering
  - State-of-the-art recommendation and generation techniques for coding tasks

- Send an (informal) email to Marcus Kessel ([marcus.kessel@uni-mannheim.de](mailto:marcus.kessel@uni-mannheim.de))