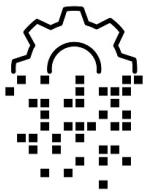
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

Prof. R. Wattenhofer

Algorithm Learning from Data



Algorithm learning is a thrilling field of research that gained significant traction over the past decade due to the advances in machine learning. Simply put, the goal is to have systems that take in some sort of specification of program behaviour and produce a concrete program that fits this input. It is hard to underestimate how desirable it is to have at least one such system that is reliable.

The topic of algorithm learning can be approached in both top-down and bottom-up fashion. Those championing the latter would likely want to learn from input-output examples, while those in favour of the former would prefer to process high-level specifications of what the program should do. Our approach begins somewhere in-between those two. At DISCO we have recently developed software for the production of partially semantized abstract syntax tree representations of code and have begun to mine these graphs to produce a dataset of "code idioms" – phrases, or code excerpts, of code elements not at all necessarily following one after another in the original source but related because of their function.

There are many things that we would like to do. Here are a few, some of which we've already started on.

1. **Behavorial Representation for C, Java, or others.** Working only with Python code up until now, we feel it is high time we expanded on this in terms of languages we analyze.
2. **Code Idiom Discovery.** So far we have been using graph methods some would call naive. We are sure this could be easily improved on.
3. **Code Completion.** Given the database of idioms available, find the right candidate to complete a piece of code.
4. **Code Search.** Given the database of idioms and a textual query such as "print all files in directory", find a piece of code that does precisely that.
5. **Program Design.** Given the database of idioms and a high-level specification of elements that the target program should consist of, produce an incomplete, high-level sketch of the program.

Candidate Profile. Varies from project to project. *Bachelor's* students will work on a smaller project, mostly coding. *Master's* students will work on an extensive project, both coding and advancing our work on the conceptual level. Generally speaking, a good candidate is a competent programmer in the language of his/her choice and is interested in one or more of the following fields: parsing, compiler design, programming language theory, statistical machine learning, deep neural networks, natural language processing.

Interested? Please contact us to learn more!

Contact (please send an email with the following as recipients)

- Peter Belcak: belcak@ethz.ch, ETZ G63
- Ard Kastrati: kard@ethz.ch, ETZ G61.3