Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed
Computing

Prof. R. Wattenhofer

Code Completion Through Graph Representations

Code completion is a thrilling field of research that gained significant traction over the past decade due to the advances in machine learning and the publicity around GitHub co-pilot. Simply put, the goal is to have systems that help programmers write code by giving hints on what to write next or on how to improve the code. It is hard to underestimate how desirable it is to have at least one such system that is reliable.

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. Instead of purely relying on models that interpret the input as text and employ natural language processing techniques to make predictions, we want to use the enriched graph representation and Graph Neural Networks (GNN) to deal with this data.

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 Completion.** Given the database of idioms available, find the right candidate to complete a piece of code.
3. **Incorporate natural language information.** Use natural language contained in code to improve the predictions of GNNs.
4. **Better graph representations.** Improve graph representations used for our learning tasks and compare different approaches.
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.
6. **Demonstrator.** Create an interactive interface that lets users experiment with our models.

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, 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
- Florian Grötschla: fgroetschla@ethz.ch, ETZ G93