literature survey

What I'm thinking

- Many tools evaluate of custom benchmark or reuse benchmarks, maybe establish a common framework?
- First create a micro benchmark
- Aggregate the real-world benchmarks, extend it?
- Evaluate tools against micro benchmark and real-world benchmarks
- Create a framework for evaluating type inference tools, make common result format

---

Learning

-

Insights so far

- Itseems the techniques so far does not consider micro-benchmarks, synthetic benchmarks. This is important given the complex nature of Python features.
- Nothing considers the Jupyter notebook angle.
- A benchmark framework can be part of Scalpel itself.

Papers

- **Static Inference Meets Deep Learning: A Hybrid Type Inference Approach for Python (2022)**
  - HiTyper
  - Combines static analysis with DL.
  - Based on type dependency graph.
  - Uses two benchmarks to evaluate. ManyTypes4Py and Typilus.
  - Does not analyze external calls, relies on typeshed project for annotations. Also they argue that DL models can do this better already.
  - A candidate for evaluation.
- **Type4Py: practical deep similarity learning-based type inference for python (2022)**
  - Another DL model for type inference.
  - Uses enhanced ManyTypes4Py dataset to create a type-checked dataset for evaluation.
  - A candidate for evaluation.

- **PyTER: effective program repair for Python type errors (2022)**
  - Focus on fixing type bugs in Python code
  - Builds a dataset for type based bug fixes by looking into git repositories
  - Uses dynamic analysis to find variable types. Claims that no ground-truth exists for dynamic languages.
  - Intra-procedural analysis, seems like not practical for real-world type inference?
  - Do not evaluate the inference by itself. Rather for type error fixing.

- **Learning type annotation: is big data enough? (2021)**
  - Introduces TypeBERT for learning types.
  - An argument that big dataset can be used to learn types.
  - Targets TypeScript.

- **ManyTypes4Py: A Benchmark Python Dataset for Machine Learning-based Type Inference (2021)**
  - Authors built a database with several features to make learning type information easier.1
  - An AST analyzer to extract type information from sources
  - A prototype model is built to predict types
  - Helpful for real-world bench for us, can be one part of our benchmark.

- **Static Type Inference for Foreign Functions of Python (2021)**
  -

- **Typilus: neural type hints (2020)**
  - A deep NN for predicting types based on stucture, names, and patterns.
  - This can be one of the target programs to avaluate against. The dataset that it uses can also be part of our benchmark.

- **TypeWriter: neural type prediction with search-based validation (2020)**
  - Another DNN to predict types.
  - Evaluated on facebook internal code.
  - Also uses a OSS code base from github with 50+ stars.
  - This again can be a target program and dataset can be added to out research.

- **Pythia: AI-assisted Code Completion System**
  - A more code completion LSTM model.
  - Says something about inferring types statically based on "object usage patterns", which is quite unclear from the text. Only discussed in a paragraph.
  - Not a direct related work, but could be interesting.

- **NL2Type: inferring JavaScript function types from natural language information (2019)**