Title: Machine learning for code completion

Laboratoire, institution et université: INRIA Lille Nord Europe

Location: Lille, France Team: Equipe RMoD – INRIA Lille Nord Europe

Internship supervisor: S. Ducasse stephane.ducasse@inria.fr

Context

Programming environments feature code completion systems: recommendation systems that guess the next word a developer is trying to type. Such recommendation systems can be based on heuristics taken from user knowledge such as developer experience or framework knowledge, and from static code analyses. A good match for code completion recommendation systems are also natural language models, NLP, and markov chains. Such statistical models can be extracted automatically using source code as datasets.

Objectives

The objective of this internship is to use machine learning model such Byte Pair Encoding Tokenization and

Use and improve the first implementation of Byte Pair Encoding https://github.com/Ducasse/BytePairEncoder

To be usable in an IDE, the code completion model needs to satisfy the following constraints:

- compact in memory
- incremental: as the user types new methods and classes the model should learn
- fast: to train and to use

A secondary objective is to make such models modular: libraries should provide pre-trained models downloadable with the given library. In such a way, the system should be quickly set-up to autocomplete in the presence of dynamically loaded code.

The student will

- study the Byte Pair Encoding implementation
- improve the implementation to be less naive
- will perform experiences to support ecompletion in Pharo
- extend the completion framework of Pharo to use BPE

The project can be followed by a M2 master internship and more.

Ressources

https://youtu.be/HEikzVL-IZU

http://www.pharo.org

http://books.pharo.org

Pharo by example, Pharo with Style

https://github.com/pharo-ai/NgramModel

https://github.com/Ducasse/BytePairEncoder

References

- Karampatsis, Hlib Babii, Robbes, Sutton, Janes, Big Code != Big Vocabulary: Open-Vocabulary Models for Source Code. ICSE '20, May 23–29, 2020, Seoul, Republic of Korea

- Hellendoorn, Vincent J et al. (2019). "When code completion fails: A case study on real-world completions". In: 2019 IEEE/ACM 41st International Conference on Soft- ware Engineering (ICSE). IEEE, pp. 960–970.

- Jurafsky, Daniel and James H. Martin (2009). Speech and Language Processing (2Nd Edition). Prentice- Hall, Inc. ISBN: 0131873210.

- Li, Jian et al. (2017). "Code completion with neural attention and pointer networks". In: arXiv preprint arXiv:1711.09573.

- Proksch, Sebastian, Johannes Lerch, and Mira Mezini (2015). "Intelligent Code Com- pletion with Bayesian Networks". In: Transactions on Software Engineering and Methodology (TOSEM) 1.25. DOI: 10.1145/2744200.

- Raychev, Veselin, Martin Vechev, and Eran Yahav (2014). "Code completion with statistical language models". In: Acm Sigplan Notices. Vol. 49. ACM, pp. 419–428.

- Robbes, Romain and Michele Lanza (2008). "How Program History Can Improve Code Completion". In: Proceedings of ASE 2008 (23rd International Conference on Automated Software Engineering), pp. 317–326. Tu, Zhaopeng, Zhendong Su, and Premkumar Devanbu (2014). "On the localness of software". In: Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering, pp. 269–280.

- NGrams for Pharo, https://github.com/pharo-ai/NgramModel