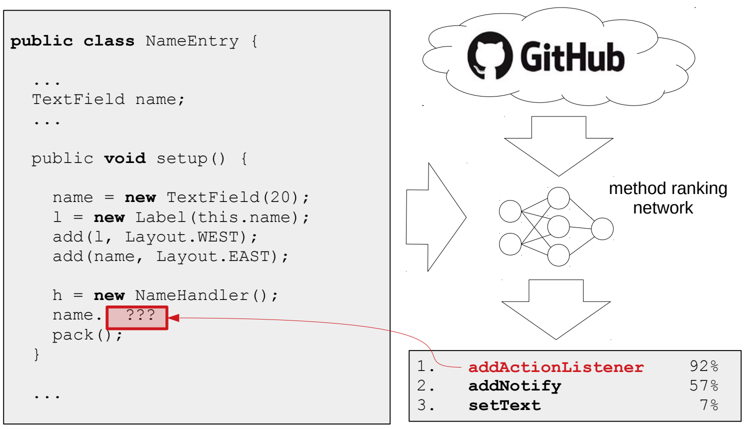
(Hrsg.): SENSYBLE-Workshop,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 1

Bidirectional Transformer Language Models for Smart Autocompletion of Source Code

Felix Binder, Johannes Villmow, Adrian Ulges[1]

AI-based support in software engineering has recently emerged as a research field: Recommenders for software commits [Da16], predicting code changes [Zh19], semantic code search [Hu19] or code captioning [ALY18] have been developed. These are usually based on machine learning components, trained on vast amounts of source code and documentation from open-source platforms such as GitHub. Another challenge – and the subject of this paper – is *smart autocompletion*: As the developer types source code, an AI-based system suggests names for methods/interfaces to use next. To do so, the system infers the plausibility of method calls from the local code context. Take a look at the following example: The AI system (more specifically, a neural *method ranking network*) analyzes a position in the current code (red, left), and infers that – out of the class `TextField`'s methods – `addActionListener` seems most plausible. The network has learned this suggestion from a vast training set of Java projects on GitHub, which contain similar usages of GUI components as the target code:



We refer to this challenge of ranking an object's method names by their plausibility in a given code context as *method ranking*. While previous work has used n-grams [Hi12],

[1] RheinMain University of Applied Sciences, DCSM Department, Wiesbaden/Germany
felix.binder@student.hs-rm.de,[johannes.villmow,adrian.ulges]@hs-rm.de

recurrent networks [Wh15] and left-to-right language modeling [In20], we evaluate the transformer network BERT [De18] based on *masked* language modeling. While masked language modeling has been very successful in *natural* language processing, it has not been used for method ranking / smart autocompletion in source code yet. Our approach first pre-trains a BERT model (more precisely, RoBERTa [Li19]) on a dataset of 10.414 open-source projects (250 million lines of code) from the GitHub Java Corpus [AS13]. Training is done by masking out tokens (more precisely, BPE tokens [SHB16]) in pieces of source code and forcing the model to predict those missing tokens. We call the resulting model *JavaBERT*.

To utilize JavaBERT for method ranking, we address the fact that method names may consist of *multiple* tokens (e.g., `add-Action-Listen-er`). We suggest two alternatives:

1. *JavaBERT-unsup*: The pre-trained (unsupervised) JavaBERT is applied by masking out variable numbers of tokens. JavaBERT's predictions on token-level are then combined in a probabilistic reasoning to predictions on method level.
2. *JavaBERT-sup*: JavaBERT is fine-tuned in an additional supervised training as a binary classifier, estimating whether a certain method call is plausible or not in context.

We evaluate both approaches in quantitative experiments on a set of random samples from the test split of the GitHub Java Corpus. Our results indicate that masked language modeling is surprisingly accurate, with a top-3 accuracy of up to 98%. We also study the impact of different contexts, e.g. only the code up to the target method call, or shorter vs. larger pieces of code.

References

[ALY18] Alon, U.; Levy, O.; Yahav, E.: code2seq: Generating Sequences from Structured Representations of Code. CoRR abs/1808.01400/, 2018.

[AS13] Allamanis, M.; Sutton, C.: Mining Source Code Repositories at Massive Scale using Language Modeling. In: The 10th Working Conference on Mining Software Repositories. IEEE, S. 207–216, 2013.

[Da16] Dam, H. K.; Tran, T.; Grundy, J.; Ghose, A.: DeepSoft: A Vision for a Deep Model of Software, 2016, arXiv: `1608.00092 [cs.SE]`.

[De18] Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018, arXiv: `1810.04805 [cs.CL]`.

[Hi12] Hindle, A.; Barr, E. T.; Su, Z.; Gabel, M.; Devanbu, P.: On the Naturalness of Software. In: 2012 34th International Conference on Software Engineering (ICSE). IEEE, S. 837–847, 2012.

[Hu19] Husain, H.; Wu, H.-H.; Gazit, T.; Allamanis, M.; Brockschmidt, M.: CodeSearchNet Challenge: Evaluating the State of Semantic Code Search, 2019, arXiv: 1909.09436 [cs.LG].

[In20] Inc., T.: The TabNine Autocompleter, https://tabnine.com, retrieved: Mar 2020.) 2020.

[Li19] Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V.: RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv preprint arXiv:1907.11692/, 2019.

[SHB16] Sennrich, R.; Haddow, B.; Birch, A.: Neural Machine Translation of Rare Words with Subword Units. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Berlin, Germany, S. 1715–1725, Aug. 2016, URL: https://www.aclweb.org/anthology/P16-1162.

[Wh15] White, M.; Vendome, C.; Linares-Vásquez, M.; Poshyvanyk, D.: Toward Deep Learning Software Repositories. In: Proceedings of the 12th Working Conference on Mining Software Repositories. MSR '15, IEEE Press, Florence, Italy, S. 334–345, 2015, ISBN: 978-0-7695-5594-2, URL: http://dl.acm.org/citation.cfm?id=2820518.2820559.

[Zh19] Zhao, R.; Bieber, D.; Swersky, K.; Tarlow, D.: Neural Networks for Modeling Source Code Edits, 2019, arXiv: 1904.02818 [cs.LG].