Machine Learning for Context-Sensitive Search in Agda

Andrej Bauer[13], Matej Petković[12], and Ljupčo Todorovski[12]

[1] University of Ljubljana, Faculty of Mathematics and Physics, Ljubljana, Slovenia
{andrej.bauer, matej.petkovic, ljupco.todorovski}@fmf.uni-lj.si
[2] Jožef Stefan Institute, Department of Knowledge Technologies, Ljubljana, Slovenia
[3] Institute for Mathematics, Physics and Mechanics, Ljubljana, Slovenia

Agda is a dependently typed functional programming language that is also used as a proof assistant. A formalisation of a theorem in Agda is written as a type, whereas the proof of the theorem is an expression of that type. We can use previously defined (and proven) theorems and lemmas in further proofs, but Agda offers only rudimentary support for searching for suitable candidates. We explore the design of a search system that goes beyond matching the type of the current goal and the types of current assumptions. The exhaustive search for an appropriate candidate is infeasible due to combinatorial explosion. Hence we aim to develop its approximation with machine learning. The preliminary experiments were conducted on the standard [4] and unimath [8] Agda libraries. Since the definitions in each of these libraries rarely reference definitions from other libraries, we create a separate training dataset for each.

There are (at least) two machine learning tasks that correspond to the problem of finding appropriate suggestions for continuing a proof. First, one can address the problem by developing a recommender system (such as those in web browsers [3]), which interprets the current context (the goal type and the partially written expression) as a query, ranks the candidates (the previous definitions in the library) by some (possibly implicit) heuristic score, and returns a ranked list with the top suggestions. Second, one can address the problem as an instance of multi-label classification (MLC), which is often used for image recognition (e.g., when labelling images with sets of labels such as {`dog`}, {`house, car`}, etc. [9]). In our case, the task is to learn a model that labels a given context (the type and the partially written expression) with a subset of a fixed finite set of labels (names of existing definitions that would complete the expression).

In the recommender-system scenario, by defining an appropriate distance among Agda definitions, we can support the search for proper matches among the existing definitions with the nearest-neighbour algorithm [2]. However, computing the distance among the definitions represented as abstract syntax trees (ASTs) can be prohibitively inefficient. A machine-learning solution for this issue is to train an embedding of the definitions to a vector space $\mathbb{R}^n$, where distances can be efficiently computed. To this end, we can use standard embedding methods, such as code2vec [1] for embedding ASTs, and word2vec [5] for capturing the semantics of names used in the definitions (e.g., *associativity* or *inverse*). The embeddings' efficiency is strongly related to the amount of training data available. For word2vec we can rely on pre-trained models for English [6], relying on the fact that the identifiers appearing in code use English words. In the case of code2vec, many training examples can be sampled as leaf-to-leaf paths in a single syntax tree.

In the second MLC scenario, the search context is defined using a set of features that the trained model can use to predict the labels. These features can be either based on the embeddings mentioned above (each feature corresponds to a single vector space dimension) or derived from a multi-graph representation of the data set. The nodes of the multi-graph represent ASTs of individual definitions, while the edges correspond to the references among the definitions. We can then extract the features using walks in the multi-graph [7]. In this case, the efficiency of machine learning is related to the length of the walks.

References

[1] Uri Alon, Meital Zilberstein, Omer Levy, and Eran Yahav. Code2vec: Learning distributed representations of code. *Proc. ACM Program. Lang.*, 3(POPL), January 2019.

[2] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[3] Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, November 2002.

[4] Nils Anders Danielsson, Matthew Daggitt, Guillaume Allais with contributions from Andreas Abel, Stevan Andjelkovic, Jean-Philippe Bernardy, Peter Berry, Bradley Hardy Joachim Breitner, Samuel Bronson, Daniel Brown, Jacques Carette, James Chapman, Liang-Ting Chen, Dominique Devriese, Dan Doel, Érdi Gergő, Zack Grannan, Helmut Grohne, Simon Foster, Liyang Hu, Jason Hu, Patrik Jansson, Alan Jeffrey, Wen Kokke, Evgeny Kotelnikov, Sergei Meshveliani, Eric Mertens, Darin Morrison, Guilhem Moulin, Shin-Cheng Mu, Ulf Norell, Noriyuki Ohkawa, Nicolas Pouillard, Andrés Sicard-Ramírez, Lex van der Stoep, Sandro Stucki, Milo Turner, Noam Zeilberger, et al. The agda standard library, version 2.0. https://github.com/agda/agda-stdlib.

[5] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[6] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[7] Matej Petković, Michelangelo Ceci, Gianvito Pio, Blaž Škrlj, Kristian Kersting, and Sašo Džeroski. Relational tree ensembles and feature rankings. *Knowledge-Based Systems*, 251:109254, 2022.

[8] Egbert Rijke, Elisabeth Bonnevier, Jonathan Prieto-Cubides, et al. Univalent mathematics in Agda. https://unimath.github.io/agda-unimath/.

[9] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2285–2294, 2016.