Just-in-Time Learning for Bottom-Up Enumerative Synthesis

MohammadAmin Raeisi

Summary

This paper introduces an alternative approach to guided program synthesis that, instead of training a model ahead of time which requires significant amounts of high-quality training data that learn probabilistic models of code automatically, shows how to learn a probabilistic context-free grammar (PCFG) for a given synthesis problem just in time, during synthesis, by learning from partial solutions encountered along the way. To make the best use of the model, it proposes a new program enumeration algorithm that improves guided bottom-up search by extending the efficient bottom-up search with guidance from probabilistic models. They implement their algorithm as $PROBE$ tool.

Their algorithm is summarized as follows: (1) They first modify the bottom-up search to enumerate programs in the order of increasing size rather than height. To this end, they modify the way subexpressions are selected from the bank in each search iteration. For example, to construct programs of size four of the form concat x y, they only replace $\langle x, y\rangle$ with pairs of programs whose sizes add up to three (the concat operation itself takes up one AST node). (2) Then generalize it to the order of decreasing likelihood defined by a probabilistic context-free grammar. (3) Finally, they illustrate how probabilistic grammar can be learned just in time by observing partial solutions during the search.

Strengths

- It has tried to use the combination of learning methods and formal methods instead of using each one separately.
- The paper has presented the motivation and problem statement well by providing an example in the introduction.
- It has shown that this method is better than the previous methods by presenting the diagrams and benchmarks.
- It has explained the different steps of the algorithm on an example, which helps to understand the algorithm better.
- They have proposed a new and interesting idea that the previous works did not give much attention to.
- The idea of their algorithm can be extended to other application domains as well.

Weaknesses

- The paper is very long, and the reader gets bored reading it!
- The pseudo-codes in the paper are very long and have a complicated notation, and at first glance, they do not convey the algorithm easily.
- It would be better to put the link to the implemented tool in the paper.
- The text of the article is not very clear and expressive and explains the content a little hard.
- It is better to include the weaknesses of the algorithm and possible ideas to solve them in the future in the paper.