Learning to Recognize Semantically Similar Program Statements in Introductory Programming Assignments

Mayur Sunil Jawalkar
Rochester Institute of Technology
mj8628@rit.edu

Hadi Hosseini
Pennsylvania State University
hadi@psu.edu

Carlos R. Rivero
Rochester Institute of Technology
crr@cs.rit.edu

ABSTRACT

With the continuously increasing population of students enrolling in introductory programming courses, instructors are facing challenges to provide timely and qualitative feedback. Automated systems are appealing to address scalability issues and provide personalized feedback to students. Many of the current approaches fail to handle flexible grading schemes and low-level feedback regarding (a set of) program statements. The combination of program static analysis in the form of program dependence graphs and approximate graph comparisons is promising to address the previous shortcomings. Current techniques require pairwise comparisons of student programs that does not scale in practice. We explore techniques to learn models that are able to recognize whether an unseen program statement belong to a semantically-similar set of program statements. Our initial results on a publicly-available introductory programming assignment indicate that it is possible to assign with high accuracy an individual program statement to some of the popular semantically-similar sets, and a large proportion is covered with these, which suggests feedback provided by instructors can be automatically propagated to other student programs.

KEYWORDS

Introductory Programming, CS1/CS2, Assessment, Program Dependence Graph, Structural Graph Clustering

1 INTRODUCTION AND BACKGROUND

The interest in computer science has originated an unprecedented growth in the number of novice programming learners [1]. Using traditional methods to provide feedback to such large numbers of students is becoming unmanageable [1]. Current approaches for assisting instructors with providing feedback to students in programming assignments mainly focus on its automation [3]; however, very few support an active role for the instructor [3]. On one hand, instructors tend to provide feedback on a smaller criteria that can be refined while grading a given assignment [2], a.k.a. holistic grading, which is not supported by existing approaches. On the other hand, instructors have very limited customization opportunities, so the feedback students receive is internally decided by each approach [3]. However, instructors possess more contextual information that should be reflected in the feedback. Furthermore, different instructors may consider certain aspects of the feedback more important than others [2].

Marin and Rivero [4] presented a technique to cluster program statements that are semantically similar, for instance, a Java program can read a number `n` from the standard input as follows: `n = s.nextInt()`, `n = Integer.parseInt(b.readLine())` and `n = s.nextShort()` (assuming that `s` is a `Scanner` and `b` is a `BufferedReader`). All these individual statements can be clustered together and be labeled as "Reading `n` from input" since they have similar semantics but not necessarily the same syntax. These clusters are promising to overcome the previous shortcomings (holistic grading and feedback customization). Unfortunately, the technique requires pairwise comparisons among student programs to discover clusters of individual program statements.

2 METHODS AND RESULTS

We explore the use of machine learning in recognizing whether an unseen program statement belongs to a cluster of semantically-similar program statements. Assuming that clusters have been annotated with feedback to be delivered to students, such feedback can be automatically propagated if a program statement belongs to a cluster. We exploit a previous technique [4], which relies on program dependence graphs,[1] to compute statement clusters over a subset of 750 Java programs of the Flipping Game.[2]

This technique discovers 74 clusters of program statements that are semantically similar. Using this as a ground truth, we exploit different features of an individual statement to learn a model to classify said statement as part of the discovered statement clusters. Our initial results show that, for the popular top-10 statement clusters, our accuracy is high (roughly 90% or more), which suggests that our approach can be used to automatically propagate customized feedback to many students programs based on few annotations.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1915404.

REFERENCES

[1] Tracy Camp, Stuart H. Zweben, Duncan Buell, and Jane Stout. 2016. Booming Enrollments: Survey Data. In *SIGCSE*. 398–399.
[2] Sue Fitzgerald, Brian Hanks, Raymond Lister, Renée McCauley, and Laurie Murphy. 2013. What are we thinking when we grade programs?. In *SIGCSE*. 471–476.
[3] Hieke Keuning, Johan Jeuring, and Bastiaan Heeren. 2019. A Systematic Literature Review of Automated Feedback Generation for Programming Exercises. *TOCE* 19, 1 (2019), 3:1–3:43.
[4] Victor J. Marin and Carlos R. Rivero. 2019. Clustering Recurrent and Semantically Cohesive Program Statements in Introductory Programming Assignments. In *CIKM*. 911–920.

SIGCSE '21, March 13–20, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-8062-1/21/03.
https://doi.org/10.1145/3408877.3439599

[1] https://github.com/victorjmarin/CIKM19/
[2] https://codeforces.com/problemset/problem/327/A