ISBN 978-981-11-7861-0
Proceedings of 2018 the 8th International Workshop on Computer Science and Engineering (WCSE 2018)
Bangkok, 28-30 June, 2018, pp. 642-646
doi: 10.18178/wcse.2018.06.107

Quantitative Analysis of Collaborative-Based Annotation in Software Engineering Education

Jie Yu[+], RenMin Wang , LingBin Zeng, Tao Wang, Gang Yin, Qiang Fan, Yue Yu

National Laboratory for Parallel and Distributed Processing, National University of Defense Technology, Changsha, China

Abstract. The primary goal of software engineering education is to develop the ability of students, a common method is to let students to read excellent open source code, so that students can improve themselves through this behavior. When reading the code, we tend to add comments to the code, in the past, the labeling of students tend to be an independent behavior, in this paper we analyze the quantitative of collaborative-based code annotation method, and the result shows the collaborative-based code annotation method is working well in improving students programming skills.

Keywords: open source software, code annotation, collaborative-based etc.

1. Introduction

The primary goal of software engineering education is to develop the ability of students and others to work together to develop software of a certain size in the industrial environment. In order to achieve this goal, a common method is to let students to read excellent open source code, so that students can learn some advanced programming skills, code specifications, design patterns and a series of software development knowledge from the open source code.

We know that if a fragment of code is rich in annotations, it will certainly help students to expedite the understanding of the function of the code, the level of code in the entire project, the structure of the project, and then grasp the code he read. So code annotation is crucial to understanding open source code, not only to help developer clearer ideas when writing code, but also to facilitate the developers to read the source code to speed up the understanding of the code.

Therefore, by reading and annotate good open source code is an important way to improve programming skills. So in the process of software engineering teaching, teachers can let students read and mark open-source code as homework, not only can help students to understand the open source code, ultimately improve students' programming ability, but also the teachers can make improvements to a teaching task by analysing students' annotation data.so that both can benefit from it.

However, many students have read the code and comment the source code often is independent behavior, and they haven't cooperated with other students, by doing this, they can't know whether their annotation is correct or not, and they can't correct their mistakes and deepen the understanding of the code through learning others annotations, ultimately they can't improve programming ability.

At the same time, there are so many web-based online learning platforms, such as CodingBat [1], CloudCoder [2], [3], Trustie [4], provide useful ways for instructors to give the students opportunities that

[+] Corresponding author.
E-mail address: yujie16@nudt.edu.cn

allow them to experience the basic concepts and techniques directly and interactively, and to help students master programing skills.

Notably, in Trustie, instructors can assign different kind of exercises to students, including homework, so teachers can use Trustie to arrange code reading and annotation tasks, it is easier to analyse data because the platform is easier to publish and collect data.

In this paper, we propose a collaborative-based annotation method to let students read the source code and comment the source code, eventually achieve the purpose of improving programming ability. So they can reduce the gap between classroom practice and industrial expectations, we can use the web-based Internet learning platform to allow students to practice open source code reading and comment. By doing this, students can learn from each other, promote each other while reading the code

The rest of paper is organized as follows. In Sect. 2, the related work is given. In Sect. 3, we introduce the learning platform Trustie and the data we have collected. Some discussions and data analysis are given after we conduct the experiments in Sect. 4, and we conclude in Sect. 5.

2. Related Work

Many researchers have already researched the important application of open source code in software engineering education.

Open source software (OSS) has shaped software engineering education programs over the past decade [5, 9, 10], OSS development poses serious challenges not only to the commercial software industry, but also to academic institutions that educate software engineers [6]. They demonstrate the importance of OSS in software engineering education, OSS has influenced the development of software engineering to some extent, the students love participating in OSS projects; With participating in OSS projects they can improve their programing skills and the capability of collaborative development. Thus, the gap between curriculum practice and industry is narrowed.

Global Software engineering (GSE) becoming standard practice, today's software engineering students will be tomorrow's global software engineers [7], embracing open source code and moving into the open source community is the best way for a student to become a software engineering professional.

At the same time, many scholars have proved the importance of code comments. The Program Comprehension (PC) is a subfield of Software Engineering(SE) aimed at elaborating methods, models and techniques to facilitate the process of understanding programs, it is very important to conceive new strategies with the objective of helping user understanding the code, and one of the most common strategy is add comments to code [8]. So we can help students understand the source code by reading annotated code.

3. Case Study Setup

To assess the effectiveness of our collaborative code-based approach in software engineering education，we conduct a case study. The goal of this case study is two-fold: (1) evaluate the effectiveness of our proposed approach; and (2) put forward some suggestions on software engineering teaching. For the case study, we need a corpus of behavioral data and annotation content of the students when comment on the platform.

Platform. TRUSTIE (Trustworthy Software tools and Integration Environment) is an online learning platform. One of the goals of the Trustie project is to provide a platform for collecting data on how students learn to program. With the platform, teachers can quickly and easily release homework, and inform each student about the relevant information. Students also read and annotate the open source code required by the teacher in the form of a repository, so that both the teacher and the student can conveniently manage the homework. And Trustie can collect data from the process that students undertake when they doing their online homework.

Data Collection. In order to verify the effectiveness of the method, with the help of the teacher, we asked the students to read and annotate the code on the platform so that we could collect their behavior data and annotated content data for effective analysis. The participants in the experiment are 24 seniors in software engineering.

4. Case Study Result

Experiment Data. In this experiment, the data we collected is summarized as shown below table.

Table 1: The data of experiment

Category	Amount
Students Number	24
Project Number	1
Code Lines Number	8887
Code Annotations Number	5824
The Code Line have been Annotated Number	2741
Annotation Rate	30.8%
Duration	1 week

As we you can see from table 1, 24 seniors collaborated on adding comments for open source project named with MiNotes with 8,887 lines of code, of which 5,824 in total and 2,741 lines were marked as multiple students could perform on the same line Marked, marked contrast ratio reached 30.8%.

Analysis of The Students Behavior Data. In this section, we will analyse the students behavioral data, we first count the number of annotations for each student, as you can see in figure 1 below:

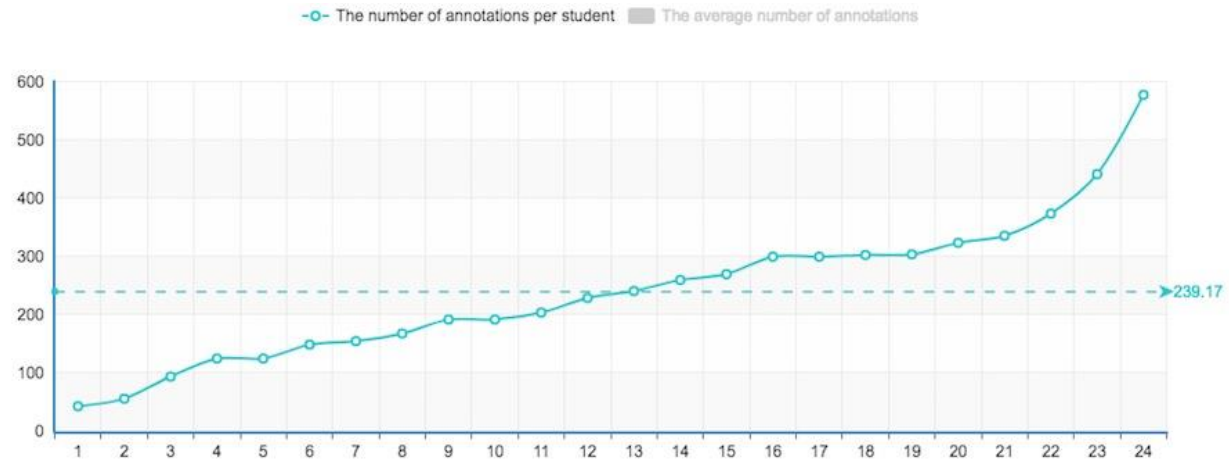

Fig. 1: The number of annotations of per students.

As shown in the figure 1, the x-axis serial number we used to represent the student, and the y-axis represents the annotation numbers, where each point on the solid line indicates the number of annotations the student generated by collaborative annotation at the time of reading source code, the dotted line is the average annotations number of all students. From the curve we can see that just half of the students annotations number less than average, and the other generally higher than the average number of annotations, and only a small number of students labeled much lower than the average or well above the average, most student annotations number is around the average.

As a result, most students hope to improve their programming skills by reading and tagging good open source code.

Analysis of The Annotations Content Data. In this section, we will analyse the annotations content data, by analysing this part of the data, we can know that collaborative-based annotation methods can improve students' understanding of open source code or not.

Our approach provides user to add annotation for a certain code line, After the student add an annotation, he can see other students' annotations of this code line and can then modify his own annotation. To guarantee the quality of annotation, We delete all annotation of files in advance and let 24 students add annotations for 39 files, which have 8887 lines in total.

After student make annotations, We arranged for 10 judges to judge all the annotations, and the average of 10 judges' score was used as the final score of one annotation. the score rules are shown in table 2.

Table 2: Score rules

item	Score
Only converted the description of the line of code	1
Explain the function of the code line	2
the annotation contains an explanation of the parameter if obtain parameters	3
Explain the role of the code line in the code fragment	4
Describe the contents of the function in the code line	5

In order to analyse whether group collaborative labeling can improve students' quality of code comments, the following data analysis was conducted.

Fig. 2: the average annotation score of every student.

At First, we extract the number of lines are annotated more than once, and extract the annotations which has been modified. After analysing the average score of annotations between the origin annotation and its modified annotation group by student, we get that the average score of the origin annotation is 2.57, while the score of the modified is 3.41. As shown in the fig 2, by using the system for making annotation, the ability to make annotation of student is improved, which prove the effectiveness of group collaborative label.

5. Conclusion

In this paper, we track the behavior data and annotations content data of students when they read and comment to the open source code in Trustie, and we collect more than 6000 data. Then, we analyse the behavior data and annotations content data. The analysis results shows that the collaborative-based annotation method is effective, it can help the students understand the source code more profound, so instructors should instruct students to learn teamwork when reading and annotating open source code so that they can better understand the open source code, accelerate development efficiency, improve software quality and improve their programming skills.

6. Acknowledgements

Please acknowledge collaborators or anyone who has helped with the paper at the end of the text. This research is supported by the National Science Foundation of China with Grant No. 61502512, No.61432020, No. 61472430 and No.61532004.

7. References

[1] Phatak, D.B.: Tools for Programming in MOOCs (Assess Student’s Knowledge)

[2] Hovemeyer, D., Hertz, M., Denny, P., et al.: CloudCoder: building a community for creating, assigning, evaluating and sharing programming exercises. In: Proceeding of the 44th ACM Technical Symposium on Computer Science Education, p. 742. ACM (2013)

[3] Hovemeyer, D., Spacco, J.: CloudCoder: a web-based programming exercise system. J. Comput. Sci. Coll. 28(3), 30 (2013)

[4] Zhang, X., Zheng, L., Sun, C.: The research of the component-based software engineering. In: Sixth International Conference on Information Technology: New Generations, ITNG 2009, pp. 1590–1591. IEEE (2009)

[5] Shaw, Mary. "Software engineering education: a roadmap." Proceedings of the Conference on the Future of Software Engineering. ACM, 2000.

[6] Jaccheri, Letizia, and Thomas Osterlie. "Open source software: A source of possibilities for software engineering education and empirical software engineering." Emerging Trends in FLOSS Research and Development, 2007. FLOSS'07. First International Workshop on. IEEE, 2007.

[7] Beecham, Sarah, et al. "How best to teach global software engineering? Educators are divided." IEEE Software 34.1 (2017): 16-19

[8] Berón, Mario, et al. "Static and dynamic strategies to understand c programs by code annotation." (2007)

[9] Zhixing Li, Yue Yu, Gang Yin, Tao Wang, Huaimin Wang. What are they talking about? Analyzing Code Reviews in Pull-based Development Model [J]. Journal of Computer Science and Technology (JCST)

[10] Yue Yu, Huaimin Wang, GangYin, and Tao Wang. 2016. Reviewer recommendaion for pull-requests in GitHub: What can we learn from code review and bug assignment? Information and Software Technology 74 (2016), 204–218.