An Empirical Study based on OpenRank Contribution Evaluation Method in Open Source Course

Zhiwei Zhu
School of Data Science & Engineering
East China Normal University
Shanghai, China
51255903077@stu.ecnu.edu.cn

Jie Wang
School of Data Science & Engineering
East China Normal University
Shanghai, China
51255903013@stu.ecnu.edu.cn

Wenrui Huang
School of Data Science & Engineering
East China Normal University
Shanghai, China
51255903058@stu.ecnu.edu.cn

Shengyu Zhao
School of Electronic & Information Engineering, Tongji University
Shanghai, China
frank_zsy@tongji.edu.cn

Xiaoya Xia
Open Source Program Office
Ant Group
Hangzhou, China
xiaoya@stu.ecnu.edu.cn

Fanyu Han
School of Data Science & Engineering
East China Normal University
Shanghai, China
52275903008@stu.ecnu.edu.cn

Wei Wang
School of Data Science & Engineering
East China Normal University
Shanghai, China
wwang@dase.ecnu.edu.cn

Yanbin Zhang
School of Data Science & Engineering
East China Normal University
Shanghai, China
ybzhang@dase.ecnu.edu.cn

***Abstract*—This paper introduces an OpenRank-based approach for evaluating open source contributions, aiming to overcome the difficulty of quantifying student contributions in open source projects. Taking the "Open Source Software Design and Development" course as a case study, a method was developed to assess student contributions in open source practices. The OpenRank algorithm, founded on developer collaboration networks, assesses student contributions in discussions, problem-solving, and coding. Experimental outcomes demonstrate that OpenRank not only coincides with traditional grading approaches but also offers a more comprehensive perspective on student contributions. Integrating OpenRank with traditional grading provides a more scientific and exhaustive evaluation of student contributions and skills in open source projects.**

***Index Terms*—Design studies, Open Source Course, Evaluation**

I. Introduction

With the rapid development of technology and the growing demand for software talent, open source software development has become a crucial aspect of software engineering education [1]. Open source projects not only encourage developers to collaborate and utilize resources and expertise in innovative ways, but they have also been shown to sustain long-term developer engagement.

Over the past two decades, colleges and universities have extensively explored practical approaches to open source education, recognizing it as a crucial means for nurturing software talent and promoting the software industry. Nevertheless, difficulties persist in cultivating open source talent, primarily due to the dearth of effective methods for evaluating students' contributions to open source projects.

Our research centers on the effective evaluation of students' contributions in open source software development within practical open source education. Employing the "Open Source Software Design and Development" course as a case study, we delve into the application of practical teaching methods in open source software education. This encompasses training in open source project practice, team collaboration, problem-solving, and communication skills. We innovatively introduce an OpenRank-based approach for evaluating developers' contributions in open source repositories.

Our main contributions include the following aspects:

1) We innovatively integrate software engineering education with open-source platforms by launching the "Open Source Software Design and Development" course, emphasizing open learning and project practice. All aspects of the course are conducted openly on open-source platforms.
2) We propose an evaluation model that utilizes collaborative networks to measure open-source contributions. This model assesses students' contributions in course repositories. We validate the efficacy of this method by comparing it with traditional evaluation approaches employed in the course.

II. RELATED WORK

A. Open Source Software and Education

Open source software is considered ideal for education because it promotes collaboration among developers, encouraging them to share resources and expertise to solve problems in innovative ways [6]. Schools and universities have been actively exploring the use of open source in education, as it has become an important method for training software professionals and advancing the software industry.

Open source software in education helps increase students' interest in development and encourages their long-term involvement [1]. However, inexperienced student developers can sometimes introduce problems to projects, such as not following established design principles or submitting disorganized pull requests. Research has shown that these types of mistakes are common among student contributions [8].

Despite these challenges, participating in large software development projects provides students with valuable experience in software engineering, which enhances their skills and professionalism [9]. Studies have also explored the motivations behind student participation in open source projects and found that such involvement is beneficial for their future careers [6]. Contributing to open source can help students develop both their technical and social skills while increasing their project experience.

B. Evaluation of Open Source Contributions

Researches on contribution to open source mainly focuse on the study of project code workload [2], [3]. In the field of open source projects, Gousios et al. proposed a linear model that considers multiple action trajectories from development and social activities to evaluate contributions [4]. Xia X studied identifying key contributors by mining activities in repositories and suggested that linear models are insufficient in reflecting social factors within an open-source software community [5]. Tsay et al.'s research provides clear evidence that social factors are highly correlated with maintainer contribution assessment during pull request reviews [11]. Joblin et al. [12] and Cheng et al. [13] respectively proposed methods based on network and user activity research into developer contributions; these studies focus on classifying developers rather than quantifying their contributions. This study adopts a collaboration network-based method to assess developer contributions and value.

PageRank [14] is a method used to calculate page importance by utilizing search engine link relevance; currently, many scholars apply PageRank's derivative algorithms in different fields such as document relevance calculation, user value calculation, movie recommendations etc. Li et al. [15] proposed a ranking algorithm ArticleRank used for measuring journal paper influence. This method takes into account the categories of cited papers in order to differentiate ranking results among papers with similar citation frequencies. Li et al. [16], based on complex network theory, introduced a movie ranking algorithm MovieRank which considers the impact of different identities of movie participants on movie rankings.

III. COURSE DESIGN

A. Course Summary

Our research focuses on the course "Open Source Software Design and Development." The course aims to cultivate students' professional skills and knowledge contributions in the field of software engineering through actual participation and contribution to open source software projects. The curriculum covers three major sections: understanding open source, contributing to open source, and developing open source, emphasizing the integration of theory and practice, as well as an open learning process. The course content is illustrated in Table 1.

The course emphasizes an open learning process, using e-textbooks and diverse teaching methods to meet individual learning needs. The course repository on GitHub provides all learning materials as open-source, allowing students to freely access, use, and share them. Course content, assignments, and discussions are conducted in the repository to promote collaboration and transparency.

B. Course Assignments

Students undertake practical assignments using open-source projects, including project selection, task allocation, Issue discussion, and code submission within the repository. Teachers and assistants select suitable open-source projects for students. During the project, students use GitHub Issues for task allocation and technical discussions, write code locally, and manage code versions with Git, including branch creation, code submission, and merging.

These open-source project tasks, conducted transparently in the course repository, aim to provide students with software development experience and an understanding of open-source community operations, collaboration culture, and project management. This approach fosters teamwork, problem-solving skills, and communication abilities.

C. Course Evaluation

When designing the student evaluation system, we drew inspiration from the Apache Way and considered the following three principles:

1) **Open Communication Principle**: Communication not recorded in the open-source community is regarded as non-existent. All discussions and collaborations must occur within the GitHub repository. This ensures that all communications and collaborations are transparently recorded and evaluated. Specifically, each discussion thread and collaboration activity should be clearly documented and timestamped.
2) **Contribution Measurement Principle**: Contributions are evaluated irrespective of personal background, focusing on long-term, in-depth involvement in open source projects, active participation in discussions, and code contributions. The more substantial the contribution, the higher the value of the Issues and pull requests (PRs) they participate in. For instance, a significant code

TABLE I
OPEN SOURCE MODULES OVERVIEW

Module	Content Overview
S1 Embracing Open Source	This module covers the basics of open source, including the history of open source, introduction to open source projects, exploring the open source world, multidisciplinary perspectives on exploring open source, the relationship between software and the software industry, commercialization of open source software, global open source software ecosystem development, and open source applications from the perspectives of software engineering and data science.
S2 Contributing to Open Source	This module focuses on how to contribute to open source projects, including the full process of open source collaboration and engineering, personal development practice tools, features and design thinking of Git, contribution and management of open source communities, team open source collaboration, and DevOps open source practices.
S3 Developing Open Source	This module focuses on open source governance and community operations, exploring the basics of enterprise open source governance, open source intellectual property, secure and reliable open source systems, digital analysis of open source communities, how to become an excellent committer, the open source path of Apache, developer relations operations, and career development in open source.

improvement or a critical bug fix could be assigned higher scores.

3) **Community Priority Principle**: A project with a mature and long-active community is always preferred over a project that initially has good quality but lacks long-term maintenance. In course practice, we measure student contributions through social collaboration relationships among students rather than individual contribution values. The evaluation of student contributions is based on the degree of collaboration among students, the quantity and quality of contributions, and their interactions. For example, a group of students who successfully complete a complex open-source project by collaborating closely will receive higher evaluations.

IV. OPENRANK ALGORITHM

The OpenRank algorithm [17] is a variant and extension of the PageRank algorithm, enabling the application of PageRank to heterogeneous information networks and extending it to high-dimensional nodes. The principle of OpenRank is similar to that of PageRank and HITS, in that the centrality of a node is determined by the centrality of other nodes pointing to it; the higher the centrality of the nodes pointing to it, the higher the centrality of the node itself. Unlike PageRank, OpenRank considers not only the structure of the collaborative network for each node's centrality but also the intrinsic value of the node. In the context of open-source collaboration, for instance, the centrality of a developer node is determined by the centrality of the PR/Issue nodes it participates in, with a higher centrality of the PR/Issue nodes it participates in leading to a higher centrality of the developer node.

A. Network Model

The OpenRank algorithm is specifically tailored for collaborative units in open-source projects, such as Issues and Pull Requests. The basic collaborative network model, as shown in Figure 2, depicts the scenario where developers collaborate around Issue and PR nodes in a GitHub repository through various collaborative behaviors.

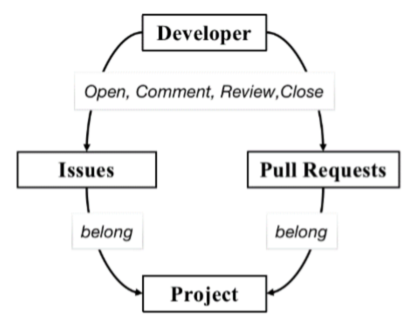

Fig. 1. The Basic Collaborative Network Model

B. OpenRank Algorithm

In the OpenRank algorithm, the OpenRank value of each node v_i in each iteration is represented by the following formula:

$$v_i = (1 - a_i) \sum_{j=1}^{|V|} \frac{w_{ji}}{d_{oj}} v_j + a_i v_0$$

where v_0 represents the initial value of the node, a_i indicates the node's dependence on its initial value, d_{oj} is the weighted out-degree of node j, and w_{ji} is the weight of the edge from node j to node i. If we organize the normalized weights w_{ji}/d_{oj} into a matrix S and a_i as diagonal values into a matrix A, according to the convergence proof, the OpenRank values of all nodes will converge to a vector, as shown below:

$$\begin{aligned} v &= \lim_{k\to\infty} v^{(k)} \\ &= \lim_{k\to\infty} [ASv^{(k-1)} + (E - A)v^{(0)}] \\ &= \lim_{k\to\infty} \left[(AS)^k v^{(0)} + \sum_{t=0}^{k-1} (AS)^t (E - A) v^{(0)}\right] \\ &= (E - AS)^{-1}(E - A)v^{(0)} \end{aligned}$$

Developers and Issue/PR nodes are connected through different types of collaborative behavior edges, including opening, commenting, reviewing, and closing. Since these behaviors represent different efforts from developers, they should be weighted differently, i.e.:

$$w_{ij} = \sum_{k=0}^{n} w_{ijk} c_k$$

where c_k represents the weight proportion of the k-th type of edge.

C. Selection of Initial Node Values

In real-world graph networks, nodes generally have some prior information or features that can be reflected as the node's initial value. The OpenRank algorithm considers the initial values of each type of node when calculating centrality. Since OpenRank is calculated based on monthly data, if a node participated in the calculation last month and has already obtained a centrality value, its initial value will be inherited from last month. If a node does not have a centrality value, its initial value will be set to 1. For Issue/PR nodes, the initial value is increased based on developers' likes. If a PR node is merged within the month, its initial value is additionally increased by 50

D. Dependence on Initial Values

When determining the centrality of different types of nodes in the network, both the initial value of the node (the node's own features) and the value obtained from interactions with other nodes (i.e., network value transfer) will be considered. It is necessary to determine the degree of dependence of the node

V. Experiment

The experimental process is divided into the following key steps: Data collection, Model construction, Results.

A. Data collection

The data used in the experiment was sourced from the "Open Source Design and Development" course during the Spring 2023 semester. To get detailed and accurate student activity data, we used the OpenDigger[1] tool to collect GitHub log data for 85 students from March to May, including Issue Comment, Issue Open, Issue Close, PR Review, PR Open, PR Close, and PR merge. We then constructed a collaboration network based on this data.

B. Network construction

We extracted the node information for developers, Issues, and PRs, as well as their collaborative relationships, from the log data. The number of each type of node over the three-month course period is shown in Table 2.

As the course progressed, the number of active developers in the course repository gradually increased (exceeding the number of course students due to the participation of teaching assistants and external personnel). The commonly used collaboration units in the course repository shifted from Issues

[1]https://github.com/X-lab2017/open-digger

TABLE II
Statistics on the Number of Network Node Types

Month	User	Issue	PR
2023-03	75	34	2
2023-04	94	58	42
2023-05	117	54	124

to PRs as the course content transitioned from theoretical to practical.

We constructed a collaboration network based on four types of nodes, including Repository nodes. Figure 3 illustrates the collaboration network of the course repository in May 2023, where blue nodes represent repositories, red nodes represent developers, and green and yellow nodes represent Issues and Pull Requests (PRs), respectively.

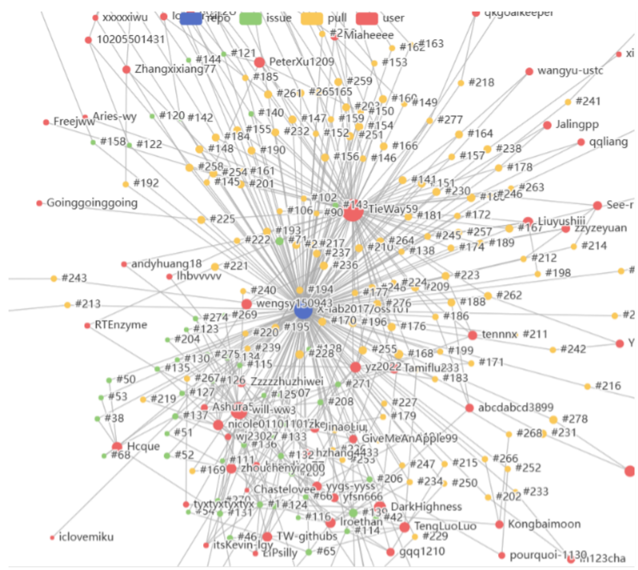

Fig. 2. Open Source Course Collaboration Network – May 2023

C. Results

We utilized OpenRank to evaluate student contributions by computing their scores from March to May 2023. Pearson and Spearman correlation analyses were conducted to compare these scores with traditional teacher assessments, including regular scores, midterm and final assignment scores, and overall grades, as detailed in Table 3. The correlation coefficients ranged from 0.25 to 0.53 for Pearson and 0.28 to 0.60 for Spearman, indicating a moderate to strong relationship between OpenRank scores and teacher-assigned grades. All correlation tests yielded p-values less than 0.01, confirming statistical significance.

OpenRank's moderate correlation with traditional grading methods suggests it complements these methods by evaluating diverse contributions in open-source projects. For example, Surefour, with an OpenRank score of 5.12, had the lowest course grade but demonstrated significant engagement in the course repository, contributing valuable code and discussions. Conversely, RTEnzyme, with a lower OpenRank score of 2.99,

TABLE III
CORRELATION ANALYSIS RESULTS BETWEEN OPENRANK AND STUDENT SCORES

Student Scores	**Pearson Correlation Coefficient**	**Pearson p-value**	**Spearman Correlation Coefficient**	**Spearman p-value**
Regular Score	0.54	1.11×10^{-7}	0.60	1.07×10^{-9}
Mid-term Assignment	0.25	1.95×10^{-2}	0.28	9.79×10^{-3}
Final Assignment	0.31	4.19×10^{-3}	0.33	1.93×10^{-3}
Total Score	0.53	1.57×10^{-7}	0.58	8.19×10^{-9}

had a high overall grade but less activity in the repository. These cases highlight OpenRank's ability to assess collaborative contributions and communication skills, which traditional methods might overlook.

OpenRank's comprehensive evaluation, which includes code submissions, pull requests, Issue discussions, and documentation contributions, offers a more detailed reflection of students' practical skills and contributions. By integrating OpenRank with traditional grading, educators can provide more nuanced feedback, guiding students towards continuous improvement and holistic development. This approach allows for a fairer and more comprehensive assessment of students' abilities and practical contributions.

VI. THREATS TO VALIDITY

Although this research has substantiated the efficacy of the OpenRank metric in evaluating students' contributions to open-source projects, it is not without limitations. Initially, the sample size is relatively small, and the diversity in student backgrounds and course contents is limited. Furthermore, the study primarily concentrates on students' contributions and performances during the course, without yet examining the long-term impact of OpenRank on learning and career development. Additionally, the qualitative analysis of the students' actual learning processes, including collaboration details and motivation analysis, lacks sufficient depth. Future research should expand the study to include a broader range of courses and participants to confirm the applicability of the findings. Additionally, tracking students' performance post-course could help to explore the long-term effects of OpenRank on their learning and career progression. By integrating in-depth interviews and behavioral observations, we can achieve a more comprehensive understanding of the generation mechanism and impacts of OpenRank scores.

VII. CONCLUSION

Our study designed and conducted the "Open Source Software Design and Development" course with the theme of open learning process and project practice. And we proposed an OpenRank indicator based on collaborative network centrality to evaluate students' contributions to open source projects. By constructing students' active data collaboration network in the course library, calculating students' OpenRank scores, and evaluating students' contributions to open source projects. The results show that the OpenRank scores of students' open source practices are consistent with the evaluation results of teachers, and OpenRank can fully reflect students' daily learning behaviors and achievements. Compared with teachers' evaluation based only on homework grades, OpenRank can more comprehensively reflect students' contributions and abilities in problem discussion, teamwork, task allocation, etc. By combining OpenRank with traditional evaluation methods, students' comprehensive abilities and actual contributions can be evaluated more comprehensively and fairly, thereby cultivating students' teamwork and practical application abilities.

REFERENCES

[1] Ellis H J C, Hislop G W, Jackson S, et al. Team project experiences in humanitarian free and open source software (HFOSS). *ACM Transactions on Computing Education (TOCE)*, 2015, 15(4): 1-23.

[2] Kan S H. Metrics and models in software quality engineering. Addison-Wesley Professional, 2003.

[3] Walston C E, Felix C P. A method of programming measurement and estimation. *IBM systems Journal*, 1977, 16(1): 54-73.

[4] Gousios G, Kalliamvakou E, Spinellis D. Measuring developer contribution from software repository data. In *Proceedings of the 2008 international working conference on Mining software repositories*. 2008: 129-132.

[5] Xia X, Weng Z, Wang W, et al. Exploring activity and contributors on GitHub: Who, what, when, and where. In *2022 29th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 2022: 11-20.

[6] Pinto G, Ferreira C, Souza C, et al. Training software engineers using open-source software: the students' perspective. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*. IEEE, 2019: 147-157.

[7] Silva J, Wiese I, German D M, et al. A theory of the engagement in open source projects via summer of code programs. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 2020: 421-431.

[8] Hu Z, Song Y, Gehringer E F. Open-source software in class: students' common mistakes. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering Education and Training*. 2018: 40-48.

[9] Holmes R, Allen M, Craig M. Dimensions of experientialism for software engineering education. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering Education and Training*. 2018: 31-39.

[10] DeKoenigsberg G. How successful open source projects work, and how and why to introduce students to the open source world. In *2008 21st Conference on Software Engineering Education and Training*. IEEE, 2008: 274-276.

[11] Tsay J, Dabbish L, Herbsleb J. Influence of social and technical factors for evaluating contribution in GitHub. In *Proceedings of the 36th international conference on Software engineering*. 2014: 356-366.

[12] JOBLIN, Mitchell, et al. Classifying developers into core and peripheral: An empirical study on count and network metrics. In: 2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE). IEEE, 2017. p. 164-174.

[13] CHENG, Jinghui; GUO, Jin LC. Activity-based analysis of open source software contributors: Roles and dynamics. In: 2019 IEEE/ACM 12th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE). IEEE, 2019. p. 11-18.

[14] Page L, Brin S, Motwani R, et al. The pagerank citation ranking: Bring order to the web. In: Proc. of the 7th International World Wide Web Conf. 1998.

[15] Li J, Willett P. ArticleRank: a PageRank-based alternative to numbers of citations for analysing citation networks. In: Aslib Proceedings. Emerald Group Publishing Limited, 2009, 61(6): 605-618.
[16] Li Y, Li C, Chen W. Research on influence ranking of chinese movie heterogeneous network based on PageRank algorithm. In: Web Information Systems and Applications: 15th International Conference, WISA 2018, Taiyuan, China, September 14–15, 2018, Proceedings 15. Springer International Publishing, 2018: 344-356.
[17] Zhao S, Xia X, Fitzgerald B, et al. OpenRank Leaderboard: Motivating Open Source Collaborations Through Social Network Evaluation in Alibaba[C]//Proceedings of the 46th International Conference on Software Engineering: Software Engineering in Practice. 2024: 346-357.