Recognizing bot activity in collaborative software development

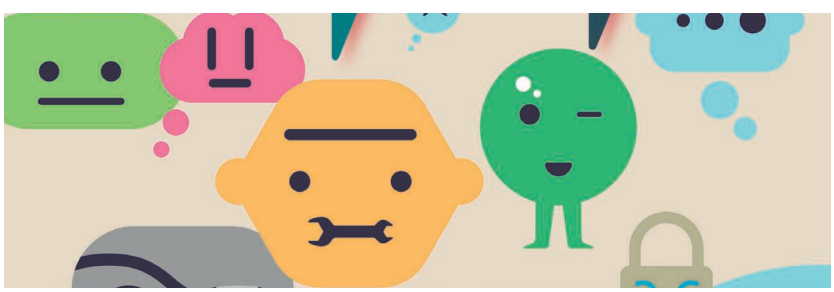
Recognizing Bot Activity in Collaborative Software Development

Mehdi Golzadeh, Tom Mens, and Alexandre Decan, University of Mons

Eleni Constantinou, Eindhoven University of Technology

Natarajan Chidambaram, University of Mons

// Using popular open source projects on GitHub, we provide evidence that bots are regularly among the most active contributors, even though GitHub does not explicitly acknowledge their presence. This poses a problem for techniques that analyze human contributor activity. //



Digital Object Identifier 10.1109/MS.2022.3178601
Date of current version: 22 August 2022

DISTRIBUTED SOFTWARE DEVELOPMENT is, by definition, a collaborative effort involving many different persons, teams, organizations, and companies. This highly collaborative software development process has led to the creation and widespread use of distributed versioning systems, such as git; social coding platforms, such as GitHub and GitLab; issue tracking tools, such as Bugzilla; code reviewing tools, such as Gerrit; and a plethora of continuous integration and deployment services.

As witnessed by initiatives, such as the CHAOSS Linux Foundation Project (https://chaoss.community) and associated software development analytics tools, such as GrimoireLab (https://chaoss.github.io/grimoirelab/), it is important to assess the health of software communities by considering the activity of each contributor. Such information is also highly relevant to credit and recognize project contributors based on their activity[1] and to allow employers to identify appropriate new team members.[2]

An important challenge in doing so is the presence of development robots (hereafter abbreviated as *bots*) that automate repetitive tasks to help software project contributors in their day-to-day activities. Not properly taking into account these bots may lead to incorrect or misleading conclusions, especially if such bots belong to the top contributors in software projects. This is likely to be the case since bots are increasingly used to automate a wide range of activities, such as welcoming newcomers, reporting test coverage, updating dependencies, detecting vulnerabilities, supporting code review, submitting pull requests, verifying licensing issues, and so on.[3]

In this article, we provide evidence that bots are regularly among

 0740-7459/22©2022IEEE

the most active contributors in popular GitHub projects, even though GitHub does not explicitly indicate these contributors as being bots. This can be problematic for tools that aim to credit human project contributors for their activity.

Acknowledging Contributions in Collaborative Development

Being able to accurately assess the contributions of project participants is valuable for many purposes. Software engineering researchers involved in empirical analyses of sociotechnical activity and productivity in software projects need such data to understand and improve the development processes.[4] Prospective employers may want to analyze developer activity profiles to identify skilled developers that match their job openings as closely as possible.[2] Individual contributors may desire to get proper credit and visibility for their—often significant—contributions to the software projects they are involved in. They may want to use this recognition for career promotion or even to get some kind of financial support for the—often voluntary—work they spend on a project.[1]

The way in which recognition is credited can differ a lot depending on the considered community. For example, OpenStack recognizes unsung heroes by discerning community contributor awards. GitHub has a similar GitHub Stars program. Initiatives such as GitHub Sponsors allow companies to sponsor open source projects to help the project contributors get the recognition they deserve. Tools such as *SourceCred*[5] aim to support communities in measuring and rewarding value creation.

It is challenging to correctly determine the contributions of each project member.[6] A first challenge concerns which types of contributions should be considered.[7,8] Typically, automated tools for identifying contributions (such as *octohattrack*[9] or *auto add contributors*[10]) provide only an impartial picture, as they tend to focus only on the types of activity that are discernible from the social coding platform (e.g., commits, pull requests, or code reviews). Other types of important contributions (e.g., finance, infrastructure, and community management) are, therefore, often ignored.[11]

A second challenge concerns how to identify contributors. If the same contributor uses multiple distinct accounts or if the same account is shared by multiple contributors, identity merging and matching techniques are needed.[12]

Another challenge concerns how to measure activity. The real effort of contributors can only be approximated. For example, the number and size of code commits could be used as a proxy of the coding effort, but this does not reflect the time required to produce such a commit since this may depend on many external factors. Moreover, such a proxy is unable to distinguish between manual or automated activity.

Last but not least, contributors may, and regularly do, use (some of) their social coding accounts to allow automated tools (i.e., bots) to carry out repetitive activity on their behalf. Whether this is intentional or not, the usage of such bots that carry out tasks on behalf of their owner can disrupt the aforementioned accreditation and recognition need. Indeed, it would be unfair to give the same recognition to a contributor whose contributions are primarily due to a bot that is committing on his/her behalf as compared to a contributor who has invested a similar effort manually. On the other hand, there is nothing wrong with contributors who try to increase their productivity by automating some of their repetitive tasks, as long as this is not intentionally done to artificially inflate one's activity. Whether and how to give proper recognition to project contributors remains an open and difficult question.

> Not properly taking into account these bots may lead to incorrect or misleading conclusions, especially if such bots belong to the top contributors in software projects.

Distinguishing Bots From Humans

A first and important step to give proper recognition to project contributors consists of distinguishing human activity from bot activity. GitHub allows project contributors to discern whether certain types of activity are automated, specifically for GitHub Apps and GitHub Actions. According to the GitHub

terms of service, bots are not permitted to register new GitHub accounts. However, things get more complex since humans are permitted to set up machine accounts to perform automated tasks (such as a continuous integration bot), provided that the human owning the account accepts the responsibility for its actions. The problem is that the GitHub application programming interface (API) does not allow all such machine accounts to be distinguished from ordinary user accounts corresponding to real human activity. As a consequence, tools that want to benefit from distinguishing human users from machine users (i.e., bots) have a hard time doing so. For example, among the available tools to accredit and acknowledge contributors, *SourceCred* and *contributors-list*[13] are limited in separating human and bot contributors by relying on the GitHub API and on a user-defined list of machine accounts to do so.

This is where bot identification tools could come to the rescue. Such tools aim to distinguish bots from humans in GitHub accounts on the basis of their behavior. The ways to do this can be quite diverse: they can be based on differences in the commenting patterns made by bots,[14] on naming conventions, or on commit activity patterns.[15] Examples of such tools are BoDeGHa,[16] which relies on comments made in pull requests and issues, and *BoDeGiC*,[17] which relies on git commit messages.

Using bot identification tools makes it easier to dissociate bot accounts from human accounts, but it can still lead to incorrect detections, notably when accounts are involved in a mix of manual human activity and automated machine-generated activity.[18] Although there is still room for improving bot identification tools,[19] they can already be very helpful in identifying bots, especially in large repositories.

Some Evidence of Bot Contributions

To justify the need to properly identify bot activity in collaborative software development projects, we provide some evidence of the presence of bots among the top contributors in popular open source projects on GitHub. We selected 10 large and active open source projects for popular programming languages (e.g., JavaScript, Java, Python, and Rust). The list notably includes VueJS,[20] a very popular front-end framework for JavaScript with more than 55,000 dependent projects on NPM; Servo,[21] an experimental browser engine written in Rust that has more than 1,000 contributors and nearly 40,000 commits; and *Cucumber-JVM*, a Java implementation of the popular test framework that has more than 53,000 dependent projects on GitHub.

We relied on the GitHub API to retrieve the contributors with the highest numbers of commits in these 10 projects as well as their account type (i.e., user or bot) as reported by the GitHub API on 9 November 2021. In contrast to prior work, which has focused on the ability of machine learning classifiers to correctly identify bots, this work focuses on the possible impact of bots that are not reported as such by the GitHub API on the attribution to contributors.

Figure 1 depicts the top 20 contributors to these 10 popular software projects, ranked in decreasing order of activity. Contributors who are responsible for at least 1% of all commits are highlighted. We classified the contributors into three categories: human users, labeled bots as reported by the GitHub API, and unidentified bots that were not reported as bots by GitHub. This classification was confirmed through a manual inspection of their activities by two authors of this article.

Figure 1 shows that the considered projects have between one and three bots among the top 20 contributors. However, fewer than half of the bots (nine out of 21) are reported as such by the GitHub API. The results are even more striking if we focus on the subset of contributors responsible for at least 1% of all commits: the overwhelming majority of the bots (18 out of 21) belong to those contributors, and most of them

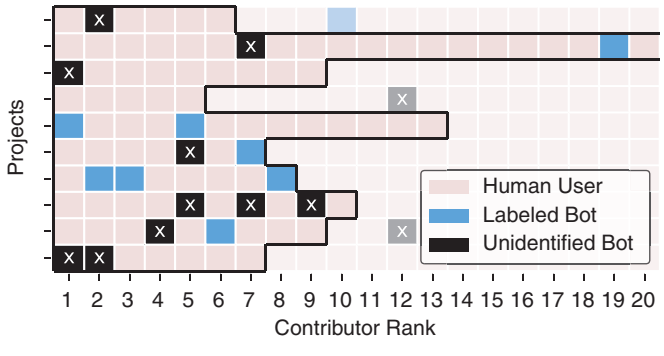


FIGURE 1. The bots observed in the top 20 most active committers in 10 popular open source projects.

(10 out of 18) are not labeled as bots by GitHub. On average, the bots are responsible for nearly one fifth of all commits in these projects.

Interestingly, we also found that some projects had explicitly credited and acknowledged bots in the list of "people that contributed to the project." While explicitly crediting and acknowledging contributors may encourage them to continue to contribute, the presence of bots in the contributor list may be perceived as a lack of consideration or respect toward human contributors.

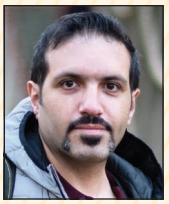
What's Next?

The results of our analysis reveal that bots play an undeniable role in large collaborative software development projects. These bots seem to carry out a significant amount of work, as many of them belong to the most active project contributors. Nevertheless, many bot accounts are not labeled as such by GitHub. Understanding why they are not labeled as bots remains an open question.

Having unidentified bots among the most active contributors may be problematic. For example, the presence of such bots in a contributor list may cause difficulties when changes in the project's Contributor License Agreement are required since such changes require the explicit approval of all human contributors. It also becomes more difficult to give due credit to human contributors for their activities and could even lead to bots (or, rather, the human owners of the associated machine accounts) receiving financial compensation for their efforts.

More advanced bot detection techniques exist,[14,15] but, even if they are more reliable than the GitHub API for

ABOUT THE AUTHORS



MEHDI GOLZADEH is a Ph.D. student at the Software Engineering Lab of the University of Mons, Mons, 7000, Belgium, in the context of the Belgian FNRS–FWO Excellence of Science research project SECO-ASSIST. His research interests include empirical software engineering research, with a focus on the social aspects of online coding and identifying automated behaviors. Golzadeh received his master's degree in information technology engineering from the University of Tehran, Iran. Contact him at mehdi.golzadeh@umons.ac.be.

TOM MENS is a full professor and head of the Software Engineering Lab at the University of Mons, Mons, 7000, Belgium. His research interests include the empirical analysis of and tooling for open source software ecosystems. Mens received his Ph.D. in sciences from Vrije Universiteit, Brussels, Begium, in the subject of software evolution. He is a Senior Member of IEEE. Contact him at tom.mens@umons.ac.be.

ALEXANDRE DECAN is a postdoctoral researcher at the Software Engineering Lab of the University of Mons, Mons, 7000, Belgium. His research interests include projects such as the UMONS Action de Recherche Concertée ECOS, the Walloon ERDF project portfolio IDEES, the FNRS–FRQ collaborative research project SecoHealth, and the Belgian FNRS–FWO Excellence of Science project SECO-ASSIST. Decan received his Ph.D. in sciences from the Faculty of Sciences of the University of Mons, Belgium, in the subject of data quality in relational databases. Contact him at alexandre.decan@umons.ac.be.

ELENI CONSTANTINOU is an assistant professor at the Eindhoven University of Technology, Eindhoven, 5612AZ, The Netherlands. Her research interests include mining software repositories, software ecosystems, and software evolution. Constantinou received her Ph.D. from the University of Thessaloniki, Greece, in the subject of software reuse. Contact her at e.constantinou@tue.nl.

NATARAJAN CHIDAMBARAM is a Ph.D. student at the Software Engineering Lab of the University of Mons, Belgium, in the context of the research project ARIAC by DigitalWallonia4.AI. His research interests include sociotechnical analysis in collaborative open source software development. Chidambaram obtained his master's degree in data science in engineering from the Eindhoven University of Technology, The Netherlands. Contact him at natarajan.chidambaram@umons.ac.be.

identifying bots, they are still not sufficient to accurately capture all bots.[19] Moreover, existing bot identification techniques mostly take into account specific coding-related activity types (e.g., commits, pull requests, issues, and so on). To cope with the diversity of contributions in collaborative development projects,[6–8] there is a need for techniques that take into account a considerably wider range of activities (e.g., discussions, bug handling, infrastructure and community management, and even financial contributions).

development activities they support and automate, there is also a need to exploit machine learning and artificial intelligence techniques to properly detect and acknowledge the presence of bots and their specific activity patterns.

> While explicitly crediting and acknowledging contributors may encourage them to continue to contribute, the presence of bots in the contributor list may be perceived as a lack of consideration or respect toward human contributors.

As a consequence, maintainers currently have no choice but to manually maintain a list of active bots in their repository by manually inspecting contributors' activities on a regular basis. While this option is feasible for smaller repositories, it is impractical to do such a manual inspection in repositories with a large number of contributors and activities. This highlights the need to rely on automatic bot identification and, in turn, calls for more research on accurate bot identification techniques. Moreover, since we expect bots to become more complex and more sophisticated in the range of

Acknowledgments

This research is supported by the DigitalWallonia4.AI research project ARIAC (grant 2010235) as well as by the Fonds de la Recherche Scientifique–FNRS under grants O.0157.18F-RG43 (Excellence of Science project SECO-ASSIST) and T.0017.18.

References

1. I.-H. Hann, J. Roberts, S. Slaughter, and R. Fielding, "Economic incentives for participating in open source software projects," in *Proc. Int. Conf. Inf. Syst.*, 2002, p. 33.
2. C. Hauff and G. Gousios, "Matching GitHub developer profiles to job advertisements," in *Proc. IEEE/ACM 12th Working Conf. Mining Softw. Repositories*, 2015, pp. 362–366, doi: 10.1109/MSR.2015.41.
3. M. Wessel *et al.*, "The power of bots: Characterizing and understanding bots in OSS projects," *Proc. ACM Hum.-Comput. Interact.*, vol. 2, no. CSCW, pp. 1–19, 2018, doi: 10.1145/3274451.
4. Z. Liao, X. Qi, Y. Zhang, X. Fan, and Y. Zhou, "How to evaluate the productivity of software ecosystem: A case study in GitHub," *Sci. Program.*, vol. 2020, Aug. 2020, Art. no. 8814247, doi: 10.1155/2020/8814247.
5. "SourceCred: A tool for communities to measure and reward value creation." GitHub. https://sourcecred.io (Accessed: Jun. 5, 2022).
6. E. Kalliamvakou, G. Gousios, D. Spinellis, and P. Nancy, "Measuring developer contribution from software repository data," in *Proc. 4th Mediterranean Conf. Inf. Syst.*, 2009, pp. 129–132, doi: 10.1145/1370750.1370781.
7. J. Cheng and J. L. C. Guo, "Activity-based analysis of open source software contributors: Roles and dynamics," in *Proc. IEEE/ACM 12th Int. Workshop Cooperative Hum. Aspects Softw. Eng.*, 2019, pp. 11–18, doi: 10.1109/CHASE.2019.00011.
8. J. L. Cánovas Izquierdo and J. Cabot, "On the analysis of non-coding roles in open source development," *Empirical Softw. Eng.*, vol. 27, no. 1, 2021, Art. no. 18, doi: 10.1007/s10664-021-10061-x.
9. "Octohatrack." GitHub. https://github.com/LABHR/octohatrack (Accessed: Jun. 5, 2022).
10. "Auto-add contributors." GitHub. https://github.com/marketplace/actions/auto-add-contributors (Accessed: Jun. 5, 2022).
11. J.-G. Young, A. Casari, K. McLaughlin, M. Z. Trujillo, L. Hébert-Dufresne, and J. P. Bagrow, "Which contributions count? Analysis of attribution in open source," in *Proc. IEEE/ACM 18th Int. Conf. Mining Softw. Repositories*, 2021, pp. 242–253, doi: 10.1109/MSR52588.2021.00036.

12. M. Goeminne and T. Mens, "A comparison of identity merge algorithms for software repositories," *Sci. Comput. Program.*, vol. 78, no. 8, pp. 971–986, 2013, doi: 10.1016/j.scico.2011.11.004.
13. "Contributors-list." Giters. https://giters.com/wow-actions/contributors-list (Accessed: Jun. 5, 2022).
14. M. Golzadeh, A. Decan, D. Legay, and T. Mens, "A ground-truth dataset and classification model for detecting bots in GitHub issue and PR comments," *J. Syst. Softw.*, vol. 175, p. 110,911, May 2021, doi: 10.1016/j.jss.2021.110911.
15. T. Dey *et al.*, "Detecting and characterizing bots that commit code," in *Proc. ACM 17th Int. Conf. Mining Softw. Repositories*, 2020, pp. 209–219, doi: 10.1145/3379597.3387478.
16. "BoDeGHa." GitHub. https://github.com/mehdigolzadeh/BoDeGHa (Accessed: Jun. 5, 2022).
17. "BoDeGiC." GitHub. https://github.com/mehdigolzadeh/BoDeGiC (Accessed: Jun. 5, 2022).
18. N. Cassee, C. Kitsanelis, E. Constantinou, and A. Serebrenik, "Human, bot or both? A study on the capabilities of classification models on mixed accounts," in *Proc. IEEE Int. Conf. Softw. Maintenance Evol.*, 2021, pp. 654–658, doi: 10.1109/ICSME52107.2021.00075.
19. M. Golzadeh, A. Decan, and N. Chidambaram, "On the accuracy of bot detection techniques," in *Proc. 4th Int. Workshop Bots Softw. Eng. (BotSE)*, Pittsburgh, PA, USA, May 9, 2022. [Online]. Available: https://doi.org/10.1145/3528228.3528406.
20. "This repo is for Vue 2." GitHub. https://github.com/vuejs/vue (Accessed: Jun. 5, 2022).
21. "The servo parallel browser engine project." GitHub. https://github.com/servo/servo (Accessed: Jun. 5, 2022).

Digital Object Identifier 10.1109/MS.2022.3194306