Bot or not? Detecting bots in GitHub pull request activity based on comment similarity

Mehdi Golzadeh Damien Legay Alexandre Decan
Tom Mens
Software Engineering Lab, University of Mons, Belgium

{ first . last }@umons.ac.be

Abstract

This is an extended abstract of a paper accepted at the ICSE 2020 co-located workshop on Bots In Software Engineering (BotSE).

Many empirical studies focus on socio-technical activity in social coding platforms such as GitHub, for example to study the onboarding, abandonment, productivity and collaboration among team members. Such studies face the difficulty that GitHub activity can also be generated automatically by various types of bots. It therefore becomes imperative to distinguish such bots from humans. We propose an automated approach to detect bots in GitHub pull request (PR) activity. Relying on the assumption that bots contain repetitive message patterns in their PR comments and using a clustering method that combines the Jaccard and Levenshtein distance. We empirically evaluate our approach by analysing 20,090 PR comments of 250 users and 42 bots in 1,262 GitHub repositories.

1 Introduction

Contemporary open source software development often takes place through online distributed social coding platforms such as GitHub [TBLJ13]. Software projects use git repositories to which their collaborators contribute by using distinct identities to commit changes, submit and review pull requests, and provide comments. In addition, projects are increasingly relying on automated bots that use one or more identities to carry out routine tasks and to interact with project members. Such bots have been shown to improve collaborative development [LSZ17, WDS+18, EGSL19, BP19], for example by improving software quality through automated refactorings [WB19], by generating patches for bugs [MUD+19], by supporting continuous integration [ASM+07] and by automatically closing abandoned issues and pull requests [WSWG19].

On the other hand, the presence of such bots raises a wide variety of issues that call for the need of detecting them in an automated way. Identity merging algorithms [GM13, KVSvdB12, WdSS+16] should take into consideration the presence of bots. The presence of bots makes it difficult for empirical studies to distinguish human behaviour from automated behaviour [LGC+19]. This can potentially lead to important biases when conducting socio-technical analyses based on historical data mined from software repositories (e.g., [TDH14, RR14, CSM19]), such as studies that aim to get insight into and improve social aspects such as onboarding [CVDF15], abandonment [CM17], team productivity [MFMZ14, VPR+15] and team collaboration [TPSZ19].

On social coding platforms such as GitHub, the current way of distinguishing bots from humans is mainly a manual and effort-prone process that is not easily reproducible. Simple heuristics, such as looking for the presence of the string "bot" in the user identity or user profile are not very reliable. We therefore propose an automated approach for detecting bots in GitHub repositories, based on similarity of their comments as bots tend to produce comments that are repetitive, whereas the variation in the comments made by humans is expected to be higher.

In: A. Editor, B. Coeditor (eds.): Proceedings of the XYZ Workshop, Location, Country, DD-MMM-YYYY, published at http://ceur-ws.org

2 Dataset

To conduct an empirical study we need a dataset containing historical data of many distinct GitHub repositories and contributor identities. According to the advice of Kalliamvakou et al. [KGB+14] , collections of repositories associated to the collaborative development of open source software packages for specific programming languages constitute good dataset candidates. For example, the Cargo package registry for the Rust programming language (`crates.io`) contains over 34K packages of which 77% are being developed on GitHub.

We relied on version 1.2.0 of the libraries.io datadump to extract the metadata of 9,954 GitHub repositories. For each of these repositories, we used the GitHub API to extract 19,632 PRs, including more than 109K distinct comments submitted by 3,250 distinct identities. Since our aim is to identify bots based on their commenting activity we require a sufficient number of PR comments per identity. Hence, we ignored identities with fewer than 20 comments.

To create a ground truth we identified bots based on a combination of different methods. First, we gathered evidence of bots that were reported by other researchers [WDS+18, WSWG19, ASM+07, AS19, EGSL19] and websites.[1] Then, we checked for presence of the word "bot" in the name and profile information of each GitHub identity [WDS+18]. Finally, we manually examined all identities in our dataset to look for further evidence of bot presence until we reached a sufficient number of bot identities to be able to carry out a proper evaluation. For all these identities two distinct authors of this paper manually and independently verified whether the identity was actually a bot, and false positives were removed Out of these 42 bots account, 17 cases ($\approx 40\%$) do not contain the string "bot" in their username. Moreover we manually selected and checked 250 random distinct human identities for inclusion in the analysis. Overall, the dataset contains 16,430 comments of 250 Human identities from 692 repositories and 3660 comments of 42 bot identities from 694 repositories.

3 Approach and results

The approach we propose is based on the assumption that bots exhibit more repetitive messages than humans, so we expect messages made by bots to be more similar than messages made by humans. To measure this similarity, we rely on two well-known metrics: the Levenshtein edit distance [Lev66] and the Jaccard distance [Jac12]. For each considered identity in the dataset, we computed both distances for all its pairs of PR messages.

Figure 1 shows for each identity (distinguishing between humans × and bots ●) the mean Levenshtein and mean Jaccard distance between its pairs of messages. The two distances produce similar results (Pearson correlation $r = 0.97$), especially for bots ($r = 0.94$ as opposed to $r = 0.79$ for humans).

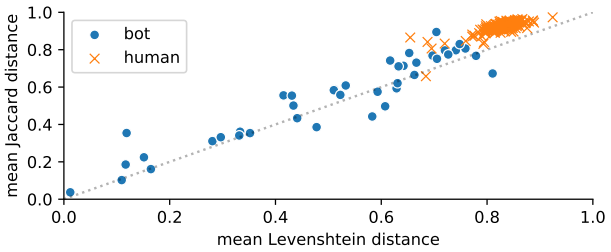

Figure 1: Mean Levenshtein and Jaccard distances between pairs of messages, per identity.

The presence of points outside the diagonal of Figure 1 (dotted grey line) indicates that these metrics are not redundant but complementary in the sense that they focus on different aspects of the messages being compared: the Jaccard distance is based on a set difference and takes lexical diversity into account; whereas the Levenshtein distance is based on a string edit distance, thereby taking the sequential structure of the message into account.

Figure 1 shows that the majority of humans have higher mean distances (0.92 for Jaccard and 0.83 for Levenshtein) than bots (0.06 for both metrics) so both distances are relatively effective in distinguishing bots from humans. Nevertheless, neither of these two metrics taken individually allow to correctly discriminate bots in the set of considered identities.

Some bots also have high mean distances, therefore mean distance is not sufficient to distinguish bots from humans. As human messages tend to be more diverse in content than bot messages, we expect them to be spread across many small clusters, whereas we expect bot messages to be concentrated in a small number of large clusters. Hence we applied an intermediate clustering step on the messages of each identity.

For this clustering step we relied on DBSCAN [EKSX96, SSE+17], a common density-based spatial clustering algorithm. DBSCAN has the advantage of not requiring to specify the number of expected clusters. Moreover, it can generate clusters of unequal sizes and supports single item clusters. Since we observed from Figure 1 that Levenshtein and Jaccard distances are not redundant, we combined them for the clustering task, as follows:

$$\mathcal{D}(C_1, C_2) = \frac{\mathcal{L}(C_1, C_2) + \mathcal{J}(C_1, C_2)}{2}$$

[1] e.g., `https://github.com/mairieli/awesome-se-bots`

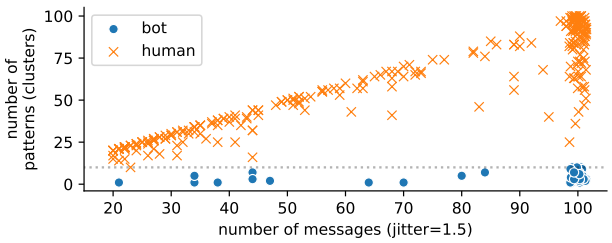

Figure 2: Number of clusters (= message patterns) and number of considered messages per identity.

Figure 2 shows the number of considered messages for each identity and the number of resulting clusters. We observe a clear separation between bots and humans. Bots have a low number of clusters regardless of the number of messages while humans have a variable, larger number of clusters.

Distinguishing bots and humans based on the number of clusters seems promising, since all 42 bots have 10 clusters or fewer, while 249 out of the 250 humans have at least 14 clusters. The remaining human contributor has exactly 10 clusters. As a result, classifying bots based on a threshold of 10 clusters (dotted grey line in Figure 2) achieves a recall of 100% and an accuracy of 97.7%.

4 Discussion

We proposed the basis for an automated approach for distinguishing bots from humans in GitHub repositories, based on their PR commenting activity. The approach is based on the assumption that bots tend to produce repetitive comments and the results we obtained are promising, since we found that the number of clusters (i.e., patterns) allows to distinguish bots from humans.

The main limitation of our work stems from the size of the dataset containing only 292 GitHub identities from Cargo packages. The dataset we relied on contains only repositories related to Cargo packages. Including a larger and broader set of repositories with more comments will certainly lead us to discover wider variety of bots and will lead to more accurate and generalizable model. We are confident that our approach will generalise to other types of bots since bots tend to rely on a limited set of patterns for their interactions. Moreover, we focused on comments made in pull requests, while other types of comments can be considered as well. Therefore, an obvious future work is to create a much larger dataset, including more (non-Cargo specific) repositories, with more (types of) comments and, consequently, of bots.

5 Conclusion

When carrying out socio-technical empirical studies based on historical data mined from GitHub repositories, it is important to be able to distinguish bots from human contributors. We proposed an approach to do so based on PR commenting activity in GitHub repositories. Relying on the assumption that bots use a limited set of repetitive message patterns in their PR comments, we proposed to detect bots by computing clusters of similar messages produced by each GitHub identity, based on a combination of the Jaccard and Levenshtein distance.

We relied on 20K PR comments from 250 randomly selected human contributors and 42 distinct bots. Our approach revealed that bots consistently exhibit fewer message patterns than humans, making this a discriminating feature to detect bots.

Acknowledgment

This work is supported by the Fonds de la Recherche Scientifique – FNRS under Grants number T.0017.18, O.0157.18F-RG43 and J.0151.20.

References

[AS19] Ahmad Abdellatif and Emad Shihab. MSRBot: Using bots to answer questions from software repositories. 2019.

[ASM+07] Ruth Ablett, Ehud Sharlin, Frank Maurer, Jörg Denzinger, and Craig Schock. BuildBot: Robotic monitoring of agile software development teams. *International Workshop on Robot and Human Interactive Communication*, pages 931–936, 2007.

[BP19] Chris Brown and Chris Parnin. Sorry to Bother You: Designing bots for effective recommendations. *1st International Workshop on Bots in Software Engineering*, pages 54–58, 2019.

[CM17] Eleni Constantinou and Tom Mens. An empirical comparison of developer retention in the RubyGems and npm software ecosystems. *Innovations in Systems and Software Engineering*, 13(101), 2017.

[CSM19] Di Chen, Kathryn T. Stolee, and Tim Menzies. Replication can improve prior results: A github study of pull request acceptance. In *International Conference on Program Comprehension (ICPC)*, pages 179–190. IEEE, 2019.

[CVDF15] Casey Casalnuovo, Bogdan Vasilescu, Premkumar Devanbu, and Vladimir Filkov. Developer onboarding in GitHub: The role of prior social links and language experience. In *ESEC/FSE*, pages 817–828. ACM, 2015.

[EGSL19] L. Erlenhov, F. Gomes de Oliveira Neto, R. Scandariato, and P. Leitner. Current and future bots in software development. In *1st International Workshop on Bots in Software Engineering*, pages 7–11, May 2019.

[EKSX96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 226—231. AAAI Press, 1996.

[GM13] Mathieu Goeminne and Tom Mens. A comparison of identity merge algorithms for software repositories. *Science of Computer Programming*, 78(8):971–986, August 2013.

[Jac12] Paul Jaccard. The distribution of the flora in the alpine zone. *New Phytologist*, 11(2):37–50, February 1912.

[KGB+14] Eirini Kalliamvakou, Georgios Gousios, Kelly Blincoe, Leif Singer, Daniel M. German, and Daniela Damian. The promises and perils of mining GitHub. In *International Conference on Mining Software Repositories (MSR)*, pages 92–101. ACM, 2014.

[KVSvdB12] Erik Kouters, Bogdan Vasilescu, Alexander Serebrenik, and Mark G. J. van den Brand. Who's who in Gnome: using LSA to merge software repository identities. In *International Conference on Software Maintenance (ICSM)*, pages 592–595. IEEE, 2012.

[Lev66] Vladimir Levenshtein. Binary codes capable of correcting deletions insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710, February 1966.

[LGC+19] Shangqing Liu, Cuiyun Gao, Sen Chen, Lun Yiu Nie, and Yang Liu. ATOM: Commit Message Generation Based on Abstract Syntax Tree and Hybrid Ranking. 14(8):1–14, 2019.

[LSZ17] Carlene Lebeuf, Margaret Anne Storey, and Alexey Zagalsky. Software Bots. *IEEE Software*, 35(1):18–23, 2017.

[MFMZ14] André N. Meyer, Thomas Fritz, Gail C. Murphy, and Thomas Zimmermann. Software developers' perceptions of productivity. *ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE)*, pages 19–29, 2014.

[MUD+19] Martin Monperrus, Simon Urli, Thomas Durieux, Matias Martinez, Benoit Baudry, and Lionel Seinturier. Repairnator patches programs automatically. *Ubiquity*, 2019(July):1–12, 2019.

[RR14] Mohammad Masudur Rahman and Chanchal K. Roy. An insight into the pull requests of GitHub. In *Working Conference on Mining Software Repositories*, pages 364–367. ACM, 2014.

[SSE+17] Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. Dbscan revisited, revisited: why and how you should (still) use dbscan. *ACM Transactions on Database Systems (TODS)*, 42(3):1–21, 2017.

[TBLJ13] F. Thung, T. F. Bissyandé, D. Lo, and L. Jiang. Network structure of social coding in GitHub. In *European Conference on Software Maintenance and Reengineering (CSMR)*, pages 323–326, March 2013.

[TDH14] Jason Tsay, Laura Dabbish, and James Herbsleb. Let's talk about it: Evaluating contributions through discussion in github. In *ACM SIGSOFT Symposium on Foundations of Software Engineering (FSE)*, pages 144–154. ACM, 2014.

[TPSZ19] Damian A. Tamburri, Fabio Palomba, Alexander Serebrenik, and Andy Zaidman. Discovering community patterns in open-source: a systematic approach and its evaluation. *Empirical Software Engineering*, 24(3):1369–1417, Jun 2019.

[VPR+15] Bogdan Vasilescu, Daryl Posnett, Baishakhi Ray, Mark G.J. van den Brand, Alexander Serebrenik, Premkumar Devanbu, and Vladimir Filkov. Gender and tenure diversity in github teams. In *Human Factors in Computing*

Systems (CHI), pages 3789–3798. ACM, 2015.

[WB19] Marvin Wyrich and Justus Bogner. Towards an autonomous bot for automatic source code refactoring. *1st International Workshop on Bots in Software Engineering*, pages 24–28, 2019.

[WDS+18] Mairieli Wessel, Bruno Mendes De Souza, Igor Steinmacher, Igor S. Wiese, Ivanilton Polato, Ana Paula Chaves, and Marco A. Gerosa. The power of bots: Understanding bots in OSS projects. *Proceedings of the ACM on Human-Computer Interaction*, 2018.

[WdSS+16] I. S. Wiese, J. T. d. Silva, I. Steinmacher, C. Treude, and M. A. Gerosa. Who is who in the mailing list? comparing six disambiguation heuristics to identify multiple addresses of a participant. In *International Conference on Software Maintenance and Evolution (ICSME)*, pages 345–355, Oct 2016.

[WSWG19] Mairieli Wessel, Igor Steinmacher, Igor Wiese, and Marco A Gerosa. Should i stale or should i close?: An analysis of a bot that closes abandoned issues and pull requests. In *1st International Workshop on Bots in Software Engineering*, pages 38–42. IEEE Press, 2019.