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

Mehdi Golzadeh
mehdi.golzadeh@umons.ac.be
Software Engineering Lab – University of Mons
Mons, Belgium

Damien Legay
damien.legay@umons.ac.be
Software Engineering Lab – University of Mons
Mons, Belgium

Alexandre Decan
alexandre.decan@umons.ac.be
Software Engineering Lab – University of Mons
Mons, Belgium

Tom Mens
tom.mens@umons.ac.be
Software Engineering Lab – University of Mons
Mons, Belgium

ABSTRACT

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 bots of a different nature. It therefore becomes imperative to distinguish such bots from human users. 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, we analyse the similarity between multiple messages from the same GitHub identity, 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. Our results show that the method is able to clearly separate bots from human users.

KEYWORDS

social coding, bot activity, empirical analysis, GitHub, text similarity, software repository mining, pull requests

ACM Reference Format:
Mehdi Golzadeh, Damien Legay, Alexandre Decan, and Tom Mens. 2020. Bot or not? Detecting bots in GitHub pull request activity based on comment similarity. In *IEEE/ACM 42nd International Conference on Software Engineering Workshops (ICSEW'20), May 23–29, 2020, Seoul, Republic of Korea.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3387940.3391503

1 INTRODUCTION

Contemporary open source software development often takes place through online distributed social coding platforms such as GitHub [25]. Software projects use git repositories to which their collaborators contribute by using distinct GitHub 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 GitHub identities to carry out routine tasks and to interact with project members. Such bots have been shown to improve collaborative development [4, 8, 17, 28], for example by improving software quality through automated refactorings [31], by generating patches for bugs [21], by supporting continuous integration [2] and by automatically closing abandoned issues and pull requests [29].

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. From an ethical point of view, human contributors should have the ability and the right to know whether they are interacting and discussing with a bot or with a human contributor. Specific projects or platforms may provide guidelines or impose specific rules that need to be respected by bots. Privacy regulations (such as the European GDPR) may differ depending on whether a contributor is a bot or a human. Identity merging algorithms [10, 15, 30] should take into consideration the presence of bots to avoid accidentally merging them with humans and hence artificially inflating their contributions. This is especially important if human contributors need to be accountable for or accredited for their own contributions. In a more general sense, the presence of bots makes it difficult for empirical studies to distinguish human behaviour from automated behaviour [19]. This can potentially lead to important biases when conducting socio-technical analyses based on historical data mined from software repositories (e.g., [6, 22, 26]), such as studies that aim to get insight into and improve social aspects such as onboarding [5], abandonment [7], team productivity [20, 27] and team collaboration [24].

Bots are quite common in social environments like Twitter and Wikipedia and methods have been proposed to identify bots according to the characteristics of their activity and their messages [3, 11, 16]. 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 either, as they tend to lead to many false positives and false negatives.

We therefore propose an automated approach for detecting bots in GitHub repositories, based on their commenting activity in pull requests (PR). The approach is based on the hypothesis that bots

ICSEW'20, May 23–29, 2020, Seoul, Republic of Korea

ACM ISBN 978-1-4503-7963-2/20/05…$15.00
https://doi.org/10.1145/3387940.3391503

Table 1: Examples of repetitive message patterns observed in PR comments of bots contributing to GitHub repositories for Cargo projects. The … part of the message tends to vary across different comments.

bot name	PR comment message patterns
coveralls	[![Coverage Status]…Coverage increased (+…%) to …% when pulling …on …into …on …. [![Coverage Status]…Coverage decreased (-…%) to …% when pulling …on …into …on …. [![Coverage Status]…Coverage remained the same at …% when pulling …on …into …. on …. ## Pull Request Test Coverage Report for [Build …]…
homunkulus	The latest upstream changes (presumably …) made this pull request unmergeable. Please resolve the merge conflicts. :brokenheart: Test failed - [status-travis]… :hourglass: Testing commit …with merge … :pushpin: Commit …has been approved by …
CLAassistant	All committers have signed the CLA. Thank you for your submission, we really appreciate it. Like many open source projects, we ask that you sign our [Contributor License Agreement](…) before we can accept your contribution
rfcbot	…proposal cancelled Team member …has proposed to merge this. The next step is review by the rest of the tagged teams: … :bell: **This is now entering its final comment period**, as per the [review above]…. :bell: The final comment period, with a disposition to **merge**, as per the [review above]…, is now **complete**. The final comment period is now complete.

tend to produce comments that very frequently feature sets of repetitive message patterns in their comments, whereas the variation in the comments made by humans is expected to be higher. Table 1 provides anecdotal evidence of this hypothesis by showing some examples of manually identified bots in Cargo projects hosted on GitHub. Although each of these bots serves a very different purpose, they all reveal repetitive message patterns.

Based on the above hypothesis, we present a technique to identify bots by computing, for each GitHub identity (which can be a human or a bot), the pairwise distance between its comment messages, clustering these distance metrics to take into account different groups of repetitive message patterns, and using quantitative information about these clusters to distinguish bots from humans.

2 DATASET

To conduct an empirical study we need a dataset containing historical data of many distinct GitHub repositories and contributor identities. Following the advice of Kalliamvakou et al. [13] we need to avoid repositories that have been created merely for experimental or personal reasons, or that only show sporadic traces of PR comments. Good candidate datasets are collections of repositories associated to the collaborative development of open source software packages for specific programming languages. 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 more than 15K Cargo packages [14] and their associated git repositories. Many packages share the same git repository. We ignored all packages that did not have any associated git repository or whose git repository was not hosted on GitHub. In addition, some git repositories were no longer available at the time of data extraction, restricting our initial dataset to 9,954 GitHub repositories.

For each of these repositories, we used the GitHub API to extract all PRs submitted during a 3-month period (from March to June 2018). Only 1,445 of the considered repositories had PRs within that time period, accounting for 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 validate our approach we need a ground truth of bot identities. To this end, we identified bots based on a combination of different methods. First we gathered evidence of bots that were reported by other researchers [1, 2, 8, 28, 29] and websites.[1] We also checked for presence of the word "bot" in the name and profile information of each GitHub identity, as this is frequently the case [28].

For all such identities two distinct authors of this paper manually and independently verified whether the identity was actually a bot, and false positives were removed. To gather instances of false negatives, i.e., bots without the word "bot" in their name (e.g., "bors") we manually eyeballed 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. In this way, we managed to identify 42 bots in our dataset, and at least two authors of this paper independently confirmed their bot status. Out of these 42 bots account, 17 cases (i.e., about 40%) do not contain the string "bot" in their username.

Since our initial dataset contains, as one would expect, many more human identities than bot identities, we manually selected 250 random distinct human identities for inclusion in the analysis. Again, these identities were manually verified and confirmed to be real humans by at least two authors of this paper. Table 2 summarizes the characteristics of the reduced dataset we have used for our case study.

Table 2: Summary of the dataset for the case study

	Humans	Bots	Total
GitHub identities	250	42	292
PR comments	16,430	3,660	20,090
GitHub repositories	692	694	1,262

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

It is important to point out that, although the proportion of humans and bots considered for our analysis (i.e., 250 against 42) does not reflect the proportion in the full dataset, where there is a significantly higher proportion of humans, this will be of no impact on our approach as it will solely rely on analysing the comments of individual accounts.

3 APPROACH AND RESULTS

The approach we propose is based on the assumption that bots exhibit more repetitive messages than humans, since they are expected to automate repetitive tasks. As a consequence, 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 [18] and the Jaccard distance [12].

The Levenshtein distance $lev(C_1, C_2)$ quantifies the difference between two character sequences C_1 and C_2 by counting the number of single-character edits (insertion or deletion of a character, or a substitution of a character by another one). In this paper, we rely on the normalized Levenshtein distance:

$$\mathcal{L}(C_1, C_2) = \frac{lev(C_1, C_2)}{max(|C_1|, |C_2|)}$$

The Jaccard distance $J(C_1, C_2)$ compares the number of distinct common words in C_1 and C_2 with the total number of distinct words in C_1 and C_2. If $words(C)$ denotes the set of words in C, then the Jaccard distance between C_1 and C_2 is computed as:

$$\mathcal{J}(C_1, C_2) = 1 - \frac{|\ words(C_1) \cap words(C_2)\ |}{|\ words(C_1) \cup words(C_2)\ |}$$

For each considered identity in the dataset, we computed the Levenshtein and Jaccard distance of 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. We observe that the two distances produce very similar results (Pearson correlation $r = 0.97$), especially for identities corresponding to bots ($r = 0.94$ as opposed to $r = 0.79$ for humans).

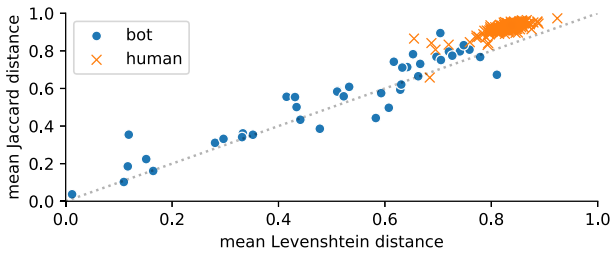

Figure 1: Mean Levenshtein and Jaccard distances between pairs of messages, per identity.

The presence of points outside the diagonal of Figure 1 (dotted grey line) however indicates that both distance metrics are not redundant. This is not surprising since, by definition, they are complementary in 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.

We also observe from Figure 1 that both distance metrics seem to be relatively effective in distinguishing humans from bots: as expected, the majority of humans have higher mean distances than bots. For instance, half of the humans have a mean distance of 0.92 for Jaccard and 0.83 for Levenshtein, whereas half of the bots have a mean distance of 0.06 for both metrics. Nevertheless, neither of these two metrics taken individually makes it possible to correctly discriminate bots in the set of considered identities. Even if humans can be identified on the basis of higher mean distances, this does not effectively distinguish them from bots, since we observe that some bots also have such high mean distances. Manual inspection of these bots revealed that, while they do have repetitive messages, these messages follow several distinct patterns, implying a higher distance value when two messages belonging to different patterns are compared.

Since the distance between messages does not take into account the presence of several patterns in messages belonging to bots, we apply an intermediate clustering step on the messages of each identity. This step aims to identify the number of patterns used by each identity. Indeed, 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.

For this clustering step we relied on DBSCAN [9, 23], a common density-based spatial clustering algorithm. Other clustering techniques could have been used for this task as well, but DBSCAN has the advantage of not requiring to specify the number of expected clusters. Moreover, it can generate clusters of unequal sizes and allows clusters to contain a single item. Since we observed from Figure 1 that Levenshtein and Jaccard distances are not redundant, we combined them for the clustering task, as follows:

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

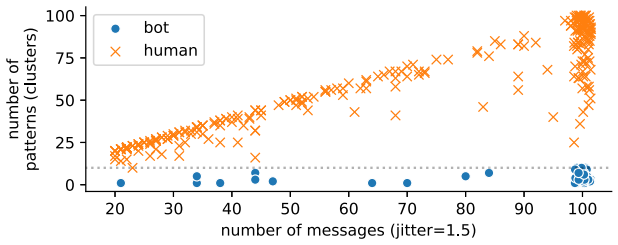

Figure 2: Number of clusters (= message patterns) and number of considered messages per identity.

Figure 2 shows the number of resulting clusters and the number of considered messages for each identity. We observe a clear separation between, on the one hand, bots having a lower number of clusters regardless of the number of messages and, on the other hand, humans with 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. Manual inspection of the messages of this specific contributor revealed that this contributor performs code reviews through an external tool that automatically generates specific and repetitive headers and footers in each message. This contributor had only 23 messages in our dataset, resulting in a lower number of clusters than other humans, close to the numbers we observed for bots.

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

In this paper, we proposed an automated approach for distinguishing bots from human contributors in GitHub repositories, based on their commenting activity in pull requests. The approach is based on the assumption that bots tend to produce comments that very frequently feature sets of repetitive messages. Such sets are identified by means of a clustering step based on a combination of the Levenshtein and Jaccard distances. The results we obtained are promising, since we found that the number of clusters (i.e., patterns) permits distinction of bots from humans, and that a value of 10 clusters is a discriminating threshold resulting in only one misclassified contributor in our setup. Moreover, the approach allowed us to identify bots that were not detected by existing methods so far.

The main limitation of our work stems from the size of the dataset containing only 292 GitHub identities. A larger dataset will enable us to perform hyper-parameter tuning, notably to assess the impact of the number of messages that must be considered for each identity. We expect the model to be more accurate when presented with more messages, as fewer messages will most likely lead the approach to misclassify contributors making use of some repetitive messages (e.g., "Thanks!"), or making heavy use of commands in their comments (e.g., "bors +r", used to merge a pull request).

The dataset we relied on contains only repositories related to Cargo packages. Including a larger and broader set of repositories will certainly lead us to discover wider variety of bots. 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. Although we did not encounter any bot that exhibited more than 10 different message patterns, some bots can be set up to produce different message patterns in different projects, thereby inflating the number of clusters reported by DBScan. Even though this did not happen for the cases considered in our current study, it could possibly lead to some bots being misclassified as humans.

Moreover, we focused on comments made in pull requests, but other data sources can be considered as well. We expect our approach to generalise to issue comments, pull review comments and commit comments.

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. Creating such a dataset is not a trivial task, though. Although data collection is made accessible thanks to data sources such as the GitHub API, the creation of a ground truth requires the manual tagging of bots and humans, which is a very laborious and time-consuming task. We are convinced that publicly releasing such an annotated dataset will be very valuable for the research community, especially given the difficulty to create and obtain it.

As soon as we are able to validate the proposed approach on a much larger dataset, we aim to implement the resulting model within a tool or as a reusable software library. Such a tool would take an identity as input, and would analyse its comments to indicate whether this identity corresponds to a bot or to a human contributor. This kind of tool could be very valuable, notably to support empirical studies of software repositories that need to distinguish between bots and humans on a large scale.

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.

Relying on the pull request comment activity in GitHub repositories of Cargo packages, we randomly selected 250 active human contributors and 42 distinct bots that were manually verified to obtain a ground truth against which to evaluate the approach. Taking into account over 20K PR comment messages, our approach revealed that bots consistently exhibit fewer message patterns than humans, making this a discriminating feature to detect bots.

These results are very encouraging, but further work is still required to refine the parameters of the approach, and to validate the resulting model on a larger dataset.

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

[1] Ahmad Abdellatif and Emad Shihab. 2019. MSRBot: Using Bots to Answer Questions from Software Repositories. (2019). https://doi.org/10.5281/zenodo.3351550 arXiv:1905.06991

[2] Ruth Ablett, Ehud Sharlin, Frank Maurer, Jörg Denzinger, and Craig Schock. 2007. BuildBot: Robotic monitoring of agile software development teams. *International Workshop on Robot and Human Interactive Communication* (2007), 931–936. https://doi.org/10.1109/ROMAN.2007.4415217

[3] Amit A. Amleshwaram, Narasimha Reddy, Sandeep Yadav, Guofei Gu, and Chao Yang. 2013. CATS: Characterizing automation of Twitter spammers. *2013 5th International Conference on Communication Systems and Networks, COMSNETS 2013* (2013). https://doi.org/10.1109/COMSNETS.2013.6465541

[4] Chris Brown and Chris Parnin. 2019. Sorry to Bother You: Designing bots for effective recommendations. *1st International Workshop on Bots in Software Engineering* (2019), 54–58. https://doi.org/10.1109/BotSE.2019.00021

[5] Casey Casalnuovo, Bogdan Vasilescu, Premkumar Devanbu, and Vladimir Filkov. 2015. Developer Onboarding in GitHub: The Role of Prior Social Links and Language Experience. In *ESEC/FSE*. ACM, 817–828. https://doi.org/10.1145/2786805.2786854

[6] Di Chen, Kathryn T. Stolee, and Tim Menzies. 2019. Replication Can Improve Prior Results: A GitHub Study of Pull Request Acceptance. In *International Conference on Program Comprehension (ICPC)*. IEEE, 179–190. https://doi.org/10.1109/ICPC.2019.00037

[7] Eleni Constantinou and Tom Mens. 2017. An empirical comparison of developer retention in the RubyGems and npm software ecosystems. *Innovations in Systems and Software Engineering* 13, 101 (2017). https://doi.org/10.1007/s11334-017-0303-4

[8] L. Erlenhov, F. Gomes de Oliveira Neto, R. Scandariato, and P. Leitner. 2019. Current and Future Bots in Software Development. In *1st International Workshop on Bots in Software Engineering*. 7–11. https://doi.org/10.1109/BotSE.2019.00009

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

[10] Mathieu Goeminne and Tom Mens. 2013. A comparison of identity merge algorithms for software repositories. *Science of Computer Programming* 78, 8 (Aug. 2013), 971–986. https://doi.org/10.1016/j.scico.2011.11.004

[11] Andrew Hall, Loren Terveen, and Aaron Halfaker. 2018. Bot detection in Wikidata using behavioral and other informal cues. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018). https://doi.org/10.1145/3274333

[12] Paul Jaccard. 1912. The distribution of the flora in the alpine zone. *New Phytologist* 11, 2 (February 1912), 37–50. https://doi.org/10.1111/j.1469-8137.1912.tb05611.x

[13] Eirini Kalliamvakou, Georgios Gousios, Kelly Blincoe, Leif Singer, Daniel M. German, and Daniela Damian. 2014. The Promises and Perils of Mining GitHub. In *International Conference on Mining Software Repositories (MSR)*. ACM, 92–101. https://doi.org/10.1145/2597073.2597074

[14] Jeremy Katz. 2018. Libraries.io Open Source Repository and Dependency Metadata (Version 1.4.0) [Data set]. http://doi.org/10.5281/zenodo.2536573.

[15] Erik Kouters, Bogdan Vasilescu, Alexander Serebrenik, and Mark G. J. van den Brand. 2012. Who's who in Gnome: using LSA to merge software repository identities. In *International Conference on Software Maintenance (ICSM)*. IEEE, 592–595. https://doi.org/10.1109/icsm.2012.6405329

[16] Sneha Kudugunta and Emilio Ferrara. 2018. Deep neural networks for bot detection. *Information Sciences* (2018). https://doi.org/10.1016/j.ins.2018.08.019 arXiv:1802.04289

[17] Carlene Lebeuf, Margaret Anne Storey, and Alexey Zagalsky. 2017. Software Bots. *IEEE Software* 35, 1 (2017), 18–23. https://doi.org/10.1109/MS.2017.4541027

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

[19] Shangqing Liu, Cuiyun Gao, Sen Chen, Lun Yiu Nie, and Yang Liu. 2019. ATOM: Commit Message Generation Based on Abstract Syntax Tree and Hybrid Ranking. 14, 8 (2019), 1–14. arXiv:1912.02972 http://arxiv.org/abs/1912.02972

[20] André N. Meyer, Thomas Fritz, Gail C. Murphy, and Thomas Zimmermann. 2014. Software developers' perceptions of productivity. *ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE)* (2014), 19–29. https://doi.org/10.1145/2635868.2635892

[21] Martin Monperrus, Simon Urli, Thomas Durieux, Matias Martinez, Benoit Baudry, and Lionel Seinturier. 2019. Repairnator patches programs automatically. *Ubiquity* 2019, July (2019), 1–12. https://doi.org/10.1145/3349589

[22] Mohammad Masudur Rahman and Chanchal K. Roy. 2014. An Insight into the Pull Requests of GitHub. In *Working Conference on Mining Software Repositories*. ACM, 364–367. https://doi.org/10.1145/2597073.2597121

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

[24] Damian A. Tamburri, Fabio Palomba, Alexander Serebrenik, and Andy Zaidman. 2019. Discovering community patterns in open-source: a systematic approach and its evaluation. *Empirical Software Engineering* 24, 3 (Jun 2019), 1369–1417. https://doi.org/10.1007/s10664-018-9659-9

[25] F. Thung, T. F. Bissyandé, D. Lo, and L. Jiang. 2013. Network Structure of Social Coding in GitHub. In *European Conference on Software Maintenance and Reengineering (CSMR)*. 323–326. https://doi.org/10.1109/CSMR.2013.41

[26] Jason Tsay, Laura Dabbish, and James Herbsleb. 2014. Let's Talk About It: Evaluating Contributions Through Discussion in GitHub. In *ACM SIGSOFT Symposium on Foundations of Software Engineering (FSE)*. ACM, 144–154. https://doi.org/10.1145/2635868.2635882

[27] Bogdan Vasilescu, Daryl Posnett, Baishakhi Ray, Mark G.J. van den Brand, Alexander Serebrenik, Premkumar Devanbu, and Vladimir Filkov. 2015. Gender and Tenure Diversity in GitHub Teams. In *Human Factors in Computing Systems (CHI)*. ACM, 3789–3798. https://doi.org/10.1145/2702123.2702549

[28] Mairieli Wessel, Bruno Mendes De Souza, Igor Steinmacher, Igor S. Wiese, Ivanilton Polato, Ana Paula Chaves, and Marco A. Gerosa. 2018. The power of bots: Understanding bots in OSS projects. *Proceedings of the ACM on Human-Computer Interaction* (2018). https://doi.org/10.1145/3274451

[29] Mairieli Wessel, Igor Steinmacher, Igor Wiese, and Marco A Gerosa. 2019. 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*. IEEE Press, 38–42. https://doi.org/10.1109/BotSE.2019.00018

[30] I. S. Wiese, J. T. d. Silva, I. Steinmacher, C. Treude, and M. A. Gerosa. 2016. 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)*. 345–355. https://doi.org/10.1109/ICSME.2016.13

[31] Marvin Wyrich and Justus Bogner. 2019. Towards an autonomous bot for automatic source code refactoring. *1st International Workshop on Bots in Software Engineering* (2019), 24–28. https://doi.org/10.1109/BotSE.2019.00015