Analyzing GitHub Users and Pull Requests to Identify Spam Bots

Shriya Sridhar*
Northern Arizona University
Flagstaff, Arizona, USA
ss3874@nau.edu

Mohammed Ali Sheriff Shaik*
Northern Arizona University
Flagstaff, Arizona, USA
ms4277@nau.edu

ABSTRACT

There is an increasing number of bots in GitHub that create pull requests, issues and other messages in the public repositories of GitHub. These bots disrupt the workflow of projects and decreases project code quality. This research paper aims to study these bots and develop a program to automatically detect them. It also tries to identify distinct features of these bots that separate them from legitimate user accounts. For this purpose, we developed a neural network model which utilises machine learning that predicts the nature of an account with 95.9% accuracy. We found that 19.3% of the user accounts we analyzed were spam bot accounts and that 11.33% of the pull requests were created by spam bots. Some distinguishing features such login ids, bio data, and e-mail addresses were identified by qualitative analysis. We conclude this paper with insights and future directions.

KEYWORDS

bots, spam, pull requests, issues, GitHub

1 INTRODUCTION

GitHub is one of the most popular repository hosting services and it provides version control using Git. It hosts more than 200 million repositories and has more than 65 million users worldwide. Of all the repositories, about 47 million are public repositories. Whenever developers want to contribute their own code to an existing project, they create a pull request. A pull request integrator then verifies that the new code, called commits, are beneficial to the project and would not cause any errors or conflicts with the main project. If it passes the tests and checks required, then the commits will be 'pulled' into the main branch of the project.

However, not every user account works for the betterment of projects and welfare of the community. Among the 65 million users, there are some user accounts that do not contribute anything useful to the community, but rather spam the publicly accessible open-source projects. The problem of weeding out spam pull requests from genuine ones creates additional workload for the pull request integrators, which may lead to delays in merging pull requests and a decrease in project code quality [12]. A research study conducted by Gong et al., (2019) revealed than nearly 21.5 percent of user accounts were malicious. These accounts may try to spread malicious codes in their commits, use public repositories as a medium of free advertising to their subscribers or try to damage the reputation of an open-source project by creating large quantities of unnecessary issues [7]. They may also try to imitate other eminent user accounts. The fake accounts become popular by taking undue advantage of the credibility of these users, thus harming the reputation of genuine developers. Another complication created by these spammers in pull requests and pull request comments is that they affect the interaction between other users since they are disruptive to the workflow. [13]

The purpose of this research paper is to analyze the pull requests and issues of several projects and identify the ones created by spam bots and study their nature. This would help to create better filters and prevent such spammers being able to create pull requests in the first place. In particular, we would like to address the following research questions:

RQ 1: How do we distinguish between a malicious bot from a genuine user?

This would help us to work towards the future goal of creating trustworthy filters that prevent spam bots from creating spurious pull requests but at the same time not restrict genuine users from contributing to projects.

RQ 2: What is the number of spam pull requests for a project compared to the total number of pull requests?

This will help us understand the depth and pervasiveness of the existing research problem and emphasize the need to study these spam bots.

RQ 3: Is there any consistent pattern in these bots? (e.g., names, geographical location, etc.)

Identifying common features between various spammers such as comment similarity, names, locations, etc. would greatly help to predict if a new user to a project is a spam bot or not.

The objective of this research paper is to collect the data on pull requests from GitHub, manually analyze the data for spam bots and then try to develop a program to automatically detect them. Using Python and the PyGithub library, scripts are written to extract the recent pull requests and issues from a few repositories and the data about their corresponding users, the comments on the pull request, its status, etc. The spammers are identified by comment similarity, checking whether they are blocked from using GitHub any further, the discussions by the project maintainers and review comments. The location, bio and other information on these users is also extracted. We would also like to identify the different methods in which they try to disguise themselves as regular users.

The results of this study could be used to create automated filters that would help reduce the time required by project managers to maintain a clean and pleasant developer community. This would in turn help foster better relationships and collaboration among the developers working for the project and improve the quality of the project.

2 RELATED WORK

GitHub often employs the use of benign bots in repositories to automate repetitive processes that are part of the distributed software development process. A representative ground-truth dataset was made by Mehdi Golzadeh et al. (2021), based on a manual analysis

*Both authors contributed equally to this research.

with high interrater agreement. It contains pull request and issue comments in 5,000 distinct Github accounts of which 527 have been identified as bots. [5] Bots are also identified to have some common characteristics such as the comment patterns, total number of comments of a commenter, etc. [6]

There have been several studies on bots which spam the various social networks such as Twitter, Facebook and e-mail that use methodologies and algorithms that can be applied to GitHub.

An algorithm based on inductive representation learning approach for detecting twitter bot accounts proposed by Seyed Ali Alhosseini et al., 2019 has obtained an improvement over other state-of-the-art algorithms by about 8 percent. [3] Other methods proposed include clustering-based methods to identify fraudulent voters during vote gaming attacks in the email platform [10] and intra-community oriented random walk strategies [9]. Some bots imitate human behaviour and while others do not post anything other than nonsensical messages. They can also be classified based on their intent into benign, neutral and malicious bots. [11]

3 METHOD

Based on the existing research on spamming accounts and bots, we use a machine learning approach to identify the spam bots.

3.1 Data Collection

For a machine learning model to function properly, we must first train it with sample data. After training the model on sample data, the model can tested against new data to check if it is working properly. For the purpose of training, we used a data set published by Gong et al., (2019). [8] The data set includes GitHub users who were registered by December 31, 2017. It utilises a data crawling method which carries out an ID-based random sampling. This gives an unbiased user dataset. The numeric IDs which do not have corresponding user accounts are ignored. The GitHub users API (i.e., https://api.github.com/user/ID) can be used to access data about the users. The authors of the dataset crawled for data of 10,667,583 randomly selected GitHub users including the demographic information, social connections and statistical indices of the activities shown on the user's profile page, such as the number of followings/followers/repositories from Jun. 20, 2018 to Aug. 27, 2018. Additionally, they also added data from GHArchive project2, which records the public GitHub event timeline since February 2012 using periodical crawling. [1]. Since GitHub allows access to only the latest 300 events or events from the previous 90 days of the users. In order to obtain a better collection of the user's historical events than one obtained just through crawling.

For the second research question, we use the GitHub API to crawl through selected projects and get the data about pull requests and the users involved. Using Python and the PyGithub library, we wrote scripts to extract the recent pull requests and issues from a few repositories and the data about their corresponding users, the comments on the pull request, its status, etc The spammers are identified by comment similarity, the discussions by the project maintainers and review comments. We need to filter out genuine pull requests and pull requests rejected due to other reasons such as error in code. The only pull requests that need to be counted are the ones which post malicious code in their commits, use the open-source projects as a medium of free advertising or scamming, etc. The validity of our data set is checked manually by the team members in order to avoid confirmation bias. The location, bio and other information on these users is extracted for the third research question.

3.2 Data Analysis

The data set from Gong et al., (2019) is a large data set consisting of more than ten million user accounts. This results in a file size of more than 50 gigabytes and is extremely cumbersome to handle using any software tool. Further, analyzing such a large data set is beyond the scope of this paper. Therefore, we split the data set into smaller files consisting of 1000 users each. Then, five files were randomly selected for analysis. Before, we implement the machine learning model, we need to first preprocess the data. This step takes care of any missing or corrupt data from the dataset. Using the scikit learn package, we replaced any missing numerical data with the average of that field and the categorical data with the mode of that data, i.e., which value occurs the most often. After taking care of the missing data, we split the dataset into training data and test data using the method described in the online course. (Eremenko et al., Machine learning A-Z Hands on Python and R in Data Science)) [4]

Once the data pre-processing is completed, we can implement a machine learning model can be designed and trained using the data. There are several models used in machine learning used in different situations. This research project is done using a neural network since the dataset is well-labeled and can utilise supervised learning.

After the training is complete, we can test the model's working by testing it on the test data. The tested model is then used to test the data set created for specific projects to help answer the research questions.

4 RESULTS

For the research questions RQ1 and RQ2, we can find answers from the output of the machine learning model. The neural network designed for this research project distinguished between the spam bots and genuine users with an accuracy of 95.9%. The distribution of True Positives (77.5%), True Negatives (18.4%), False Positives (3.2%), and False Negatives (0.9%) can be seen from the confusion matrix in Figure 1.

```
Confusion Matrix
[[775   9]
 [ 32 184]]
Accuracy
0.959
```

Figure 1: Confusion Matrix for the neural network.

The machine learning model predicted that there were 227 malicious user accounts among the 1000 accounts in the test set. Within this set of accounts, 193 were identified as bots with an accuracy of 95.9%, i.e. 184 accounts were actually spam bots. This indicates that 19.3% of the accounts in the test set were malicious spam bots. The spam bots contribute to 85.02% of the accounts that are classified

as malicious. On analyzing the pull requests in three repositories using the same model, we predicted that 17 out of 150 of the pull requests were created by spam bots, which amounts to 11.33% of the pull requests on an average.

For research question RQ3, we perform qualitative and quantitative analysis on our data sets.

The user data gives us several numerical values which can be easily used in quantitative analysis. Some examples of the data available for each user are the date the account was created, the date it was updated, the number of followers for that account, the number of people the user follows, the number of commits they have made, etc.

Figure 2 shows the box plots for the time difference between the time when the account was created and the account was last updated. The InterQuartile Range for the genuine users is higher than that of spam bots. There is a correlation of 0.211, which indicates a weak positive correlation between the two parameters. Hence, we could say that genuine users have a slightly higher possibility of updating their accounts more frequently than spam bots.

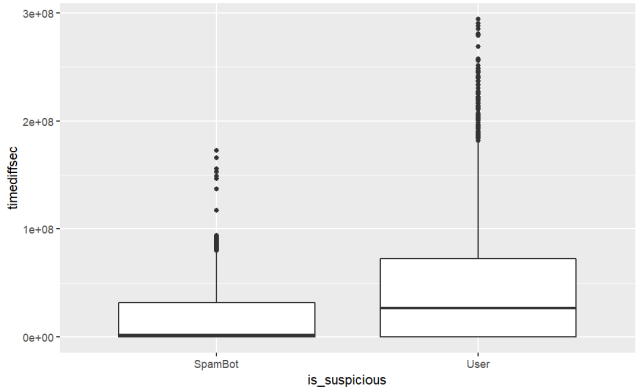

Figure 2: Box plots of the difference in time between creating and updating the accounts.

When analyzing the other numerical data available, we found that there is no significant correlation between the authenticity of an account and the other numerical parameters such as number of followers, commits, public gits, number of accounts followed by the user, etc. Figure 3 shows the box plots for three parameters - number of followers, following (i.e. number of accounts followed by the user), and commits. In all these plots, we can see that 75 percentile of the accounts tend to have zero followers, following and commits. This applies to both bots and genuine users. Therefore, we could not come to any concrete conclusion about how these factors are likely to be a certain value for spam bots.

We analyzed the email id data both quantitatively and qualitatively. First, we separate the data set into spam bots and genuine users and measure how many accounts had an email id published in their profile. The results can be seen in Figure 4 . Here we can see that almost an equal proportion of users (93.87%) and bots (95.32%) do not have an email id published in their profile. Hence the quantitative analysis does not give any useful results. However, when we read some of the email addresses of the spam bots, we found

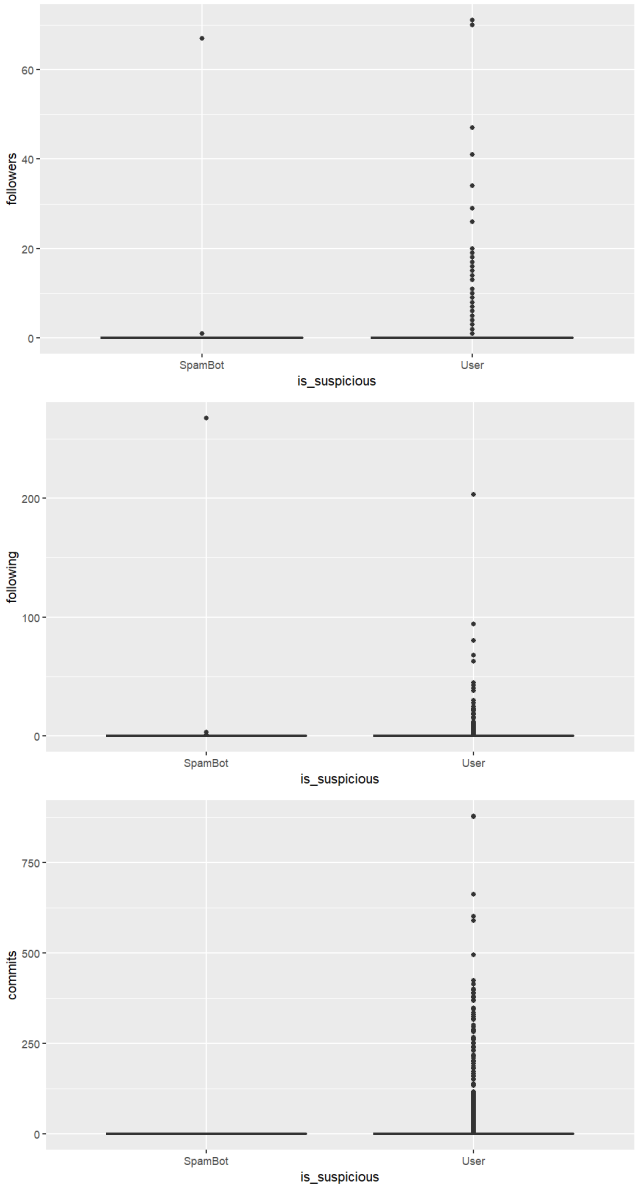

Figure 3: Box plots of the number of followers, following and commits.

that they were not from regular mail servers or organizational mail addresses. On further investigation, we found some of these mail servers on public blacklists for spam attacks. [2] Some examples of the email addresses of spam bots include "tovjpw@everytg.ml" and "epaces@everytg.ml". (Figure 5)

We also performed qualitative analysis on other categorical data such as names and bio's (user published bio-data on GitHub). Several of the spam bots had incomprehensible login id's and bio's that were a long string of several special characters or random combination alphabets or numbers. There were names such as

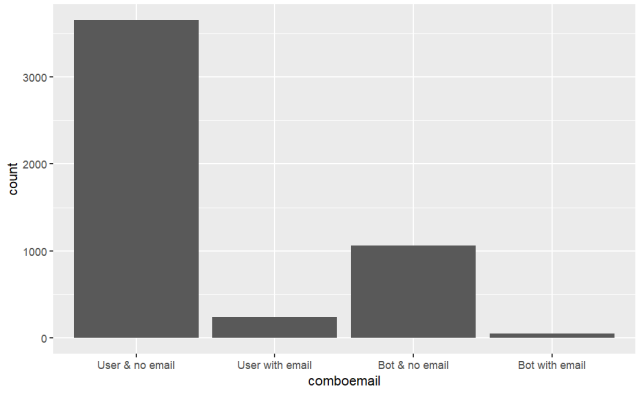

Figure 4: Box plots of the number of followers, following and commits.

"å“ªé‡Œè´ä¹°è¿·é¦™çƒŸ" and "q6d2l48011" and bio's that were undoubtedly advertising accounts such as "Best Body Pillow Reviews". (Figure 5)

5 DISCUSSION

Comparing our results with the results of other research studies on bots, [5] we can see that in a set of user accounts that are already identified as malicious accounts, bots are more likely to be found compared to the number of bots in general data set of all users. This is because legitimate accounts are more likely to be humans simply due to the sheer number of human accounts. On the other hand, people creating malicious spam accounts are presumably more inclined to create bots, since one person can easily create several bots and thereby have a much larger impact.

The creators of a number of these bots seem to think that the credibility of the bot accounts is improved by adding email addresses, company names and bio's, which is a logical conclusion. However, they do not put any effort into creating human readable ones. Several bio's for these accounts were incomprehensible. Some bot accounts had names and companies that seemed to be advertisements, such as "Sunshine coast dog behaviour training, dog training, dog training tips etc" and "Best Body Pillow Reviews".

6 LIMITATIONS AND THREATS TO VALIDITY

Our research is limited by the machine learning models even though they are just as useful. A number of spam accounts have features that are easily discernible to the human eye, but cannot be easily identified by software. It is also subject to human errors while trying to decide whether a user account is a bot or not. In some cases, it was hard to determine whether a pull request and the user were rejected because they were simple errors in the commits or because they were unrelated spam messages without knowing the context of the project.

7 CONCLUSION AND FUTURE WORK

In this paper, we investigated the presence of spam bots on GitHub and tried to determine how much a project is affected by bots. We developed a neural network model that uses machine learning to

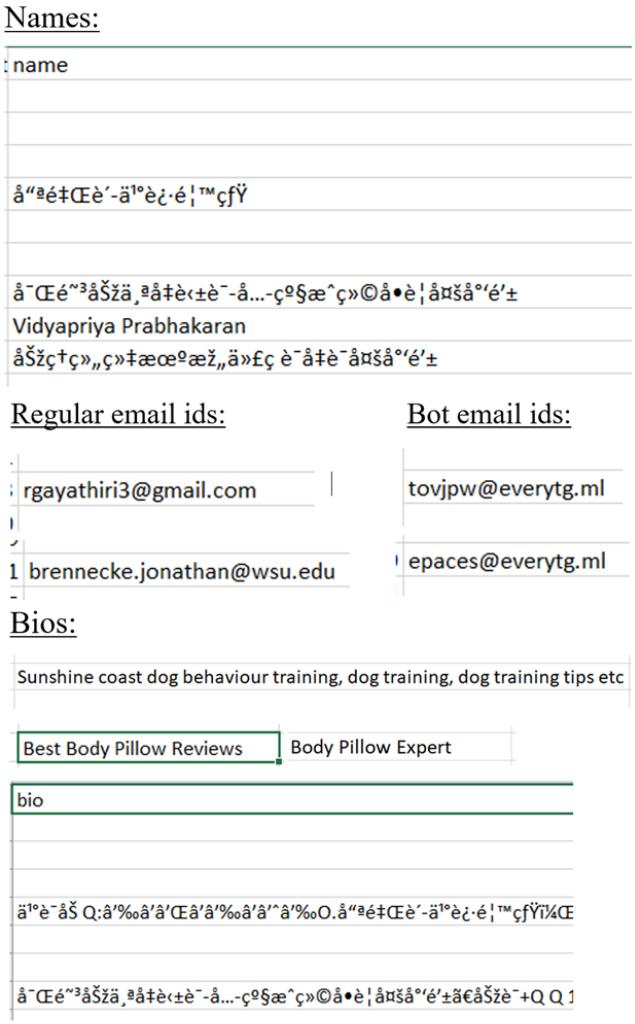

Figure 5: Some examples of the names, email id's and bio's, (mostly that of bots) from the data set.

predict whether a user account is a spam bot or a genuine user. We were able to determine that the project repository has 11.33% of pull requests from spam bots and the machine learning model was able to predict this 95.9% accuracy.

In future works, we would like to improve the accuracy of our model. We could test the efficiency of different machine learning models and develop an approach which has the best accuracy. With these improvements, this model could be used to further develop spam filters that would help reduce the burden of project integrators and maintain a healthy community in open source projects.

REFERENCES

[1] [n. d.]. https://www.gharchive.org/
[2] [n. d.]. https://cleantalk.org/blacklists/esr-everytg.ml
[3] Seyed Ali Alhosseini, Raad Bin Tareaf, Pejman Najafi, and Christoph Meinel. 2019. Detect Me If You Can: Spam Bot Detection Using Inductive Representation Learning. In *Companion Proceedings of The 2019 World Wide Web Conference* (San Francisco, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 148–153. https://doi.org/10.1145/3308560.3316504

[4] Kirill Eremenko, Hadelin de Ponteves, Ligency | Team, and SuperDataScience Support. [n. d.]. Machine learning A-Z (python amp; R in data science course). https://nau.udemy.com/course/machinelearning/

[5] Mehdi Golzadeh, Alexandre Decan, Damien Legay, and Tom Mens. 2021. A ground-truth dataset and classification model for detecting bots in GitHub issue and PR comments. *Journal of Systems and Software* 175 (2021), 110911. https://doi.org/10.1016/j.jss.2021.110911

[6] 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 *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops* (Seoul, Republic of Korea) *(ICSEW'20)*. Association for Computing Machinery, New York, NY, USA, 31–35. https://doi.org/10.1145/3387940.3391503

[7] Qingyuan Gong, Jiayun Zhang, Yang Chen, Qi Li, Yu Xiao, Xin Wang, and Pan Hui. 2019. Detecting Malicious Accounts in Online Developer Communities Using Deep Learning. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (Beijing, China) *(CIKM '19)*. Association for Computing Machinery, New York, NY, USA, 1251–1260. https://doi.org/10.1145/3357384.3357971

[8] Qingyuan Gong, Jiayun Zhang, Yang Chen, Yu Xiao, Xiaoming Fu, Pan Hui, Xiang Li, and Xin Wang. 2018. data.tar.gz.ab. In *A Representative User-centric Dataset of 10 Million GitHub Developers*. Harvard Dataverse. https://doi.org/10.7910/DVN/T6ZRJT/TSGW3H

[9] Phu Pham, Loan T.T. Nguyen, Bay Vo, and Unil Yun. 2022. Bot2Vec: A general approach of intra-community oriented representation learning for bot detection in different types of social networks. *Information Systems* 103 (2022), 101771. https://doi.org/10.1016/j.is.2021.101771

[10] Anirudh Ramachandran, Anirban Dasgupta, Nick Feamster, and Kilian Weinberger. 2011. Spam or Ham? Characterizing and Detecting Fraudulent "Not Spam" Reports in Web Mail Systems. In *Proceedings of the 8th Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference* (Perth, Australia) *(CEAS '11)*. Association for Computing Machinery, New York, NY, USA, 210–219. https://doi.org/10.1145/2030376.2030401

[11] Stefan Stieglitz, Florian Brachten, Björn Ross, and Anna-Katharina Jung. 2017. Do social bots dream of electric sheep? A categorisation of social media bot accounts. *arXiv preprint arXiv:1710.04044* (2017).

[12] Erik Van Der Veen, Georgios Gousios, and Andy Zaidman. 2015. Automatically prioritizing pull requests. In *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories*. IEEE, 357–361.

[13] Mairieli Wessel and Igor Steinmacher. 2020. The Inconvenient Side of Software Bots on Pull Requests. In *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops* (Seoul, Republic of Korea) *(ICSEW'20)*. Association for Computing Machinery, New York, NY, USA, 51–55. https://doi.org/10.1145/3387940.3391504