Exploring Social Contagion in Open-Source Communities by Mining Software Repositories

Zakariyah Shoroye, Waheeb Yaqub, Azhar Ahmed Mohammed, Zeyar Aung, and Davor Svetinovic(✉)

Institute Center for Smart and Sustainable Systems,
Department of Electrical Engineering and Computer Science,
Masdar Institute of Science and Technology, P.O. Box 54224, Abu Dhabi, UAE
{zshoroye,wyaqub,amohammed,zaung,dsvetinovic}@masdar.ac.ae

Abstract. The emergence of data mining has helped improve our understanding of social contagion in networks. The magnitude of contagion in networks such as Facebook and Twitter has been studied in detail. Study of social contagion in software development networks can provide interesting findings in order to increase return on investment and improve quality of software. For example, developers could be incentivised and the time to start an open-source projects optimized by analyzing social contagion in online repositories. In this study, open-source repositories' data was analyzed and it was observed that highly followed developers tend to attract more contributors to a project. Also, the number of commits was aggregated on a yearly basis to provide insight into the question of the best time to start a project. GitHub online repository data was collected since its inception until 2014. The number of commits in the online repository was found to follow the "power law". By considering only large projects, a correlation between the number of followers a user has and the contagion rate of their commits was observed. Understanding these questions and social contagion can help software companies to leverage on the open-source community and improve their own internal social networks.

Keywords: Software repositories · Social network analysis · Social contagion

1 Introduction

Collaborative software development is gaining momentum in the competitive market as companies have to innovate faster and cut costs. Recently a new trend in software development is emerging in which companies across industries are collaborating to develop common code bases, which can then be extended. In March 2014, Linux Foundation carried out an invitation-only survey to the companies that included Cisco, Fujitsu, HP, IBM, Intel, Google, and Samsung among others [1]. Exactly 686 software developers took part in this survey from organizations that make 500 million dollars or more in annual revenue. The findings of the study showed that open-source development is no longer driven by the love for programming but by money. 35 % of the developers still contribute to open-source projects in their free time but 44 % of the developers admitted that the job requirements were the reason they started contributing [1,2]. Thus, the recent trend is that companies encourage their staff to contribute to open-source projects. As such, commercial development interest and influence in open-source projects is on the rise.

This approach of collaborative development is a win-win situation for both the organizations and the developers. Half of the managers surveyed were of the opinion that collaborative development gives them the flexibility to innovate and help transform the industry. Cloud computing, mobile devices, the Internet of Things, software-defined-networking were identified as the top five that would see increasing use of collaborative development practices [1]. The Linux Foundation report emphasizes the use of open-source collaboration and hence it is imperative to answer the following questions:

- When is the best time to start a new open-source project?
- Who should be given the incentive to start new open-source project?

The study of the social contagion in the open-source development networks can help us understand and answer such questions. The answers can in turn help software companies to leverage the open-source community and improve their own internal social networks.

In this study, we try to answer these questions by analyzing GitHub [3] data. GitHub is a web-based Git repository hosting service that offers source code management (SCM) and reversion control of projects in distributed fashion. As of 2014, GitHub is the largest open-source projects host [4]. Our study showed that the best starting time for a project is between June and August. We also found that there is a correlation between the followers a user has and the number of such followers that contributed to a particular project after their contribution. This makes such developers potential targets for incentivisation.

2 Related Work and Research Method

Social networks have been widely studied by researchers for its various characteristics such as contagion, geography, centrality, etc. For example, Ugander et al. [5] studied the structural diversity in social contagion on the Facebook network. The studies on the veracity of GitHub [3] repository data for research purposes has shown that although it is rich in data, it has some underlying issues which developers have to be aware of [6]. These include the presence of discrepancies in the pull commit data and that a lot of the projects are small and personal. Social networks are being more and more leveraged in software development [7].

GitHub data is either accessible through its application program interface (API) which limits the number of queries requested per hour to 60 records if using an unauthenticated account and 5,000 records for an authenticated account.

Given the scale of data required to be accessed by us, 5,000 records every hour was merely not enough. Therefore we obtained all of GitHub using GHTorrent [8]. The total size of the data was 40 GB.

GHTorrent provides archives of close to 600 million rows of MySQL data which can be downloaded as a *.sql* file that can be used to build a native database. GHTorrent provides data in a very organized manner dividing it into several tables. The two tables used in this study are *commits* and *followers*. The size of *commits* table is 39 GB with almost 1.5 billion records and that of *followers* table is 397 MB with 5 million records.

First, we tired to categorize the repositories based on their respective sizes. As of 2014, there are 16.7 million repositories in GitHub with 3.4 million users. These have varying number of commits with some having zero commits and some having more than 400,000 commits. Contrary to the method used by Kolassa et al. [9], in which the number of contributors was chosen as a representative of the size of GitHub repository, we chose the number of commits to represent the size of the project. This is because there can be some projects with few authors but a large number of commits. Furthermore, while Kolassa et al.'s method cannot observe the disparity between the two projects with the same number of developers but a notably different number of commits, our proposed commit-based method can do so. As we mentioned before, large software companies are adapting open-source and collaborative approach to develop projects in order to limit the cost of projects, etc. Later we will show that large projects are the ones that fall in the long tail of the "power law distribution" where the number of commits is the representative of project size.

Before trying to fit the data to any distribution, it can be assumed that the empirical data follows normal (a.k.a. Gaussian) distribution. The normal distribution is a natural guess since it is heavily used in natural sciences [10]. From the "central limit theorem" it is deduced that if we take small sequence of small independent random variables then their average will be distributed across normal distribution. On the other hand a study on the distribution of web links showed something different from central limit theorem [10]. The number of web pages that have m links is proportional to $\frac{1}{m^2}$. Such extreme observations are better captured using the power law distribution.

Since population in large cities, biological extinction, genetic networks or the World Wide Web all follow the power law [11,12] and so does the sizes of commits in GitHub [13], we hypothesize that the number of commits would also follow the power law. We are interested in the projects that fall in the heavy tail of the power distribution. We believe that these projects represent the scale and size of repositories similar to projects that companies are willing to collaborate on. The total number of commits for more than 6.5 million repositories was used as an input to fit the power law distribution. We followed the method introduced by Clauset et al. in [14] to fit the empirical data to power law distribution. In [14,15], it was shown that when testing empirical data, creating the cumulative distribution function and fitting the resulting function to the linear form by least-squares linear regression is subject to systematic and potentially large errors. To avoid such errors incurred in [9,13], the steps listed below were followed for analysing the data.

1. Estimate the x_{min} and α using bootstrap method to get handle on parameter uncertainty.
2. Compute the goodness of fit between the empirical data and the power law using goodness-of-fit test, which generates a p-value that represents the soundness of the hypothesis. These tests depend on the distance between distribution of empirical data and the hypothesized model.
3. Eliminate the possible competing distributions such as exponential or log normal.

3 Results and Discussions

3.1 Power Law Distribution

The histograms in Figs. 1 and 2 depict the uncertainty in estimated parameters of x_{min} and α. Based on the bootstrapping method proposed by Clauset et al. [14], we ran and estimated the parameters for 100 simulations. The frequency of parameters x_{min} and α was 384 and 2.372, respectively. Based on the hypothesis that data is generated from the power law distribution and $p = 0.82 > 0.1$, we cannot rule out the power law. The p-values examined for other distributions were less than that of the power law. Hence, we found that the power law is the best fit for this data.

The data was fitted to the power law distribution. Based on this fitting, we were able to categorize the size of projects based on its commits size. This helps to avoid some of the shortcomings of using the GitHub data as discussed in Kalliamvakou et al. [6].

3.2 Best Time to Start New Projects

Close to three million projects were launched on GitHub during its five years of existence: 2008 until 2013. We limited the retrieved data until the end of 2013 because complete data for year 2014 was not available at the time of conducting our experiments. Figure 3 shows the monthly commits activity of these projects. The values were normalized to scale uniformly across all years. The plot shows a gradual increase in the commit activity with a slight decrease in the month of December. The decrease in the commit activities in December (except for 2011) is possibly because of the festive season. Figure 3 shows that the best time to start the project on GitHub would be from June to August as across most of the years the commit activity continues to increase during this time. This can be attributed to the fact that the summer vacation in most parts of the world starts during that period and consequently developers find more free time. It would thus be imperative to start the project at the peak times as, based on basic probability studies, it is most likely going to get the much needed attention.

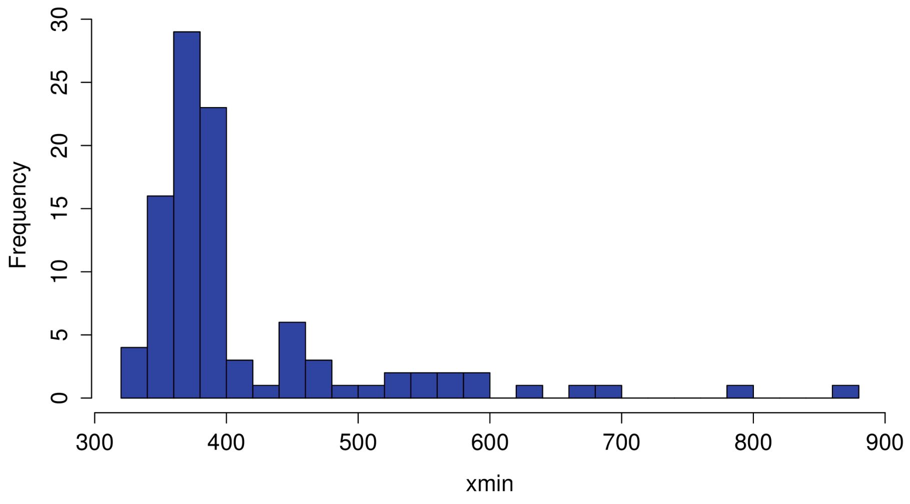

Fig. 1. Frequency of different values of x_{min}. Y-axis represents the frequency of each possible X-min values from the bootstrapped method. X-axis represents the minimum possible values after which the power law holds.

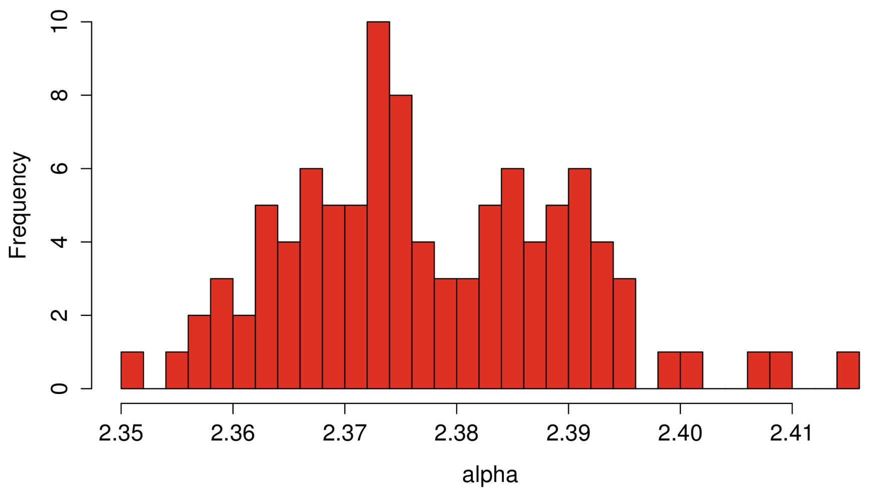

Fig. 2. Frequency of different values of alpha. Y-axis represents the frequency of each possible α values from the bootstrapped method.

3.3 Social Contagion in Open-Source Software Development

We were also able to identify those who should be given an incentive to start a project because of their high contagion rate. During the analysis, we considered only the large projects using the criteria established above using the power law. We analyzed 443 projects that had more than 10,000 commits. This is because

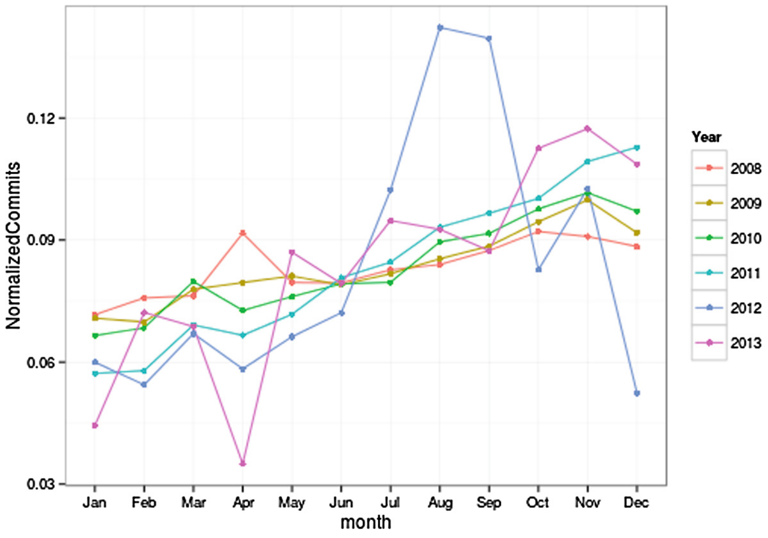

Fig. 3. Normalized number of commits for all months from 2008 to 2013.

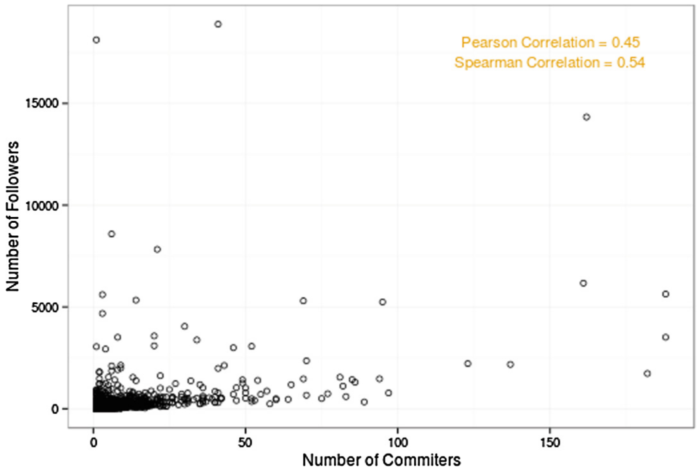

Fig. 4. Correlation of number of followers vs. number of followers who are committed.

smaller projects might skew the data as the probability of a developer follower committing in the same project as him decreases if only a few developers are partaking in the project. The threshold of 10,000 commits was thus chosen. We were able to compare over 70,000 unique developers. Figure 4 shows the correlation between the number of followers a user has and the followers who committed in a project after she has committed. We found a Spearman correlation of 0.54 and a Pearson correlation of 0.45. Cohen provides the following guidelines for

Pearson correlation in social sciences: *small* if the value is 0.1, *moderate* if it is 0.3 and *large* if it is 0.5 [16,17]. Hence, we found that the contagion factor is highly prevalent in the GitHub open-source community. This indicates that the developers are likely to participate in projects where their followees have earlier participated. Thus, coding can be considered as contagious in the follower network of GitHub.

4 Limitations and Future Work

In this study only two variables were considered: the number of commits and the number of followers. Our assumptions can be made stronger by considering other variables such as different criteria on the size of the project, bi-directional following, time-relative following trends, etc. Another interesting direction would be to analyze the behavior of the GitHub users in correlation with their behavior on the other social networking sites, e.g., Twitter. A good question to answer would be how many of the followers on Twitter have joined GitHub. We used the number of commits made as a measure of contribution to the project. The use of other actions like pull requests might yield additional insights. Our preliminary results show that there exists a correlation between the number of commits made by the followers and the total number of followers. The correlation however does not show causation and hence a causal analysis is necessary.

5 Conclusion

In this study, we tried to answer two interesting questions derived from the Linux Foundation report [1]. The answers to these questions might help companies decide when is the best time to start a project and whom to incentivise to start and lead the project. Our preliminary analysis showed that starting a project between June and August is probably optimal to leverage free time and social contagion within open-source community networks. We also found that there is a correlation between the followers a user has and the number of such followers that commit to a particular project after she committed. This is an evidence of the existence of the social contagion within the development networks. This preliminary study provides us with sufficient support for extending the study of the social contagion in the open-source development communities, especially in the early stages of development, e.g. [18,19], and its further incorporation within our main research initiative [20].

References

1. Linux Foundation. Collaborative development trends report. Technical report (2014)
2. Vaughan-Nichols, S.J.: The new open source motivation, Show me the money (2014)

3. GitHub: Build software better, together (2014)
4. Gousios, G., Vasilescu, B., Serebrenik, A., Zaidman, A.: Lean GHTorrent: GitHub data on demand. In: Proceedings of the 11th Working Conference on Mining Software Repositories (MSR), pp. 384–387. ACM (2014)
5. Ugander, J., Backstrom, L., Marlow, C., Kleinberg, J.: Structural diversity in social contagion. Proc. Natl. Acad. Sci. **109**(16), 5962–5966 (2012)
6. Kalliamvakou, E., Gousios, G., Blincoe, K., Singer, L., German, D.M., Damian, D.: The promises and perils of mining GitHub. In: Proceedings of the 11th Working Conference on Mining Software Repositories (MSR), pp. 92–101. ACM (2014)
7. Adepetu, A., Ahmed, K.A., Al Abd, Y., Al Zaabi, A., Svetinovic, D.: Crowdrequire: a requirements engineering crowdsourcing platform. In: AAAI Spring Symposium: Wisdom of the Crowd (2012)
8. Gousios, G.: The GHTorrent dataset and tool suite. In: Proceedings of the 10th Working Conference on Mining Software Repositories (MSR), pp. 233–236. ACM (2013)
9. Kolassa, C., Riehle, D., Salim, M.A.: A model of the commit size distribution of open source. In: van Emde Boas, P., Groen, F.C.A., Italiano, G.F., Nawrocki, J., Sack, H. (eds.) SOFSEM 2013. LNCS, vol. 7741, pp. 52–66. Springer, Heidelberg (2013)
10. Easley, D., Kleinberg, J.: Networks, Crowds, and Markets: Reasoning About a Highly Connected World. Cambridge University Press, Cambridge (2010)
11. Bak, P.: How Nature Works. Oxford University Press, Oxford (1997)
12. Barabási, A.-L., Albert, R.: Emergence of scaling in random networks. Sci. **286**(5439), 509–512 (1999)
13. Arafat, O., Riehle, D.: The commit size distribution of open source software. In: Proceedings of the 42nd Hawaii International Conference on System Sciences (HICSS), pp. 1–8. IEEE Press (2009)
14. Clauset, A., Shalizi, C.R., Newman, M.E.J.: Power-law distributions in empirical data. SIAM Rev. **51**(4), 661–703 (2009)
15. Goldstein, M.L., Morris, S.A., Yen, G.G.: Problems with fitting to the power-law distribution. Eur. Phys. J. B: Condens. Matter Complex Syst. **41**(2), 255–258 (2004)
16. Cohen, J.: Statistical Power Analysis for the Behavioral Sciences. Routledge Academic, New York (2013)
17. Cohen, J.: A power primer. Psychol. Bull. **112**(1), 155–159 (1992)
18. Svetinovic, D., Berry, D.M., Day, N.A., Godfrey, M.W.: Unified use case statecharts: case studies. Requir. Eng. **12**(4), 245–264 (2007)
19. Svetinovic, D.: Architecture-level requirements specification. In: STRAW, pp. 14–19 (2003)
20. Svetinovic, D.: Strategic requirements engineering for complex sustainable systems. Syst. Eng. **16**(2), 165–174 (2013)