Towards Automatically Identifying Paid Open Source Developers

Maëlick Claes
M3S, ITEE, University of Oulu, Finland
maelick.claes@oulu.fi

Mika Mäntylä
M3S, ITEE, University of Oulu, Finland
mika.mantyla@oulu.fi

Miikka Kuutila
M3S, ITEE, University of Oulu, Finland
miikka.kuutila@oulu.fi

Umar Farooq
M3S, ITEE, University of Oulu, Finland
umar.farooq@oulu.fi

ABSTRACT

Open source development contains contributions from both hired and volunteer software developers. Identification of this status is important when we consider the transferability of research results to the closed source software industry, as they include no volunteer developers. While many studies have taken the employment status of developers into account, this information is often gathered manually due to the lack of accurate automatic methods. In this paper, we present an initial step towards predicting paid and unpaid open source development using machine learning and compare our results with automatic techniques used in prior work. By relying on code source repository meta-data from Mozilla, and manually collected employment status, we built a dataset of the most active developers, both volunteer and hired by Mozilla. We define a set of metrics based on developers' usual commit time pattern and use different classification methods (logistic regression, classification tree, and random forest). The results show that our proposed method identify paid and unpaid commits with an AUC of 0.75 using random forest, which is higher than the AUC of 0.64 obtained with the best of the previously used automatic methods.

ACM Reference Format:
Maëlick Claes, Mika Mäntylä, Miikka Kuutila, and Umar Farooq. 2018. Towards Automatically Identifying Paid Open Source Developers. In *MSR '18: MSR '18: 15th International Conference on Mining Software Repositories , May 28–29, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3196398.3196447

1 INTRODUCTION

As open source software simplifies the acquisition of large amounts of data related to software development, it has been the subject of numerous repository mining studies. Despite the large focus on open source software in the research community, and long existing studies showing that large companies invest money in open source development [1, 2], many empirical studies still often assume that the software industry is split between open source projects developed mainly by non-paid hobbyist developers [3–10], and closed-source projects developed by hired developers paid by companies.

Because of the differences between volunteer work and paid work, a common threat to validity in repository mining studies relying on open source data, is that findings may not hold for traditional closed-source projects [11] and vice versa. This is particularly true when the research focuses on social, human or behavioral aspects of software development. Another related issue is that some findings about open source software may only hold when the developers are volunteer. For example, Herraiz et al. [12] found that the onion model doesn't apply to core-developers working for companies but only for volunteers. There are also fundamental differences between paid and volunteer work as volunteer work tend to be performed by individual with greater well-being and increase well-being at the same time [13] and that a motive for volunteer work is different from hired work [14].

Because of earlier results we obtained studying working hours of Mozilla developers [15, 16], we believe there are some important differences between paid closed-source, paid open source software and volunteer open source software development. In order to increase the industrial relevance, generalizability to correct population and credibility of mining software repositories studies we provide a first step towards (semi-)automatically identifying whether developers are paid or not.

To achieve this goal, we rely on code repository meta-data and manually extracted information about developer employment status in order to identify which characteristics better predict whether a developer or a commit should be considered as "hired" or "volunteer". We focus on the Mozilla projects because they contain a large amount of developers paid by the Mozilla Foundation. This simplifies the process of manually gathering employment status of developers needed to train and test our classifiers. We then compare our results with the ones obtained by applying techniques used in the literature we are aware of [15, 17].

2 RELATED WORK

The most reliable technique to identify paid open source developers is gathering data manually by searching for personal information on web search engines [18]. Indeed, it is common for open source developers to advertise themselves and mention for which company they work on websites such as LinkedIn, GitHub or their personal websites. However, it is long and laborious and cannot scale to

MSR '18, May 28–29, 2018, Gothenburg, Sweden

thousands of developers. Unfortunately, the only alternative methods we found in the existing literature are the use of the time of activity [17] and e-mail address domain names [15].

Riehle et al. [17] studied the amount of paid work in various open source projects, including the Linux kernel. They considered that any commit made on a weekday between 9am and 5pm was paid work and work outside this time period was voluntary work. However, this technique completely ignores unemployment, flexible working hours and overwork. Open source developers can potentially be students or unemployed and thus contribute outside the what is considered as regular office hours. Additionally, developers can also have flexible or irregular working hours. These two issues can potentially hinder the precision and the recall of the results. Moreover, even developers with regular office hours can be subject to overwork. This can also be a non negligible source of false negatives.

In our previous work [15], we studied abnormal working hours in Mozilla Firefox. In order to easily identify paid developers, we used the email address provided in the commit meta-data and considered that every developer using an email registered at mozilla.com is a Mozilla employee. However, the technique is not perfect as we found Mozilla employees who do not use their Mozilla e-mail address in commits. On the other hand, active volunteers could also own and use a Mozilla email address.

3 METHODOLOGY

3.1 Data extraction

To answer the research questions, we mined data from Mozilla's Mercurial repositories[1]. We extracted the history of all commits using *git-remote-hg*[2] and the *GrimoireLab* tools[3].

In addition, we extracted issue comments from Mozilla's Bugzilla repository[4] (i.e., the database containing reported issues, such as bug reports or feature requests). In order to identify which commits from Mozilla's code repositories are related to which Mozilla subproject (e.g., Firefox for Desktop vs. Firefox for Android), we linked commit messages to the corresponding issue report by looking for an issue identifier in a given message. Out of the 396,180 extracted commits, 330,078 were successfully linked to a bug issue. After linking, we then filtered the commits to only keep the ones related to the following major *products*: *Firefox*, *Core*, *Firefox OS*, *Firefox for Android*, *Thunderbird* and *SeaMonkey*.

In order to study the individual developers, we performed a basic merging of the different authors' identities. We first cleaned the name and email used in the version control system's author field. Then we grouped together identities using the same name or email addresses. Finally, two of the authors manually checked the result in order to avoid any false positive.

In order to obtain ground truth about hired developers at Mozilla, we ran a manual background check for the developers with more than 100 commits, leaving us with 391 developers (out of 2,755). 261 (66.8%) of those developers were hired by Mozilla and made 87% out of all the Firefox commits.

[1]https://hg.mozilla.org
[2]https://github.com/felipec/git-remote-hg
[3]https://grimoirelab.github.io/
[4]http://bugzilla.mozilla.com

We also collected periods of employment, which we could retrieve for 212 of the hired developers, and found 16 developers who contributed both as volunteers and hired developers. 6 of them have committed more as a volunteer than as a hired and were considered as volunteers in the data set. For the others, the amount of commits made as a volunteer was relatively small (less than 20%) and we considered them as hired developers.

3.2 Classification algorithms and metrics

First, we use three algorithms to predict developer employment status. First we use logistic regressions because our predicted variable is only binary (hired or volunteer). Then, we use classification and regression tree (CART) models using the *rpart* R package [19]. Finally, we also use random forests. While random forest usually gives better results than a single decision tree, *rpart* allows us to visualize the decision tree and better understand which features better predict employment status. We built all of these models using a repeated 10-fold cross validation with the *caret* R package [20].

Second, we define a set of metrics about developer commit activity that will be used as features for our classification algorithms. These metrics are summarized in Table 1. For the metrics related to the time of the day or time of the week, we took into account the timezone given in the commit timestamp and thus consider the developer's local time.

Table 1: List of metrics computed for each developer

Metric	Description
period	n of days between first and last commit
days	n of days with at least one commit
weeks	n of weeks with at lest one commit
timediff	median of days between successive commits
commits	n of authored commits
loc per commit	median loc modified per commit
weekend	% of commits during the weekend
night	% of commits between midnight and 6am
morning	% of commits between 6am and noon
afternoon	% of commits between noon and 6pm
evening	% of commits between 6pm and midnight
office	% of commits between 8am and 5pm
most active hour	h of day with highest amount of commits
beginning regular	h of day when weekday activity starts
length regular	Length of weekday activity period
end regular	h of day when weekday activity ends

4 EMPIRICAL ANALYSIS

In this section, we first build logistic regression, decision tree and random forests models using a 10-fold cross validation. Secondly, we compare these results with simpler techniques used in the literature. Then, we use the information from the most active developers to predict whether individual commits are paid. Finally, we discuss the differences obtained between the different models built and the different simple techniques used in the literature.

4.1 Prediction of employment status of individual developers

First we ran our three classification algorithms with a 10 repeated 10-fold cross validation using all the metrics defined in Table 1 as features. Table 2 reports the ROC area under the curve, precision and recall of the three different models.

Table 2: Predicting employment status using all metrics

Classifier	ROC AUC	Precision	Recall
logit	0.73	0.751	0.884
rpart	0.65	0.735	0.879
randomforest	0.77	0.767	0.859

In addition, we also ran the same three algorithms using the subset of features not related to the period or amount of activity of a developer. We left out *commits*, *days*, *weeks* and *period* from Table 1. Because we want to be able to train a model with a relatively small number of developers, potentially the most active, and still be able to predict the outcome for less active developers. The performance metrics, using a 10 repeated 10-fold cross validation, are reported in Table 3.

Table 3: Predicting author employment status without using metrics related to the number of commits or periods of activity.

Classifier	ROC AUC	Precision	Recall
logit	0.68	0.732	0.897
rpart	0.63	0.73	0.873
randomforest	0.75	0.756	0.881

Comparing Table 2 and Table 3, we observe that the performances are quite similar for *rpart* and *randomforest*. This means that the importance of the number of commits, days, weeks and period of activity is not critical to predict the employment status of a developer. It also means it should be possible to train a model with only a small number of developers and still be able to predict the employment status of the other developers

4.2 Comparison with simpler automatic techniques

In order to assess the performance of the three models, we computed the same performance metrics obtained with simple automatic techniques used in the existing literature to detect paid developers. These are reported in Table 4.

First, because only one third of all the considered authors are volunteers, we computed the performance metrics when considering all authors as hired (*allhired*). Then, we relied on the domain name of the email address used by developers. Like in our previous study about developer working hours in Firefox [15], we considered as paid, the developers who made at least one commmit with a mozilla.com email address (*email*).

Finally, we also tried to consider the approach used by Riehle et al. [17]. They considered that hired developers are developers for which at least 95% of their commits were made during regular office hours (9am-5pm) and volunteers less than 5% of their commits. All developers having made more than 5% but less than 95% during the same time interval were considered as having a mixed status.

Table 4: Predicting employment status without machine learning

Classifier	ROC AUC	Precision	Recall
allhired	0.5	0.668	1
email	0.64	0.772	0.701
95%officehours	0.5	NA	0
5%officehours	0.5	0.668	1
50%officehours	0.63	0.75	0.805

Because we only consider a developer as hired or volunteer, we computed the performance metrics for the case where hired developers are those with more than 95% of their commits during office hours (*95%officehours*), and the case case where hired developers are those with more than (*5%officehours*). These are equivalent as considering either all developers as volunteers or as hired. Indeed all considered developers had between 5% and and 95% of their commits made during regular office hours. In addition we also consider the case where a hired developer is a developer with more than 50% of commits during office hours (*50%officehours*).

Overall, all of these simple automatic techniques give performances below the random forest classifier in terms of AUC.

4.3 Predicting paid commits

To detect individual paid commits instead of individual paid developers, we build our *logit*, *rpart* and *randomforest* models using the information from the 67 most active developers who have contributed, altogether, at least 50% of all of the considered commits. We tested the performance of these models on all the commits and on the commits of the least active developers (Table 5). We also computed the performance metrics when considering as paid commits, all the commits (*allpaid*), the commits made by a developer with at least a Mozilla email address (*email*) as done in our previous study [15], and the commits made during regular office hours (*officehours*) as done by Riehle et al. [17].

4.4 Discussion

First, when building our models to predict developer employment status with a repeated k-fold cross validation, we find that *randomforest* performs better than the two other techniques, particularly *rpart*. On the other hand, contrarily to *randomforest*, both *logit* and *rpart* models allow us to have some insights about the metrics that better predict employment status. Fig. 1 depicts the decision tree built by *rpart* using all metrics as features.

We observe that the first splitting node of the decision tree produced by *rpart* uses the share of commits made during the weekend. To further refine the model, it uses the median *time difference* between successive commits and the length of the *period* of activity.

Similarly, the most significant coefficient of the *logit* model is the share of weekend commits (5.94). However the *time difference* between commits and the length of the *period* of activity have much less influence (respectively 0.36 and -0.0004). The other most

Table 5: Models built with the most active developers and tested on all commits (top) and on commits of the least active developers (bottom).

Classifier	ROC AUC	Precision	Recall
logit (all)	0.683	0.86	0.865
rpart (all)	0.766	0.843	0.954
randomforest (all)	0.87	0.879	0.957
allpaid (all)	0.5	0.781	1
email (all)	0.594	0.853	0.672
officehours (all)	0.467	0.743	0.414
logit (least active)	0.608	0.781	0.714
rpart (least active)	0.724	0.768	0.935
randomforest (least active)	0.69	0.754	0.904
allpaid (least active)	0.5	0.704	1
email (least active)	0.639	0.813	0.702
officehours (least active)	0.448	0.644	0.393

important metrics in the *logit* model are the share of *afternoon* (-1.99), *morning* (-1.08), *office* (-0.67) and *night* (0.57) activity.

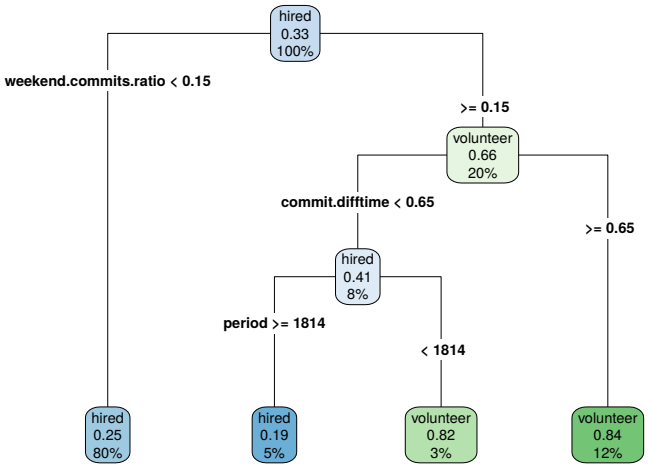

Figure 1: *rpart* decision tree built using all metrics. Each node of the tree contains the ratio of volunteer developers and the percentage of developers considered.

This shows that determining whether a developer is paid, can be done by relying mostly on the usual commit time of the day or the week and gives better results than with the simple automatic techniques used in the literature. Although using email addresses offers a slightly better precision than any of our methods, there are still a large number of volunteer developers who uses an e-mail address provided by the Mozilla foundation. Moreover, this technique has a recall two times worse than with all of our machine learning models as 30% of the hired developers have never used a Mozilla email address.

Using a threshold of 5% or 95% of activity during office hours don't work in the case of Mozilla because all contributors, paid or not, committed during office hours between 10.4% and 94.7% of the time. In the best case, using a threshold of 5% is equivalent to considering that all developers are paid by Mozilla. In the end, using a threshold of 50% gives an AUC (0.63) close to the one achieved with email addresses (0.64) but still far from the AUC of 0.75 obtained with *randomforest*.

Finally, when trying to predict paid commits instead of paid developers, while the AUC of email addresses stay unchanged, relying on regular office hours gives the worst results with an AUC below 0.5 (random guessing) because Mozilla's paid developers work often outside regular office hours.

For paid commits, we observe that not only *logit* is now performing worse than the two other models. On the other hand, *randomforest* is now performing worse (AUC 0.69) than *rpart* (AUC 0.72).

5 THREATS TO VALIDITY

In order to identify developers hired by Mozilla, we manually looked for information online. Although it allows us to identify a large amount of the most active developers as hired by Mozilla, we might have missed developers who do not share online their CV.

We merged developers' identities using a very basic identity merging technique. We manually checked for false positives in order to avoid merging the work pattern of two developers as a single one and thus overestimating their amount of activity. However, there might be false negatives remaining, which could be particularly problematic in the case of developers using their work e-mail during office hours and personal emails outside office hours.

Our study only includes open source projects from Mozilla. The results obtained are specific to the organization's culture and developer habits (external validity). Thus we cannot guarantee that our data set would be representative of the entire open source industry.

6 CONCLUSION AND FUTURE WORK

In this paper, we have taken an initial step towards semi-automatically recognizing paid development in open source projects. It appears that the best predictors are weekend and evening work but also time difference between commits. While our models beat simple automatic classifiers, these results are only an initial step as the current techniques still can't compete with manually gathered data.

In the future, we plan to include email-address and commit message content analysis with natural language processing to improve our predictions. We will also add other data sources, such as issue tracker, to improve the amount of information about each developer. Furthermore, we think giving our machine learning more information through automated web-scraping as it is a promising way to further enhance our ability to detect paid development in open source project and provide more precise results for future MSR studies.

Finally, the main limitation of our study is the lack of generalizability. Therefore, we want to extend the current study to a different corpus of open source project. In particular we are interested in knowing how accurate the technique is in the context of open source projects where more than one company is involved.

ACKNOWLEDGMENTS

The authors have been supported by Academy of Finland grant 298020.

REFERENCES

[1] R. A. Ghosh, R. Glott, B. Krieger, and G. Robles, "Free/libre and open source software: Survey and study (floss)," International Institute of Infonomics, University of Maastricht, Tech. Rep., 2002.

[2] K. Lakhani and R. Wolf, *Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects.* Cambridge: MIT Press, 2005.

[3] L. Torvalds and D. Diamond, *Just for Fun: The Story of an Accidental Revolutionary.* HarperInformation, 2001.

[4] R. T. G. Madey, V. Freeh, "The open source software development phenomenon: An analysis based on social network theory," in *Americas Conf. on Information Systems*, 2002, pp. 1806–1813.

[5] S. Krishnamurthy, S. Ou, and A. K. Tripathi, "Acceptance of monetary rewards in open source software development," *Research Policy*, vol. 43, no. 4, pp. 632 – 644, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0048733313001868

[6] J. Choi, B. Ferwerda, J. Hahn, J. Kim, and J. Y. Moon, "Impact of social features implemented in open collaboration platforms on volunteer self-organization: Case study of open source software development," in *Proceedings of the 9th International Symposium on Open Collaboration*, ser. WikiSym '13. New York, NY, USA: ACM, 2013, pp. 25:1–25:2. [Online]. Available: http://doi.acm.org/10.1145/2491055.2491081

[7] C.-G. Wu, J. H. Gerlach, and C. E. Young, "Examining the determinants of effort among open source software volunteer developers," *International Journal of Information and Decision Sciences*, vol. 5, no. 2, pp. 117–139, 2013. [Online]. Available: https://ideas.repec.org/a/ids/ijidsc/v5y2013i2p117-139.html

[8] M. AlMarzouq, V. Grover, and J. B. Thatcher, "Taxing the development structure of open source communities," *Decis. Support Syst.*, vol. 80, no. C, pp. 27–41, Dec. 2015. [Online]. Available: http://dx.doi.org/10.1016/j.dss.2015.09.004

[9] S. Athey and G. Ellison, "Dynamics of open source movements," *Journal of Economics & Management Strategy*, vol. 23, no. 2, pp. 294–316, 2014. [Online]. Available: http://dx.doi.org/10.1111/jems.12053

[10] C. Jergensen, A. Sarma, and P. Wagstrom, "The onion patch: migration in open source ecosystems," in *Joint European Soft. Eng. Conf. and ACM SIGSOFT Int. Symp. on Foundations of Software Engineering.* New York, NY, USA: ACM, 2011, pp. 70–80.

[11] M. W. Godfrey and Q. Tu, "Evolution in open source software: A case study," in *Int'l Conf. Software Maintenance.* IEEE Computer Society, 2000, pp. 131–142.

[12] I. Herraiz, G. Robles, J. J. Amor, T. Romera, and J. M. González Barahona, "The processes of joining in global distributed software projects," in *Proceedings of the 2006 International Workshop on Global Software Development for the Practitioner*, ser. GSD '06. New York, NY, USA: ACM, 2006, pp. 27–33. [Online]. Available: http://doi.acm.org/10.1145/1138506.1138513

[13] P. A. Thoits and L. N. Hewitt, "Volunteer work and well-being," *Journal of Health and Social Behavior*, vol. 42, no. 2, pp. 115–131, 2001. [Online]. Available: http://www.jstor.org/stable/3090173

[14] L. Prouteau and F.-C. Wolff, "On the relational motive for volunteer work," *Journal of Economic Psychology*, vol. 29, no. 3, pp. 314 – 335, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167487007000669

[15] M. Claes, M. Mäntylä, M. Kuutila, and B. Adams, "Abnormal working hours: effect of rapid releases and implications to work content," in *Proceedings of the 14th International Conference on Mining Software Repositories.* IEEE Press, 2017, pp. 243–247.

[16] ——, "Do programmers work at night or during the weekend?" in *Int'l Conf. Software Engineering*, 2018.

[17] D. Riehle, P. Riemer, C. Kolassa, and M. Schmidt, "Paid vs. volunteer work in open source," in *2014 47th Hawaii International Conference on System Sciences*, Jan 2014, pp. 3286–3295.

[18] J. Teixeira, G. Robles, and J. M. González-Barahona, "Lessons learned from applying social network analysis on an industrial free/libre/open source software ecosystem," *Journal of Internet Services and Applications*, vol. 6, no. 1, p. 14, Jul 2015. [Online]. Available: https://doi.org/10.1186/s13174-015-0028-2

[19] T. Therneau, B. Atkinson, B. Ripley, and M. B. Ripley, "Package 'rpart'," *Available online: cran. ma. ic. ac. uk/web/packages/rpart/rpart. pdf (accessed on 20 April 2016)*, 2015.

[20] M. Kuhn *et al.*, "Caret package," *Journal of statistical software*, vol. 28, no. 5, pp. 1–26, 2008.