Sentiment Analysis of Travis CI Builds

Rodrigo Souza
Department of Computer Science
Federal University of Bahia (UFBA)
Salvador, Brazil
Email: rodrigorgs@ufba.br

Bruno Silva
University of Salvador (UNIFACS)
Salvador, Brazil
Email: bruno.carreiro@pro.unifacs.br

***Abstract*—Human factors such as sentiments, emotions, mood, and stress along with their potential effect on software development are of paramount importance in software engineering, as we still strongly rely on human-to-human interaction for performing software development activities and driving results. With the advance of sentiment analysis tools, software engineering researchers have investigated the interplay between developers' sentiment and software engineering tasks such as issue fixing times. However, there is a lack of studies analyzing whether there is a relation between developers' sentiment and builds performed by continuous integration servers. Build breakage is not desired as it represents a signal that something went wrong in the software development activity and that extra work or rework should be done. In this paper, we report an empirical assessment over Travis CI builds and the corresponding commits in order to understand a potential association between developers' sentiment and build breakage. We found evidence that negative sentiment both affects and is affected by the result of the build process, although the influence seems to be small. Also, we found that developers tend to be more positive when writing about the CI server in commit messages.**

***Keywords*-Continuous Integration; Sentiment Analysis; Human factors on Software Engineering.**

I. Introduction

Software development is primarily conducted by humans. Even with the advance of software engineering technologies to support the automation of several tasks, we still strongly rely on human-to-human interaction for performing software development activities and driving results. Human factors such as mood and emotion are often neglected in the software engineering field whereas they are strongly associated to problem-solving [1]. For that matter, researchers have recently investigated human aspects in software engineering, including developers' sentiment expressed through software artifacts such as issue comments and commit logs [2]–[6]. In software engineering, as in other human-oriented activities, the exploration of emotional awareness may improve the way engineers assign tasks and collaborate in order to better coordinate software development activities [7].

Researchers have found an association between developers' sentiment and the number of files involved in software changes [6], issue fixing times [5], programming language and team geographical distribution [2]. They also found that happy developers are indeed better problem solvers in terms of their analytical abilities [1]. However, there is a lack of investigation on whether different levels of developers' sentiment affect the status of builds in continuous integration (CI) servers.

CI servers have achieved widespread use as they play an essential role in automating the integration process by building the system every time source code changes are pushed to the repository, thus reliably anticipating integration issues. Broken builds — those builds in which the source code does not compile or fails to pass all automated tests — are not desired and they represent a signal that something went wrong in the software development activity and that extra work or rework should be done. Therefore, understanding whether and how developers' sentiment is associated with project builds is important to discover new ways of improving software development. Based on this assumption, we pursued the following research questions:

RQ1: Are commits with negative sentiment more likely to result in a broken build? Our intuition is that negative sentiment, as expressed by developers in commit messages, may lead developers to make mistakes that ultimately result in a broken build.

RQ2: Are commits following a broken build more likely to be negative? We hypothesize that broken builds arouse negative feelings such as frustration, anger, and hopelessness, which may be expressed in commit messages.

RQ3: What is the overall sentiment in commit messages mentioning the CI server? We expect messages mentioning the CI server to appear after broken builds, expressing negative sentiment.

In order to address our research questions, we conducted an empirical assessment involving 1,262 projects hosted on GitHub that use Travis CI, a popular CI server for open source projects. To this end, we used the TravisTorrent data set [8], containing information about more than 609k builds of those projects.

The rest of the paper is organized as follows: Section II explains our study settings; Section III presents results and corresponding discussion; and Section IV summarizes findings and future work.

II. Methods

We analyzed 1,262 projects in Java and Ruby that use Travis CI, whose build metadata was made available in the TravisTorrent data set [8][1]. For those projects, we also cloned their GitHub repository and extracted the messages associated

[1]We used the data set released on February 2, 2017.

TABLE I
SUMMARY OF THE DATA SET

Number of projects	1,262
Number of builds	609,467
Number of commits	1,016,017
% of broken builds	26.5%
% of builds with negative sentiment	11.7%
% of builds with positive sentiment	6.6%

with all commits in their history. Information about the data is presented in Table I.

A. Continuous integration

Every time new commits are pushed to a project's GitHub repository, Travis CI starts the build process, which can be split into one or more *build jobs* – either to parallelize the execution of automated tests or to test multiple platforms. Each build job can be either successful (if its status is *passed*) or broken (if its status is either *failed*, *errored*, or *canceled*). We consider a build to be successful only if all its build jobs are successful.

B. Sentiment analysis

To analyze the sentiment in commit messages, we used the Java version of the SentiStrength tool [9], also applied in previous software engineering studies [2], [6], [7], [10]. SentiStrength assigns to each word in a sentence a sentiment score, which is a pair composed of a positive score from 1 (not positive) to 5 (extremely positive), and a negative score from -1 (not negative) to -5 (extremely negative)[2], according to a prepopulated database of positive and negative words. Additional rules can modify the scores for a word, such as booster words (e.g. *very*), repeated letters in a word (e.g. *haaaaappy*), and exclamation marks.

The sentiment score of a text is then computed as the maximum of the positive scores and the minimum of the negative scores of the words in the text. For instance, in the sentence "I love tests, but dislike the awful API", three words are found in the sentiment database, together with their scores: *love* [3], *dislike* [-3], and *awful* [-4]. As a result, the sentence is both positive and negative, and its sentiment score is 3,-4.

As each build can be associated with multiple commits, we computed the sentiment score of each commit and determined the sentiment score of the build to be the maximum of the positive scores and the minimum of the negative scores.

C. Tailoring the word database for software engineering

By manually inspecting the sentiment scores computed for a sample of all commit messages, we found that some words classified as negative are in fact technical terms used in software engineering, such as *failure*, *bug*, and *violation*. As a result, many neutral commits were being classified as highly negative, biasing the results. To mitigate the problem, we removed those words from SentiStrength's database, together with other words identified by a previous study [7]. After further inspection, we also removed the words *broken*, *coerce*, *against*, and *mock* (they appear, for instance, in neutral sentences such as "test against mocked command output").

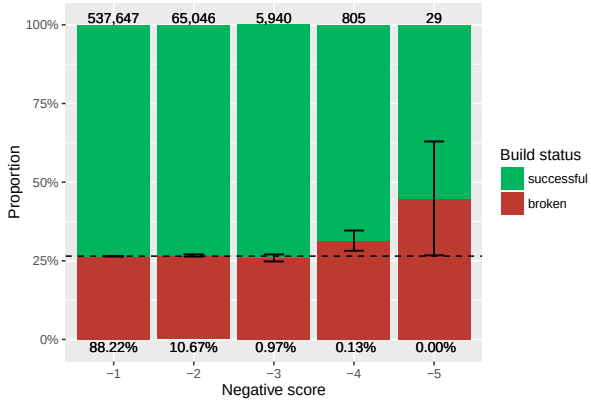

Fig. 1. Proportion of broken builds for each negative score. The error bars show the confidence interval for the proportion at the 95% confidence level. The dashed line represents the proportion of broken builds overall. The figures above the bars are the number of builds for each negative score; below the bars, the corresponding proportion.

D. Data analysis

To answer the research questions, we computed contingency tables by counting the number of broken and successful builds, the number of builds associated with each sentiment score, and the number of builds with commit mentioning Travis CI.

For RQ1, we used the chi-squared test of independence to assess the association between negative/positive messages and build status. To measure effect size, we used Cramer's V, interpreting $V < 0.3$ as small, $V < 0.5$ as medium, and $V > 0.5$ as large effect size [11].

For RQ2 and RQ3, the positive and negative scores are ordinal, dependent variables. For this reason, we assessed statistical significance using the Mann-Whitney U test, and measured the effect size using Cliff's delta, interpreting $|delta| < 0.33$ as small, $|delta| < 0.474$ as medium, and $|delta| > 0.474$ as large [12].

The data analysis scripts are available online[3].

III. RESULTS AND DISCUSSION

A. RQ1: Are commits with negative sentiments more likely to result in a broken build?

Figure 1 shows the proportion of broken and successful builds among builds with negative scores of -1 down to -5. Although highly negative messages (scores -4 and -5) are rare, they are more likely to result in a broken build.

The chi-squared test shows that the difference in proportions is statistically significant, with $p = 0.001$, although the effect size is small, with Cramer's $V = 0.005$. The analysis was replicated for positive commit messages but failed to show a statistically significant difference.

Table II shows a sample of negative messages for commits that resulted in broken builds. Numbers inside square brackets are sentiment scores for the preceding words. Commit (a)

[2]SentiStrengh does not consider zero as a sentiment score.

[3]https://gitlab.com/rodrigorgs/msr17-challenge

TABLE II
SAMPLE OF NEGATIVE COMMITS OF BROKEN BUILDS

	Score	Annotated commit message (snippet)
(a)	1,-4	Use different exclusion[-2] filter, jruby is sad[-4]
(b)	2,-4	Make the Bundler warning less scary[-4] and more friendly[2]
(c)	1,-4	Remove terrible[-4] dependencies spec
(d)	1,-4	no tests make me a sad[-4] boy, but i must run & will backfill this one tomorrow
(e)	1,-4	This fixes a really nasty[-3][-1 booster word] bug

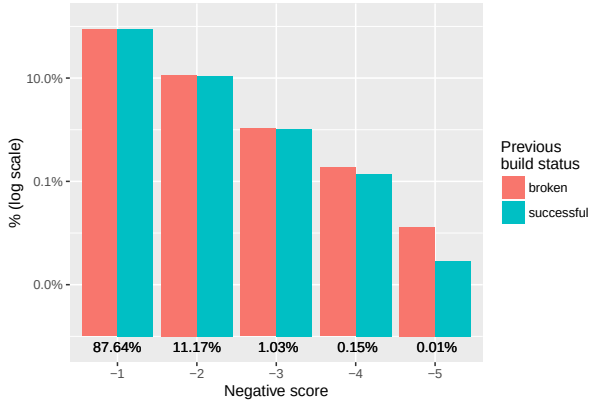

Fig. 2. Proportion of builds (log scale) associated with each negative score, split by whether the previous build was broken or successful. The numbers below the bars are the proportion of previous builds associated with each negative score.

shows a case where SentiStrength incorrectly classified a technical term ("exclusion") as negative. The remaining commit messages express negative sentiments such as fear (b) and sadness (d), as well as subjective opinions about bugs and specifications, as shown in (c) and (e).

> Commits with negative sentiment are slightly more likely to result in broken builds.

B. RQ2: Are commits following a build breakage more likely to be negative?

Figure 2 shows the proportion of builds associated with each negative score for builds following either a broken or a successful build. We ignored builds with more than one predecessor build, which can occur because of merge commits. The proportion of builds with more extreme negative scores is higher after broken builds ($p < 0.001$ for Mann-Whitney U test), although the effect size is small (Cliff's $delta = -0.014$).

The same analysis was performed for positive sentiment, as shown in Figure 3. In this case, the proportion of builds with higher positive scores is lower after broken builds, except for the positive score of 5 ($p < 0.001$ for Mann-Whitney U test), although the effect size is small (Cliff's $delta = -0.009$).

Table III shows a sample of negative messages for commits included in the next build after a broken build. Commit (a) refers to a release-related hack; commit (b) is a workaround

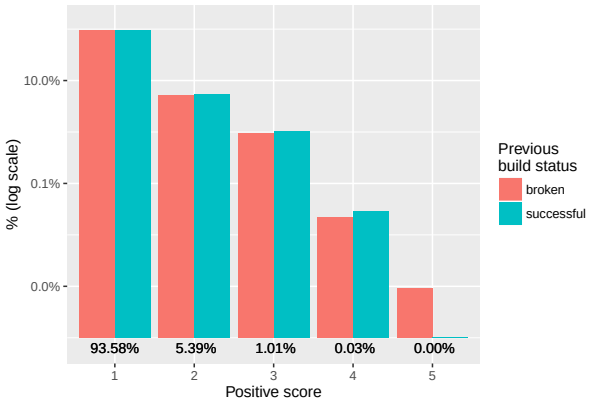

Fig. 3. Proportion of builds (log scale) associated with each positive score, split by whether the previous build was broken or successful.

TABLE III
SAMPLE OF NEGATIVE COMMITS AFTER BROKEN BUILDS

	Score	Annotated commit message (snippet)
(a)	1,-5	Horrible[-4], horrible[-4][-1 multiple negative words] hack[-2] to re-bundle on the 1.9 build
(b)	1,-3	Ridiculously[-3] long cucumber timeout so we're more resiliant to travis traffic
(c)	1,-3	Avoid[-2] ugly[-3] warning
(d)	1,-3	Ugh[-3], simple fix to quotes in Gemfile
(e)	1,-5	Definitely hating[-4][-1 booster word] #320

to cope with heavy network traffic in Travis CI's infrastructure; commit (c) seems unrelated to Travis CI, since the developer had been ignoring the previous ten builds; commits (d) and (e) are due to a previous build with failing tests; specifically for commit (e), multiple builds failed before the developer managed to fix bug #320, causing him to hate the bug.

> Commits following a build breakage tend to be more negative and less positive, although the effect size is small.

C. RQ3: What is the overall sentiment in commit messages mentioning Travis CI?

Table IV shows the distribution of positive and negative scores for both commits that mention Travis CI and commits that do not mention it. The Mann Whitney U test show a statistically significant difference in both cases: commits mentioning Travis CI tend to be more positive ($p = 0.004$) and less negative ($p = 0.005$). Both effect sizes are small (Cliff's $delta = -0.018$ and 0.010, respectively).

This result seems to contradict the intuition that developers talk about Travis CI when they experience the negative emotions of frustration and anger because of broken builds. This result is more surprising if we consider that developers were 3.5x more likely to mention Travis CI in commit messages when the last build was broken.

Table V shows a sample of positive commit messages mentioning Travis CI. In commit (a), "MOAR travis FUN",

TABLE IV
SENTIMENT SCORES VS. MENTIONING TRAVIS CI

Mentions Travis?	1	2	3	4	5
No	93.38%	5.56%	1.03%	0.03%	<0.01%
Yes	92.87%	5.91%	1.20%	0.02%	0.00%
Mentions Travis?	-1	-2	-3	-4	-5
No	88.20%	10.70%	0.97%	0.13%	<0.01%
Yes	88.88%	9.87%	1.06%	0.17%	0.01%

TABLE V
SAMPLE OF POSITIVE COMMITS MENTIONING TRAVIS CI

	Score	**Annotated commit message (snippet)**
(a)	3,-1	MOAR travis FUN[2]![+0.6 punctuation emphasis]
(b)	3,-1	MY FIRST COMMIT <3 [1 emoticon] (it would be so awesome[3] if this would make travis green) :P
(c)	3,-1	travis is kinda down, lets hope[3] nothing breaks while I sleep
(d)	3,-1	Make specs even slower, hopefully Travis will enjoy[3] this.
(e)	2,-1	Final slash may, or may not, be here (thanks[2] Travis)

the capitalization of words and the use of "moar" (internet slang for "more") are evidence of sarcasm. Commits (b) and (c) express hope that the next build will be successful. Finally, commits (d) and (e) show a personification of Travis CI; (d) signals the need to please Travis CI, and (e) expresses gratitude because Travis CI helped the developer discover a corner case in a test case.

> Commits mentioning Travis CI tend to be more positive and less negative.

D. Discussion

The results suggest that negative sentiment impairs the success of a build, which is consistent with the hypothesis that negative sentiment can affect developers' analytical abilities [1]. We also found that broken builds are more likely to be followed by negative sentiment, creating a feedback loop. Both results, however, had a small effect size, which suggests that sentiment plays a minor role in the success of a build.

The analysis of commits mentioning Travis CI showed a trend towards positive sentiment, with examples of thankfulness and hope. We also observed that, since SentiStrength is not able to identify sarcasm and irony in the messages, some commits classified as positive may actually be an ironic expression of negative sentiment.

In fact, the precision of SentiStrength is the main threat to the validity of the results. In preliminary analyses, with the original SentiStrength word database, the observed effect size was much larger. Although we removed from the database a list of technical words that carry no negative connotation in the software domain, the list was not comprehensive; therefore, the actual effect sizes may be even smaller than reported.

IV. CONCLUSION

To our knowledge, this is the first study that analyzed the association between developers' sentiment and build breakage in a continuous integration (CI) process. We found evidence that negative sentiment both affects and is affected by the result of the build process, although the influence seems to be small. We also found that developers tend to be more positive when writing about the CI server in commit messages.

As future work, we suggest an additional effort to improve the calibration of sentiment analysis tools, taking domain-specific words into consideration. Another option is to manually classify a subset of all commit messages and then apply supervised sentiment analysis. Furthermore, to better understand developers' sentiment towards the CI server, we plan to classify commits according to a more detailed set of sentiments, such as hope, frustration, gratitude, stress, among others. Although no automated tool currently identifies all those sentiments, tools such as TensiStrength[4] may be used to analyze stress and relaxation. Finally, we intend to analyze how sentiment differs among commits with distinct purposes, such as bug fixing, feature implementation, and refactoring.

REFERENCES

[1] D. Graziotin, X. Wang, and P. Abrahamsson, "Happy software developers solve problems better: psychological measurements in empirical software engineering," *PeerJ*, vol. 2, p. e289, Mar. 2014.

[2] E. Guzman, D. Azcar, and Y. Li, "Sentiment analysis of commit comments in GitHub: an empirical study," in *Proc. of the 11th Working Conf. on Mining Software Repositories*. ACM, 2014, pp. 352–355.

[3] A. Murgia, P. Tourani, B. Adams, and M. Ortu, "Do developers feel emotions? An exploratory analysis of emotions in software artifacts," in *Proceedings of the 11th Working Conference on Mining Software Repositories*, ser. MSR 2014. ACM, 2014, pp. 262–271.

[4] N. Novielli, F. Calefato, and F. Lanubile, "The challenges of sentiment detection in the social programmer ecosystem," in *Proceedings of the 7th International Workshop on Social Software Engineering*, ser. SSE 2015. New York, NY, USA: ACM, 2015, pp. 33–40.

[5] M. Ortu, B. Adams, G. Destefanis, P. Tourani, M. Marchesi, and R. Tonelli, "Are bullies more productive?: Empirical study of affectiveness vs. issue fixing time," in *Proc. 12th Working Conference on Mining Software Repositories*, ser. MSR '15. IEEE Press, 2015, pp. 303–313.

[6] V. Sinha, A. Lazar, and B. Sharif, "Analyzing developer sentiment in commit logs," in *Proceedings of the 13th International Conference on Mining Software Repositories*, ser. MSR '16. New York, NY, USA: ACM, 2016, pp. 520–523.

[7] M. R. Islam and M. F. Zibran, "Towards understanding and exploiting developers' emotional variations in software engineering," *2016 IEEE 14th International Conference on Software Engineering Research, Management and Applications (SERA)*, pp. 185–192, 2016.

[8] M. Beller, G. Gousios, and A. Zaidman, "TravisTorrent: Synthesizing Travis CI and GitHub for full-stack research on continuous integration," in *Proc. 14th Working Conf. on Mining Software Repositories*, 2017.

[9] M. Thelwall, *The Heart and Soul of the Web? Sentiment Strength Detection in the Social Web with SentiStrength*. Cham: Springer International Publishing, 2017, pp. 119–134.

[10] R. Jongeling, S. Datta, and A. Serebrenik, "Choosing your weapons: On sentiment analysis tools for software engineering research," in *ICSME*. IEEE, 2015, pp. 531–535.

[11] J. Cohen, *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, 1988.

[12] J. Romano, J. Kromrey, J. Coraggio, and J. Skowronek, "Appropriate statistics for ordinal level data: Should we really be using t-test and Cohen's d for evaluating group differences on the NSSE and other surveys?" in *annual meeting of the Florida Association of Institutional Research*, 2006, pp. 1–3.

[4] http://sentistrength.wlv.ac.uk/TensiStrength.html