Received May 13, 2020, accepted June 10, 2020, date of publication June 15, 2020, date of current version June 25, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3002663

How Do You Propose Your Code Changes? Empirical Analysis of Affect Metrics of Pull Requests on GitHub

MARCO ORTU[1], GIUSEPPE DESTEFANIS[2], DANIEL GRAZIOTIN[3], MICHELE MARCHESI[4], AND ROBERTO TONELLI[4]

[1]Dipartimento di Ingegneria Elettrica ed Elettronica, University of Cagliari, 09124 Cagliari, Italy
[2]Department of Computer Science, Brunel University London, Uxbridge UB8 3PH, U.K.
[3]Institute of Software Technology, University of Stuttgart, 70174 Stuttgart, Germany
[4]Dipartimento di Matematica e Informatica, University of Cagliari, 09124 Cagliari, Italy

Corresponding author: Giuseppe Destefanis (giuseppe.destefanis@brunel.ac.uk)

ABSTRACT Software engineering methodologies rely on version control systems such as git to store source code artifacts and manage changes to the codebase. Pull requests include chunks of source code, history of changes, log messages around a proposed change of the mainstream codebase, and much discussion on whether to integrate such changes or not. A better understanding of what contributes to a pull request fate and latency will allow us to build predictive models of what is going to happen and when. Several factors can influence the acceptance of pull requests, many of which are related to the individual aspects of software developers. In this study, we aim to understand how the affect (e.g., sentiment, discrete emotions, and valence-arousal-dominance dimensions) expressed in the discussion of pull request issues influence the acceptance of pull requests. We conducted a mining study of large git software repositories and analyzed more than 150,000 issues with more than 1,000,000 comments in them. We built a model to understand whether the affect and the politeness have an impact on the chance of issues and pull requests to be merged—i.e., the code which fixes the issue is integrated in the codebase. We built two logistic classifiers, one without affect metrics and one with them. By comparing the two classifiers, we show that the affect metrics improve the prediction performance. Our results show that valence (expressed in comments received and posted by a reporter) and joy expressed in the comments written by a reporter are linked to a higher likelihood of issues to be merged. On the contrary, sadness, anger, and arousal expressed in the comments written by a reporter, and anger, arousal, and dominance expressed in the comments received by a reporter, are linked to a lower likelihood of a pull request to be merged.

INDEX TERMS Software engineering, behavioral software engineering, human aspects, sentiment analysis, software quality, version control systems.

I. INTRODUCTION

Several modern software engineering methodologies and techniques are based on version control systems, which allow storage of source code artifacts, often referred to as codebase, as well as management of changes to said artifacts [41]. One of these systems is git, a decentralized version control system that has become the de-facto standard version control system in use by open source companies as well as software companies [8], a success likely fostered by the leading company GitHub. GitHub hosts more than 44 million git repositories of source code used by more than a million individuals all over the world.[1] Even though git is decentralized, a classic mode of usage foresees a central repository from which contributors sync their changes. Such a central repository is often hosted on a remote server, and GitHub provides this central role.

Pull requests are part of these code synchronization activities. A pull request is the ability to propose changes to a

The associate editor coordinating the review of this manuscript and approving it for publication was Liang-Bi Chen.

[1]See https://octoverse.GitHub.com

remote codebase, which is often the one considered to be the central repository. A pull request is based on those changes, or delta, between a locally cloned codebase and the target central repository. As its name indicates, a pull request is not an immediate change in the codebase, but a request for it. Pull requests are an important aspect of collaborative and open source software development; they are vastly popular on GitHub and form the so called pull-based software development model [15].

Pull requests are social in nature. Requests for a change are submitted to the project reviewers and integrators who open a discussion and eventually decide whether the changes are merged into main branch.

Ultimately, a pull request is either accepted, rejected, or staled. With the first two cases, a decision to merge the pull request is ultimately taken by people in the roles of core developers (those who directly perform changes in the codebase) or integrators (those who are responsible for integrating the changes into the codebase) [16]. The decision-making process and the communication behind the pull request model is all but simple. Much discussion arises and contributes to the fate of the pull request and the time it takes to execute its fate, also known as pull request latency [47]. Understanding what contributes to the fate of a pull request is important to evaluate precedents of code that is important to fulfill desired functionality of a software product or enhance the existing codebase in terms of quality. Discovering contributing factors that prolong or shorten the latency of a pull request helps the body of knowledge to understand productivity of a software development endeavor, as we obtained desired changes earlier under certain conditions with respect to others. All in all, a better advancement of what contributes to a pull request fate and latency allows us to build predictive models of what is going to happen and when. These triggers can invoke support tools to accelerate the productivity of developers and the quality of the codebase (e.g., a bot for static analysis could be invoked when the time is appropriate [46]).

Much research attention has started to focus on how social factors play a role on pull requests fate and how long it takes to get there [47]. Examples of social factors are the social distance between those in a discussion and prior interactions among individuals [43], [43]. A special instance of social factors are those at the individual level. Feedback of individuals on someone's creation elicits discussions colored by sentiment, emotions, and happiness. Research has already established that pull request discussions generate strong variations of emotions among all involved individuals [40], [42]. Emotions, moods, sentiment, and happiness (in this paper referred to as *affect* overall) color the everyday lives of developers and software engineering research is certainly not neglecting them any longer.

Related research has shown how positive and negative sentiment and affect drive the productivity of software developers [20], their problem solving skills [19], and many other aspects including communication, knowledge sharing, morale and collaboration, attractiveness of of open source projects, and burnout [17], [20], [24], [29], [34]. Negative emotions and unhappiness, on the other hand, were found to be linked to reduced productivity [17] and nefarious consequences including, but not limited to, low motivation, task avoidance, work withdrawal, and, more related to the present study, a tendency to discard code [18]. We turn our attention to how emotions expressed in pull requests affect the likelihood of an issue to be merged or not, as related work seems to suggest it might be the case.

Work in psychology and education has paid particular attention to politeness, that is the act of expressing ourselves as ''marked by an appearance of consideration, tact, deference, or courtesy'' [26], and how it plays a role in motivating individuals to carry cognitive tasks and reduce resisting to requests. Intuitively, we are more prone to respond positively to polite requests than to impolite requests; a polite message is perceived as less threatening, and there is evidence that politeness alleviates friction in the process of resistance to requests [22], [49]. Furthermore, politeness has been linked to higher proficiency in learning outcomes [22] which suggests that higher motivation arises to carry cognitive processing activities. Initial research in software engineering has explored the role of politeness in issue requests. Core team members tend to express themselves politely when interacting with new submitters [42], [43], and polite comments in Jira[2] repositories provide a shorter issue fixing time [10], [33].

Based on the above illustrated previous work, we aim to investigate whether politeness and affect are associated with merged issues, and we set the following three research questions.

RQ1: Are merged issues influenced by affect and politeness expressed in their related discussions?

RQ2: To which extent affect and politeness influence merged issues?

RQ3: Are affect metrics a good predictor to the likelihood of pull requests to be merged?

The overall objective of the present study is to provide a better understanding of how politeness and sentiment expressed in pull request discussions contributes to the probability for a pull request to be merged into the codebase. For answering RQ1 and RQ2, we conducted a mining study of large software repositories and analyzed more than 150,000 issues with more than 1,000,000 comments in them. We built a model to understand whether the affect and politeness have an impact on the chance of issues and pull requests to be merged, meaning that the code which fixes the issue is integrated in the code base. For answering RQ3, we built two logistic classifiers, one without affect metrics and one with them, and by comparing the two classifiers we show that the affect metrics improve the prediction performance.

A deeper understanding of comments around decisions on source code contributions, in our case pull requests, will lead

[2]Jira, https://www.atlassian.com/software/jira is a software system to create, track, and manage issues and feature requests for software products.

to the a creation of policies and tools to better handle open source software as well as proprietary software using a pull request approach. For example, if sentiment and politeness are good predictors of pull request acceptance (or rejection), we can automatize the invocation of tools for integration, conflict resolution, and notification systems.

Section II provides related work. Section III reports on the methodology, that is the experimental setup, dataset construction and cleanup, as well as measurement theory and application. Section IV provides the results organized following the three research questions. Section V discusses the results and relates them with other works, as well as the limitations of the study and implications of the results. Section VI concludes the paper with suggestions for future studies.

II. RELATED WORK

Tsay *et al.* [43] conducted a study of how developers evaluate and discuss pull requests on GitHub. The authors analyzed 423 comments in pull requests coming from 115 developers and later supplemented their qualitative findings with interviews with 47 GitHub users. While their study provided insights on discussion and evaluation of pull requests that are out of scope for the present paper, the results highlight how intense and emotion-laden are the discussion between contributors and core developers. In particular, the study highlighted strong instances of politeness in pull request discussions, especially in cases of core team members interacting with new contributors. How the politeness played a role in the fate of the pull request was not in scope of the contribution, but another study by Tsay *et al.* [42] concluded that there is evidence that developers use social information, in addition to technical information, for evaluating a pull request.

Yu *et al.* [48] investigated whether and how previous approaches used in bug triaging and code review can be adapted to recommending reviewers for pull requests, and how to improve the recommendation performance. Their results indicated that combining social factors such as common interests among developers, and technical factors e.g., developers expertise, is an efficient way to build recommender systems in social coding platforms (recommending reviewer, bug fixer or coding partner).

Ortu *et al.* [33] analyzed all JIRA comments contained in 14 open source projects. They found that in 10 out of 14 projects, the issue fixing time for polite issues was significantly faster than the fixing time for impolite issues. Similarly, Destefanis *et al.* [10] studied 22 open source software projects developed with the Agile board of the JIRA repository and found that the level of politeness in the communication process among developers has a positive effect on the time required to fix issues.

Santos *et al.* [40] analyzed mood variations on more than 268,000 comments in 78,000 GitHub pull requests. Among the results, they found that in about a third of cases a strong variation in mood occurs within an hour after receiving feedback on a pull request. Receiving a negative feedback on their first pull request causes 11% to 24% (according to project size) to never contribute again to the project. The authors conclude in wishing future research to investigate the role that politeness (and impoliteness) play in the success of open source projects and their productivity.

On the same wavelength is the work of 2002 by Erez and Isen [12] who experimented how motivation is influenced by positive affect.

Software engineering was early involved in this type of study. As Cockburn *et al.* wrote in 2001 [6], the human factor was identified to be the key in a development team. In particular, according to Capretz [5], software engineers' personalities cover the all sphere of human personality and discover that diversity is an advantage in development teams.

Rigby and Hassan [37] analyzed the big five personality traits of software developers in the Apache server mailing list. The authors used a psychometrically-based linguistic analysis tool and found that two developers responsible for the major Apache releases had similar personalities and their personalities were different from other developers.
Feldt *et al.* [13] focused on personality as a relevant psychometric factor. They presented results from an empirical study about correlations between personality and attitudes to software engineering processes and tools. Authors defined the personality dimensions and found that higher levels of ''conscientiousness'' correlated with attitudes towards work style, openness to changes and task preference.

Acuna *et al.* [1], performed empirical research examining the work climate within software development teams. The authors attempted to understand if team climate (defined as the shared perceptions of team work procedures and practices) bear any relation to software product quality. They found that high team vision preferences and high participative safety perceptions of the team were significantly related to better software.

Iyer *et al.* [21] presented an empirical study showing the effects of developers' personality traits on pull request acceptance of GitHub. Results showed that the likelihood of pull request acceptance is significantly influenced by personality traits of developers. The authors observed requesters who were high on Openness, Conscientiousness, and low on Extroversion had a higher likelihood of getting the pull request accepted. Similarly, a closer who were high on Openness, Conscientiousness, Extroversion, and Neuroticism accepted more pull requests. While our work focuses on pull request acceptance as well, we analyze the process from a different perspective, considering politeness and affect, and complementing the research conducted by Iyear *et al.*

III. METHOD

We designed our study to investigate the if and how affect and politeness expressed by developers during pull request discussions influence the likelihood of a pull request to be merged. To conduct our experiment, we first built a dataset of pull request issues from GHTorrent dataset [14] identifying about 66K GitHub contributors (both users and developers),

TABLE 1. Dataset statistics.

Project	Years	Language	# Issues	# Comments
Rails	13 years	Ruby	31731	138039
Homebrew	8 years	Ruby	25235	95342
Bootstrap	6 years	Less	43846	142345
Angular.js	5 years	JavaScript	11770	77824
Docker	4 years	Go	14998	198781
Kubernetes	4 years	Go	11793	189408
Rust	8 years	Rust	16994	174218
total	-	-	156367	1015957

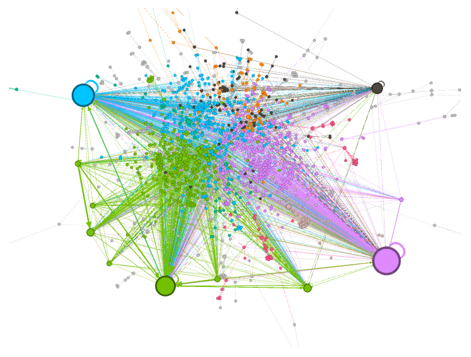
FIGURE 1. Example of Issue Collaboration Network (Angular.js).

156K issues and 1M comments from seven open source software as shown in Table 1. We provide the dataset as open data [32].

Contributors start a communication process when commenting pull requests. This process can be modelled using a direct graph network. Each contributor of the project can be represented by a node in the graph, and when contributor A comments a pull request created by contributor B, a link is generated from node A to node B. Such direct graph is called Issue Collaboration Network, and Figure 1 shows, as introduced by [29], [31], an example of such a network built using the data from Angular.js. The different colors in the graph represents communities of contributors (calculated using clustering algorithms), while the diameter of each node is proportional to the degree of the node. We used Gephi[3] (an interactive visualization and exploration tool) [2] to analyze and build the Issue Collaboration Network, and we ran the modularity algorithm, based on the algorithms developed by Blonde [3] and Lambiotte [23], to obtain the network communities [34].

Given a node A of the direct graph representing the communication process among contributors of a given project, it is possible to measure (I) the affect expressed by a contributors, e.g. the affect calculated from the comment written by contributor A, and represented in the graph by the out-links, and (II) the affect "received" from other contributors, e.g., the affect of the in-links to the node A, which represents the interactions among all the contributors commenting a pull request issued by the node A.

[3] http://gephi.GitHub.io/

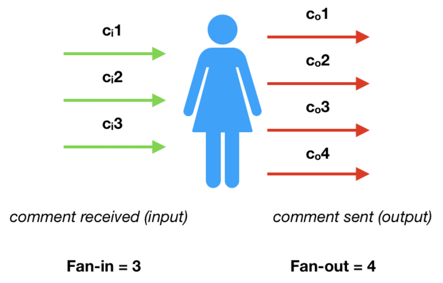

FIGURE 2. Contributor Fan-in and Fan-out.

To calculate the affect expressed (out-links) and received (in-links) by a contributor (node of the graph), we used the same approach followed by Destefanis *et al.* [9], [35], who analyzed issues and comments on GitHub projects and built collaboration networks dividing contributors into two categories: users and commenters. The authors identified as commenters those users who only post comments without posting any issues nor committing changes in the source code. The authors calculated and compared the affectiveness of the issues' comments written by users and commenters in terms of sentiment, politeness, and emotions, and provided empirical evidence that commenters are less polite, less positive and in general they express a lower level of emotions in their comments than users. The results confirmed that GitHub contributors consist of different groups which behave differently.

In Figure 2 we show how we model (I) the affect received by GitHub contributors on the comments of their own opened issues (Fan-in, or the green arrows in Figure 2) and (II) the affect expressed by contributors on comments in (non-owned) issue they commented on (Fan-out). We consider affect Fan-in and Fan-out as a representation of the communication behavior of a contributor.

To quantify the influence of affect on merged pull requests we built a logistic regression model using the affect metrics as independent variables. We included the following set of control variables:

- # Issues previously created by a reporter
- # Comments previously created by a reporter
- # Commits of the reporter
- Whether the reporter is a developer (has commits on the main branch) or not

Since a pull request can be merged or not in the main branch, we used a logistic regression for modelling this binary output, given the binary output of the model. Additionally, we have more features as input metrics, and the logistic regression model with a multivariate analysis is suitable for better explaining the variance of the inputs.

In the following subsections we report in more details the measurements used in our experiment.

A. MEASURING AFFECT

In line with recent work in psychology (e.g., [39]) we consider affect as a fundamental building block for emotions and moods. So, we use the term *affect* to refer to all emotions, moods, and sentiment.

In this work, we considered affect using both the discrete approach and the dimensional approach. The discrete approach of affect identifies emotional states that can be distinguished uniquely [36], and that possess high cross-cultural agreement when evaluated by people in literate and preliterate cultures [11].

The dimensional approach of affect groups emotional states into major dimensions that allow a clear distinction among them [38]. The Valence-Arousal-Dominance (VAD, also known as Pleasure-Arousal-Dominance, PAD) model is a one dimensional approach [25]. Valence, arousal, and dominance are affective dimensions that can be used to derive a person's interest (attraction), level of activation and perceived level of control for a particular stimulus, or a situation from textual communication.

1) MEASURING DISCRETE AFFECT

In order to measure discrete affect we used a machine learning classifier proposed by Ortu *et al.* [28] and extended by Murgia *et al.* [27]. In the extended work, they parsed Apache's Jira-based repository in July 2013, fetching all the issue reports since the 19th of October 2000, building a classifier model able to detect *love* (with 0.82 F1 measure), *joy measure* (with 0.7 F1 measure), *anger* (with 0.82 F1 measure) and *sadness* (with 0.84 F1 measure). They used a refining algorithm for training the machine learning classifiers. We used the same classifiers for detecting emotions in contributors' comments. We selected this tool for emotion detection since it has been specifically trained for the software engineering domain.

2) MEASURING VALENCE, AROUSAL AND DOMINANCE AFFECT DIMENSIONS

All measures of VAD are based on a list of words that have manually been analyzed and assigned a VAD score. Warriner *et al.*'s [45] leading lexicon contains $13,915$ English words with VAD scores for Valence, Arousal, and Dominance. To calculate the corresponding VAD scores for a piece of text (i.e., a list of words $\bar{w} = [w_1, w_2, \ldots, w_n]$), the Range of the words' individual VAD scores is computed by taking the two words with the Max and Min Valence, Arousal or Dominance. For the particular cases when Max has lower than average value or when Min has higher than average value we set the Max or Min to the average of all words of the lexicon ($\bar{W} = [W_1, W_2, \ldots, W_N]$, where N is 13,915).

B. MEASURING POLITENESS

To compute the politeness of the contributors in our dataset, we adopted the python library developed by Danescu *et al.* [7]. Given an input text, the tool calculates its overall politeness in terms of a binary output, i.e. *polite* or *impolite*.

The tool was trained and validated through machine learning on a gold standard of over $10,000$ manually labeled requests from Wikipedia and StackOverflow, a Q/A website for software developers. The gold standard was built to

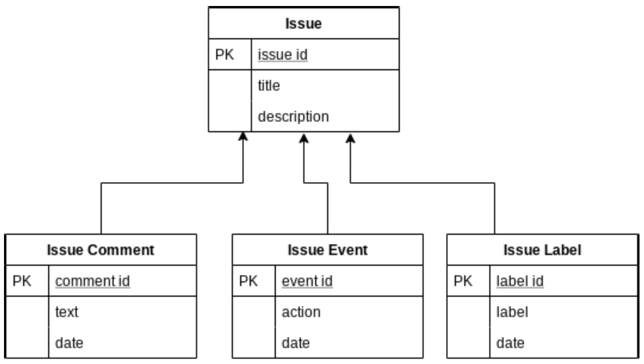

FIGURE 3. **GHTorrent relational schema.**

include comments written by authors from all over the world while the annotators were selected among U.S. residents, based on a linguistic background questionnaire. Since the tool has been trained on StackOverflow, it is suitable for the domain of Software Engineering (SE). We have applied the politeness tool on a domain very similar to StackOverflow (regarding context) since we analyzed GitHub commit messages describing and discussing how the work is done.

C. MEASURING ISSUE REPORTER EXPERIENCE

To measure the issue reporter's experience we considered the number of issue created, the number of comments, the number of commits and whether the contributor is a developer (they must have at least one commit in the project under analysis). We consider these variables as control variables: if a developer with high experience with a repository submits their code changes with a pull request, the request is more likely to be merged on the main branch than a request performed by a new user.

D. INFERRING MERGED ISSUES

Figure 3 shows the portion of GHTorrent relational schema used to infer whether an issue is merged in the main branch.

When a contributor creates a pull request, an issue ticket is automatically created. In this schema, Issues have Events, and each Event has an action. When this action is "merged" we assume that the related Issue has to be merged in the main branch of the repository. For "merged" issues we considered only comments posted *before* the merge event, since after the merge event we detected a bias effect toward positive affect, due to contributors thanking for "the good job". We found a total of 14,513 "merged" issues accounting for 9.28% of the total issues analyzed.

IV. RESULTS

A. RQ1: ARE MERGED ISSUES INFLUENCED BY AFFECT AND POLITENESS EXPRESSED IN THEIR RELATED DISCUSSIONS?

Motivation: In the last years, an increasing number of researchers have investigated the role played by affect during software development. Research has shown that affect, here expressed in text comments by contributors of open source software, is linked to morale and collaboration, attractiveness of open source projects, task and work engagement, productivity (in terms of issue fixing time), performance, and a tendency to not discard code. Research in psychology and education suggests that expressing requests in a polite way may reduce a resistance to respond to such requests. Initial research in software engineering has qualitatively observed this phenomenon. Here we investigated how affect and politeness influence the likelihood of an pull request to be "merged" in the main branch.

Approach: We used a GHtorrent dataset of GitHub activities, looking at all comments related to about 156K pull request issues and measuring discrete affect and VAD metrics of all the contributors involved. We built a logistic regression model using the experience of a contributor as control variables, as explained in Section III.

The output variable of our model is a binary variable representing the merged event as output.

Findings: **Pull request issues with higher level of *out_arousal*, *out_dominance* and *out_sadness* (highlighted in dark grey in Table 2), are less likely to be merged, while pull request issues with higher level of *reporter_in_valence*, *out_joy* and *out_valence* are more likely to be merged. (highlighted in light grey in Table 2)**

Table 2 shows the details about the logistic regression model for merged events.

The signs of the values in the column Estimate indicate the direction of the influence on the output variable. Negative values indicate a lower probability for the pull request to be merged, while positive values indicate a higher likelihood for the pull request to be merged. Control variables such as the number of issues created (*num_issues_created* in Table 2), number of commits (*num_commits* in Table 2), and *is_reporter_a_developer* are all significant metrics and are linked to a higher probability of the pull request being merged. The number of comments (*num_comments* in Table 2, values of Estimate and p-value highlighted in pink) posted by the reporter is a significant metric as well and linked to lower probability of the pull request being merged. However, the impact on the dependent variable is negligible, as shown in Table 3 (row 8).

Among the affect variables we can see that *out_anger*, *out_love*, *out_politeness*, *reporter_in_love* and *reporter_in _politeness* are not significant metrics for our model. The Fan-in affect metric of the reporter are all significant with the exception of *reporter_in_sadness*, *reporter_in_love* and *reporter_in_politeness*. Higher level of arousal, dominance and anger received by the issue's reporter are linked with lower probability for the pull request to be merged, while higher level of joy and dominance are linked to higher probability for the pull request to be merged. The same results hold for affect metrics related to the issue.

TABLE 2. **Logistic Model For Issue Merged Likelihood.**

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-4.1730	0.1174	-35.55	0.0000
out_arousal	-1.1658	0.1888	-6.18	0.0000
out_valence	2.2564	0.2065	10.93	0.0000
out_dominance	-1.4458	0.2421	-5.97	0.0000
out_anger	-0.1206	0.3052	-0.40	0.6928
out_sadness	-0.3768	0.1058	-3.56	0.0004
out_joy	0.6792	0.1315	5.17	0.0000
out_love	-0.0611	0.1076	-0.57	0.5703
out_politeness	-0.0592	0.0874	-0.68	0.4984
reporter_in_arousal	-1.9493	0.4508	-4.32	0.0000
reporter_in_valence	3.7649	0.4848	7.77	0.0000
reporter_in_dominance	-2.5361	0.5985	-4.24	0.0000
reporter_in_anger	-1.7416	0.6309	-2.76	0.0058
reporter_in_sadness	-0.0964	0.1838	-0.52	0.6000
reporter_in_joy	1.1331	0.2227	5.09	0.0000
reporter_in_love	0.2474	0.1880	1.32	0.1882
reporter_in_politeness	0.0954	0.1548	0.62	0.5379
is_reporter_a_developer	2.7010	0.0717	37.66	0.0000
num_commits	0.0003	0.0001	4.81	0.0000
num_issues_created	0.0044	0.0003	14.44	0.0000
num_comments	-0.0002	0.0000	-9.68	0.0000

B. RQ2: TO WHICH EXTENT AFFECT AND POLITENESS INFLUENCE MERGED ISSUES?

Motivation: For the first research question we found which affect metrics are significant for our model. Now we are interested in quantifying how significant they are.

Approach: We compare the odds of each metric. Odds close to 1 have no influence on the output variable. Odds greater than 1 show a positive influence on the output variable (the higher the value, the stronger the odds). Odds less than 1 have a negative influence on the output variable (the lower the value, the stronger the odds).

Findings: ***reporter_in_valence*, *is_reporter_a_developer* and *out_valence* have the highest positive impact while *out_sadness*, *out_arousal* and *out_dominance* have the highest negative impact on the likelihood of issues to be merged.**

Table 3 shows the impact of each model metric on the output variable. We report in light grey (the first four) those metrics that have a positive impact on the output: an increase of these metrics is related to an increase of the likelihood of a pull request to be merged. We report in dark grey (the last six) those metrics which have a negative impact on the output: an increase of these metrics is related to a lower likelihood of a pull request to be merged. The metrics *num_issues_created*, *num_commits* and *num_comments* have a negligible influence on the output variable.

We can see how the most impacting metric of our model is the *reporter_in_valence*, which has a positive impact on the output variable. Valence represents the level of pleasure (> 0) and displeasure (< 0) expressed, and the *in* suffix is referred to the level of pleasure received by contributors from the issues created by the reporter. The second most impacting metric is *is_reporter_a_developer*, and indicates whether the reporter is a developer or not. This was an expected results,

TABLE 3. Odds Of Issue Merged Model.

Metric	Odds
(Intercept)	0.014
reporter_in_valence	38.433
is_reporter_a_developer	17.639
out_valence	14.956
out_joy	2.482
num_issues_created	1
num_commits	1
num_comments	0.999
out_sadness	0.667
out_arousal	0.238
out_dominance	0.178
reporter_in_anger	0.176
reporter_in_arousal	0.152
reporter_in_dominance	0.085

because if the reporter is a developer with previous pull requests with a sufficient number of positive comments in the discussion, it is likely that a new pull request authored by the reporter will be merged. This fact can be related to the ''perceived experience'' of the developers in the community.

The other two variables which positively impact the pull request merge process are *out_valence* and *out_joy*, which both express the positive polarity of the comments. On the other hand, *reporter_in_anger*, *reporter_in_arousal* and *reporter_in_dominance* have the most negative impact on the output. These results match common sense: if in the comments of a pull request there is a a high level of anger, a high level of arousal (which represent calmness (< 0), and excitement (> 0)), and a high level of dominance (which represents being-controlled (< 0) and being-in-control (> 0)), the pull request has a higher probability of being rejected.

C. RQ3: ARE AFFECT METRICS A GOOD PREDICTOR TO THE LIKELIHOOD OF PULL REQUESTS TO BE MERGED?

Motivation: Affect metrics can explain the likelihood of pull requests to be merged, and we have presented their impact on the regression model we used. To corroborate our results, we investigate whether the affect metrics are able to improve the performance of the machine learning classifiers.

Approach: First, we used two different classifiers: logistic classifier (Table 4) and random forest model (Table 5) with the affect metrics, to study how they perform against the same models without the affect metrics. For each classifier we built an incremental model starting with the control variables only, and adding a set of features (VAD metrics, emotions) for each step.

Findings: **Affect metrics are able to improve precision, recall, F1 measure and AUC.**

We compared the performance of two classifiers: (I) the logistic regression classifier and (II) the random forest classifier. We incrementally added features, starting from a model without affect metrics. We then added VAD metrics, discrete emotion metrics and finally all other metrics considered in our study. By definition, the ZeroR model has perfect recall

TABLE 4. Logistic Regression Model Performance.

Classifier	Class	Precision	Recall	F1	AUC
Without affect metrics	Not-Merged	0.592	0.912	0.718	0.712
	Merged	0.840	0.424	0.564	
	Weighted Avg.	0.721	0.658	0.638	
With only Emotions affect metrics	Not-Merged	0.785	0.729	0.75	0.825
	Merged	0.747	0.8	0.773	
	Weighted Avg.	0.766	0.765	0.764	
With only VAD affect metrics	Not-Merged	0.765	0.720	0.742	0.850
	Merged	0.756	0.798	0.776	
	Weighted Avg.	0.761	0.760	0.760	
With all affect metrics	Not-Merged	0.759	0.721	0.740	0.862
	Merged	0.755	0.790	0.772	
	Weighted Avg.	0.757	0.757	0.757	

TABLE 5. Random Forest Classifier Performance.

Classifier	Class	Precision	Recall	F1	AUC
Without affect metrics	Not-Merged	0.842	0.849	0.845	0.825
	Merged	0.847	0.840	0.844	
	Weighted Avg.	0.845	0.844	0.844	
With only VAD affect metrics	Not-Merged	0.703	0.941	0.805	0.915
	Merged	0.911	0.602	0.725	
	Weighted Avg.	0.807	0.772	0.765	
With only Emotions affect metrics	Not-Merged	0.724	0.938	0.817	0.921
	Merged	0.912	0.642	0.753	
	Weighted Avg.	0.818	0.790	0.785	
With all affect metrics	Not-Merged	0.715	0.944	0.814	0.924
	Merged	0.918	0.624	0.743	
	Weighted Avg.	0.817	0.784	0.779	

for ''Merged'', but its precision suffers, and recall for the ''Not-Merged'' class is zero, which results in an average weighted precision and recall (across both classes) of 0.25 and 0.5 respectively. The ZeroR model represents the baseline comparison for the two classifiers. Our models obtained good performances: precision, recall, F1 measure and AUC are all improved when the affect metrics are added to the models. AUC is the area under the receiver operating characteristic curve (ROC). AUC can be interpreted as the probability that, when randomly selecting a positive (''Merged'') and a negative (''Not-Merged'') example, the model assigns a higher score to the positive example. For a random model, this probability would be 0.5, which is the AUC obtained for the ZeroR. The logistic model obtained a value of 0.862 and the random forest model 0.924, significantly higher than 0.5. We then compared the incremental models every time we added a set of features, with the previous model with the ANOVA analysis (using a Chi-squared Test) obtaining a p-value of *2.2e-16* *** for each comparison. All models are significantly different from the previous model (with fewer metrics) and, by adding the affect metrics, precision, recall, F1 measure and AUC improved. The results hold for both the logistic and the random forest models.

V. DISCUSSION

In our study we firstly analyzed if merged pull request issues were influenced by politeness and affect expressed in their discussions. Tsay *et al.* [43] analyzed how developers in open work environments evaluate and discuss pull requests, finding that developers raised issues around contributions over both the appropriateness of the problem that the submitter attempted to solve and the correctness of the implemented solution. The authors concluded by suggesting that social relationships seemed to also have an impact on the pull request acceptance process. We investigated politeness and affect expressed in comments related to pull requests building on previous works in which the same entities were studied for better understanding the development process in terms of productivity, e.g., Ortu *et al.* [29], Destefanis *et al.* [10]. In this study we considered the pull request merge process, which can be indirectly linked with the concept of productivity. Having a deep knowledge and understanding of pull requests acceptance, can be an element for controlling and increasing productivity. We evaluated affect and VAD metrics of all the contributors involved, and found that those pull request issues with higher level of arousal, dominance and sadness, were less likely to be merged, while pull request issues with higher level of valence and joy were more likely to be merged. Similar effects have been shown in StackOverflow [44] where, due to the gamification strategy, users who ask question in rude manner, among other things such as showing no prior research on the website, receive negative scores and tends to receive less and less answers over time. Metrics resulting from politeness (out_politeness and reporter_in_politeness) as well as love (out_love and reporter_in_love) were found to be non-significant in our analysis. The number of comments (num_comments) posted by the reporter resulted in being significant and linked with lower probability of the pull request being merged in the main branch. This fact is related to previous results [9], [35], where it has been shown that the number of comments posted by developers was lower than the number of comments posted by users (where users are defined as "project contributors without commits"). The studies also showed that issues posted by developers attracted more comments than those posted by users. Therefore, a high value of number of comments, can be related to a contributor which is not a developer, explains the link with lower probability of this metric with the pull request being merged.

We then continued the study by exploring the significance of the considered metrics and compared the predictive models for understanding if affect metrics are good predictors for the likelihood of pull requests to be merged in the main branch. Results show that *reporter_in_valence* and *is_reporter_a_developer* have both a positive impact on the output variables, indicating a positive contribute to the probability of the pull request being merged. Again, this shows the different (and positive) perception of "developers" among the contributors in an open source project, highlighting the fact that roles are diverse and differently perceived in the community as presented in [9], [35]. While *out_valence* and *out_joy* have both a positive impact in the pull request merge process, *reporter_in_anger*, *reporter_in_arousal* and *reporter_in_dominance* have the most negative impact on the output. These results strengthen and complement the findings of numerous studies in the area of human aspects in software engineering, showing that good manners, communication skills, and positive attitude are crucial and have an impact on the development process.

Having more insights on pull requests, e.g., probability of a specific one to be merged, can be crucial for controlling the development process and for better understanding how developers interact, lowering and preventing conflicts, and understanding who are the "more listened to" contributors in the case of an open source project and in a distributed development environment.

A deeper understanding of comments around pull requests, can lead to the creation of systems and protocols for managing both open source and proprietary development procedures based on pull request approaches. Shedding light on the good predictors of pull request acceptance or rejection, can lead to automatize the invocation of tools for integration, conflict resolution, and notification systems.

The results presented in this study can be also helpful when defining a new team of developers and when distributing roles. Understanding how pull requests are merged, and studying the behaviors of the authors can add more information to the profile of the contributors to the project and can be helpful from a managing point of view for optimising and better allocating resources.

A. THREATS TO VALIDITY

Threats to external validity correspond to the generalization of experimental results. In this study, we used several empirical approaches to evaluate the collaboration network of seven projects from GitHub repositories and computed the affect of more than 1 million comments related to the GitHub projects. We considered the various datasets as a representative sample of the open source world. Hence, replications on commercial and other open source projects are needed to confirm our findings.

Threats to internal validity concern confounding factors that can influence the obtained results. Based on empirical evidence, we assume a causal relationship between the emotional state of developers and what they write in their discussion [45].

Another threat is related to the reliability of the emotion analysis tool applied in the software engineering domain.

For the emotion detection we used a tool specifically trained using Jira comments [30] while for detecting politeness we used the tool developed by Danescu *et al.* [7] which was trained using StackOverflow questions and answers following the same approach of Calefato *et al.* [4] for sentiment detection in software engineering.

Threats to reliability correspond to the degree to which the same data would lead to the same results when repeated. This research empirically investigates the affect of developers and users during software development by means of collaboration networks. To the best of our knowledge, no previous studies exist to allow a direct comparison with our results.

VI. CONCLUSIONS

Software development is an activity organised around team-based environments. The implementation of team structures is not simple and does not necessarily result in success, it is not enough to simply *put people together* in teams and to presume that everybody knows or agrees on what to do. In such teams, each developer is working on the same source code but has different tasks. Someone might be updating the front-end, someone else the back-end, but every change introduced by a developer is going to affect all the others. All developers are working on the same code at the same time, and this is when git becomes vital. Without control versioning, each developer would overwrite other people's work. Git allows to make contributions without letting the developers interfer with each other's work.

It is however necessary to reconcile the changes introduced by developers in the same piece of code, and this can happen through a pull request. A pull request can be seen as an answer to the question "what has been changed in the code?", and instead of directly merging the changes in the main branch, the pull request gives the possibility of "getting feedback" on the change and improving the submitted code (e.g., finding mistakes which the developer who submitted the pull request is not aware of). Pull requests generate discussion between the submitter of the proposed changes to the codebase and those who review the request.

This discussion-part which might arise is indeed a social activity related to the pull request. In this context, people working together apply different personal assumptions and interpretations to their work tasks and therefore conflicts among developers are possible and can affect pull requests. Conflicts affect teams' productivity, and team leaders are certainly interested in knowing how to prevent, avoid, or, in the worst case, manage conflicts which might occur.

Understanding what contributes to the fate of a pull request is important to evaluate precedents of code that is important to fulfill desired functionality of a software product or enhance the existing codebase in terms of quality.

While many factors influence the acceptance of these changes, we focus our study on the affect expressed by contributors and the politeness of requests. Our purpose was to find whether the way the changes are proposed and discussed influence the final merge. Similarly, we investigated whether the way the code changes are proposed and discussed influence the likelihood of these changes to be merged.

The overall objective of the present study was to provide a better understanding of how politeness and sentiment expressed in pull request discussions contributes to the probability for a pull request to be merged into the codebase.

First, we conducted a mining study of large software repositories and analyzed more than 150,000 issues with more than 1,000,000 comments in them. We built a model to understand whether the affect and politeness have an impact on the chance of issues and pull requests to be merged, meaning that the code which fixes the issue is integrated in the code base. We established how this likelihood is influenced by the affect expressed during development. We analyzed pull request discussion considering the affect expressed by participants as well as the affect received by the pull request reporter in previous discussions, as explained in Section III. We found that the *reporter_in_valence* is the most positively impacting variable, meaning that the pull request reporter who received high level of valence-related to high value of pleasure expressed in text-in his/her previous pull requests is likely to have his/her pull request merged. On the other hand, we found that the most negative impacting variables are *reporter_in_dominance*, *reporter_in_arousal*, *reporter_in_anger*. This means that pull request reporters with high level of anger, arousal and dominance in their previous pull request are more likely to have their pull request rejected.

Second, we built two logistic classifiers, one without affect metrics and one with them, and by comparing the two classifiers we show that the affect metrics improve the prediction performance.

A deeper understanding of comments around decisions on source code contributions, in our case pull requests, will lead to the creation of policies and tools to better handle open source software as well as proprietary software using a pull request approach. We see a future where we automatize the invocation of tools for integration, conflict resolution, and notification systems, based on the affect expressed in a discussion among developers.

Most of our findings confirm intuition, but more studies, including other types of repositories, are necessary to further explore these ideas. Our work contributes to the field of human and behavioral aspects in software engineering by raising our understanding whether emotions and politeness expressed during software development influences the likelihood of an issue to be merged.

ACKNOWLEDGEMENT

The authors are thankful to Katharina Plett for proofreading the paper.

REFERENCES

[1] S. T. Acuña, M. Gómez, and N. Juristo, "Towards understanding the relationship between team climate and software quality—A quasi-experimental study," *Empirical Softw. Eng.*, vol. 13, no. 4, pp. 401–434, 2008.

[2] M. Bastian, S. Heymann, and M. Jacomy, ''Gephi: An open source software for exploring and manipulating networks,'' in *Proc. ICWSM*, vol. 8, 2009, pp. 361–362.
[3] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, ''Fast unfolding of communities in large networks,'' *J. Stat. Mech., Theory Exp.*, vol. 2008, no. 10, 2008, Art. no. P10008.
[4] F. Calefato, F. Lanubile, and N. Novielli, ''How to ask for technical help? Evidence-based guidelines for writing questions on stack overflow,'' *Inf. Softw. Technol.*, vol. 94, pp. 186–207, Feb. 2018.
[5] L. F. Capretz, ''Personality types in software engineering,'' *Int. J. Hum.-Comput. Stud.*, vol. 58, no. 2, pp. 207–214, Feb. 2003.
[6] A. Cockburn and J. Highsmith, ''Agile software development: The people factor,'' *Computer*, vol. 11, pp. 131–133, Nov. 2001.
[7] C. Danescu-Niculescu-Mizil, M. Sudhof, D. Jurafsky, J. Leskovec, and C. Potts, ''A computational approach to politeness with application to social factors,'' in *Proc. ACL*, 2013, pp. 1–10.
[8] B. de Alwis and J. Sillito, ''Why are software projects moving from centralized to decentralized version control systems?'' in *Proc. ICSE Workshop Cooperat. Hum. Aspects Softw. Eng.*, May 2009, pp. 36–39.
[9] G. Destefanis, M. Ortu, D. Bowes, M. Marchesi, and R. Tonelli, ''On measuring affects of GitHub issues' commenters,'' in *Proc. 3rd Int. Workshop Emotion Awareness Softw. Eng.*, 2018, pp. 14–19.
[10] G. Destefanis, M. Ortu, S. Counsell, S. Swift, M. Marchesi, and R. Tonelli, ''Software development: Do good manners matter?'' *PeerJ Comput. Sci.*, vol. 2, p. e73, Jul. 2016.
[11] P. Eckman, ''Universal and cultural differences in facial expression of emotion,'' in *Proc. Nebraska Symp. Motivat.*, vol. 19, 1972, pp. 207–284.
[12] A. Erez and A. M. Isen, ''The influence of positive affect on the components of expectancy motivation,'' *J. Appl. Psychol.*, vol. 87, no. 6, p. 1055, 2002.
[13] R. Feldt, R. Torkar, L. Angelis, and M. Samuelsson, ''Towards individualized software engineering: Empirical studies should collect psychometrics,'' in *Proc. Int. Workshop Cooperat. Hum. Aspects Softw. Eng.* New York, NY, USA: ACM, 2008, pp. 49–52.
[14] G. Gousios, ''The GHTorrent dataset and tool suite,'' in *Proc. 10th Work. Conf. Mining Softw. Repositories (MSR)*, Piscataway, NJ, USA: IEEE Press, 2013, pp. 233–236.
[15] G. Gousios, M. Pinzger, and A. V. Deursen, ''An exploratory study of the pull-based software development model,'' in *Proc. 36th Int. Conf. Softw. Eng. Explor. Study Pull-Based Softw. Develop. Model*. New York, NY, USA: ACM, 2014, pp. 345–355.
[16] G. Gousios, A. Zaidman, M.-A. Storey, and A. van Deursen, ''Work practices and challenges in pull-based development: The integrator's perspective,'' in *Proc. 37th Int. Conf. Softw. Eng.* Piscataway, NJ, USA, 2015. IEEE Press, vol. 1, May 2015, pp. 358–368.
[17] D. Graziotin and F. Fagerholm, ''Happiness and the productivity of software engineers,'' in *Rethinking Productivity in Software Engineering*. Berkeley, CA, USA: Apress, 2019, pp. 109–124.
[18] D. Graziotin, F. Fagerholm, X. Wang, and P. Abrahamsson, ''What happens when software developers are (un)happy,'' *J. Syst. Softw.*, vol. 140, pp. 32–47, Jun. 2018.
[19] 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.
[20] D. Graziotin, X. Wang, and P. Abrahamsson, ''Do feelings matter? On the correlation of affects and the self-assessed productivity in software engineering,'' *J. Softw., Evol. Process*, vol. 27, no. 7, pp. 467–487, 2015.
[21] R. N. Iyer, S. A. Yun, M. Nagappan, and J. Hoey, ''Effects of personality traits on pull request acceptance,'' *IEEE Trans. Softw. Eng.*, early access, Dec. 17, 2019, doi: 10.1109/TSE.2019.2960357.
[22] M. Jenkins and M. Dragojevic, ''Explaining the process of resistance to persuasion,'' *Commun. Res.*, vol. 40, no. 4, pp. 559–590, 2013.
[23] R. Lambiotte, J.-C. Delvenne, and M. Barahona, ''Laplacian dynamics and multiscale modular structure in networks,'' 2008, *arXiv:0812.1770*. [Online]. Available: https://arxiv.org/abs/0812.1770
[24] M. Mäntylä, B. Adams, G. Destefanis, D. Graziotin, and M. Ortu, ''Mining valence, arousal, and dominance: Possibilities for detecting burnout and productivity?'' in *Proc. 13th Int. Workshop Mining Softw. Repositories*. New York, NY, USA: ACM, 2016, pp. 247–258.
[25] A. Mehrabian, *Basic Dimensions for a General Psychological Theory: Implications for Personality, Social, Environmental, and Developmental Studies*, vol. 2. Cambridge, MA, USA: Oelgeschlager, Gunn & Hain, 1980.
[26] *Polite*, Merriam-Webster, Springfield, MA, USA, 2019.
[27] A. Murgia, M. Ortu, P. Tourani, B. Adams, and S. Demeyer, ''An exploratory qualitative and quantitative analysis of emotions in issue report comments of open source systems,'' *Empirical Softw. Eng.*, vol. 23, no. 1, pp. 521–564, 2018.
[28] A. Murgia, P. Tourani, B. Adams, and M. Ortu, ''Do developers feel emotions? An exploratory analysis of emotions in software artifacts,'' in *Proc. 11th Work. Conf. Mining Softw. Repositories*. New York, NY, USA: ACM, 2014, pp. 262–271.
[29] 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 Work. Conf. Mining Softw. Repositories (MSR)*, May 2015, pp. 303–313.
[30] M. Ortu, G. Destefanis, B. Adams, A. Murgia, M. Marchesi, and R. Tonelli, ''The JIRA repository dataset: Understanding social aspects of software development,'' in *Proc. 11th Int. Conf. Predictive Models Data Anal. Softw. Eng.* New York, NY, USA: ACM, 2015, pp. 1–4.
[31] M. Ortu, G. Destefanis, S. Counsell, S. Swift, R. Tonelli, and M. Marchesi, ''Arsonists or firefighters? Affectiveness in agile software development,'' in *Proc. Int. Conf. Agile Softw. Develop.* Edinburgh, U.K.: Springer, 2016, pp. 144–155.
[32] M. Ortu, G. Destefanis, D. Graziotin, M. Marchesi, and M. Tonelli. (May 2020). *Dataset—How do You Propose Your Code Changes? Empirical Analysis of Affect Metrics of Pull Requests on GitHub*. [Online]. Available: https://doi.org/10.5281/zenodo.3825044
[33] M. Ortu, G. Destefanis, M. Kassab, S. Counsell, M. Marchesi, and R. Tonelli, ''Would you mind fixing this issue? An empirical analysis of politeness and attractiveness in software developed using agile boards,'' in *Agile Processes in Software Engineering and Extreme Programming*. Helsinki, Finland: Springer, 2015, pp. 129–140.
[34] M. Ortu, G. Destefanis, M. Kassab, and M. Marchesi, ''Measuring and understanding the effectiveness of JIRA developers communities,'' in *Proc. IEEE/ACM 6th Int. Workshop Emerg. Trends Softw. Metrics*, May 2015, pp. 3–10.
[35] M. Ortu, T. Hall, M. Marchesi, R. Tonelli, D. Bowes, and G. Destefanis, ''Mining communication patterns in software development: A GitHub analysis,'' in *Proc. 14th Int. Conf. Predictive Models Data Anal. Softw. Eng.* New York, NY, USA: ACM, 2018, pp. 70–79.
[36] R. Plutchik and H. Kellerman, *Emotion, Theory, Research, and Experience*, vol. 1. London, U.K.: Academic, 1980.
[37] P. C. Rigby and A. E. Hassan, ''What can OSS mailing lists tell us? A preliminary psychometric text analysis of the apache developer mailing list,'' in *Proc. 4th Int. Workshop Mining Softw. Repositories*. Washington, DC, USA: IEEE Computer Society, May 2007, p. 23.
[38] J. A. Russell, ''A circumplex model of affect,'' *J. Personality Social Psychol.*, vol. 39, no. 6, pp. 1161–1178, Dec. 1980.
[39] J. A. Russell, ''Emotion, core affect, and psychological construction,'' *Cognition Emotion*, vol. 23, no. 7, pp. 1259–1283, Nov. 2009.
[40] M. Santos, J. Caetano, J. Oliveira, and H. T. Marques-Neto, ''Analyzing the impact of feedback in GitHub on the software developer's mood,'' in *Analyzing the Impact of Feedback in GitHub on the Software Developer's Mood*. North Bethesda, MD, USA: KSI Research Inc. and Knowledge Systems Institute Graduate School, 2018.
[41] D. Spinellis, ''Version control systems,'' *IEEE Softw.*, vol. 22, no. 5, pp. 108–109, Sep. 2005.
[42] J. Tsay, L. Dabbish, and J. Herbsleb, ''Influence of social and technical factors for evaluating contribution in GitHub,'' in *Proc. 36th Int. Conf. Softw. Eng.*, May 2014, pp. 356–366.
[43] J. Tsay, L. Dabbish, and J. Herbsleb, ''Let's talk about it: Evaluating contributions through discussion in GitHub,'' in *Proc. 22nd ACM SIGSOFT Int. Symp. Found. Softw. Eng.*, 2014, pp. 144–154.
[44] B. Vasilescu, A. Serebrenik, P. Devanbu, and V. Filkov, ''How social Q&A sites are changing knowledge sharing in open source software communities,'' in *Proc. 17th ACM Conf. Comput. Supported Cooperat. Work Social Comput.* New York, NY, USA: ACM, 2014, pp. 342–354.
[45] A. B. Warriner, V. Kuperman, and M. Brysbaert, ''Norms of valence, arousal, and dominance for 13,915 English lemmas,'' *Behav. Res. Methods*, vol. 45, no. 4, pp. 1191–1207, 2013.
[46] M. Wyrich and J. Bogner, ''Towards an autonomous Bot for automatic source code refactoring,'' in *Proc. IEEE/ACM 1st Int. Workshop Bots Softw. Eng. (BotSE)*, May 2019, pp. 24–28.
[47] Y. Yu, H. Wang, V. Filkov, P. Devanbu, and B. Vasilescu, ''Wait for it: Determinants of pull request evaluation latency on GitHub,'' in *Proc. IEEE/ACM 12th Working Conf. Mining Softw. Repositories*, May 2015, pp. 367–371.

[48] Y. Yu, H. Wang, G. Yin, and T. Wang, "Reviewer recommendation for pull-requests in GitHub: What can we learn from code review and bug assignment?" *Inf. Softw. Technol.*, vol. 74, pp. 204–218, Jun. 2016.

[49] Q. Zhang and D. A. Sapp, "Psychological reactance and resistance intention in the classroom: Effects of perceived request politeness and legitimacy, relationship distance, and teacher credibility," *Commun. Edu.*, vol. 62, no. 1, pp. 1–25, 2013.

MARCO ORTU received the B.E. and M.Eng. degrees in electrical and electronic engineering and the Ph.D. degree in software engineering from the University of Cagliari, Italy. He is a Postdoctoral Researcher with the Department of Electronic and Electrical Engineering, University of Cagliari. His main research interests include natural language processing, data mining, and empirical software engineering. He teaches web analytics and text mining for the master's students in data science at the Business School, University of Cagliari.

GIUSEPPE DESTEFANIS from the B.E. and M.Eng. degrees from the University of Pisa and the Ph.D. degree from the University of Cagliari. He was a Lecturer with the School of Computer Science, University of Hertfordshire. He has been a Postdoctoral Researcher at Brunel University and the Computer Research Institute of Montreal (CRIM), Canada, and worked closely with the Montreal aerospace industry to support testing activities during the development of flight simulators. While completing his Ph.D. studies, he visited the University of Auckland, New Zealand, and the Hong Kong University of Science and Technology. He is a Lecturer with the Department of Computer Science, Brunel University London. His research interests include mining software repositories, empirical software engineering, Agile methodologies, software metrics and patterns, blockchain, and cryptocurrencies.

DANIEL GRAZIOTIN received the Ph.D. degree in computer science from the Free University of Bozen-Bolzano, Italy. He is a Senior Researcher (Akademischer Rat) with the University of Stuttgart, Germany. His research interests include human, behavioral, and psychological aspects of empirical software engineering, studies of science, and open science. He was awarded the Data Journalism Award in 2015, the European Design Award (bronze) in 2016, and the Alexander von Humboldt Fellowship for Postdoctoral Researchers in 2017. He is an Associate Editor of the *Journal of Open Research Software* and an Academic Editor of the *Research Ideas and Outcomes* (RIO) journal.

MICHELE MARCHESI received the degree in electronic engineering from the University of Genova, in 1975.

He has been a Full Professor with the Faculty of Engineering, University of Cagliari, since 1994. Since 2016, he has been a Full Professor with the Department of Mathematics and Computer Science, University of Cagliari, where he teaches software engineering courses. He has authored over 200 international publications, including over 70 in magazines. He has been one of the first in Italy to deal with OOP, since 1986. He was a Founding Member of TABOO, the Italian association on object-oriented techniques. He has also worked on object analysis and design, UML language and metrics for object-oriented systems since the introduction of these research themes. In 1998, he was the first in Italy to deal with extreme programming (XP) and agile methodologies for software production. He organized the first and most important world conferences on extreme programming and agile processes in software engineering, Sardinia, from 2000 to 2002. Since 2014, being among the first in Italy, he has extended his research interests to blockchain technologies, obtaining significant results in the scientific community.

ROBERTO TONELLI received the Ph.D. degrees in physics and in computer engineering, in 2000 and 2012, respectively. He is currently a Temporary Researcher and a Professor with the University of Cagliari, Italy. The main topic of his research has been the study of power laws in software systems within the perspective of describing software quality. Since 2014, he has been extended his research interest to the blockchain technology. His research interests include widespread and multidisciplinary.