Destructive Criticism in Software Code Review Impacts Inclusion

SANURI DANANJA GUNAWARDENA, University of Auckland, New Zealand
PETER DEVINE, University of Auckland, New Zealand
ISABELLE BEAUMONT, University of Auckland, New Zealand
LOLA GARDEN, University of Auckland, New Zealand
EMERSON MURPHY-HILL, Google, USA
KELLY BLINCOE*, University of Auckland, New Zealand

The software industry lacks gender diversity. Recent research has suggested that a toxic working culture is to blame. Studies have found that communications in software repositories directed towards women are more negative in general. In this study, we use a destructive criticism lens to examine gender differences in software code review feedback. Software code review is a practice where code is peer reviewed and negative feedback is often delivered. We explore differences in perceptions, frequency, and impact of destructive criticism across genders. We surveyed 93 software practitioners eliciting perceived reactions to hypothetical scenarios (or vignettes) where participants are asked to imagine receiving either constructive or destructive criticism. In addition, the survey collected general opinions on feedback obtained during software code review as well as the frequency that participants give and receive destructive criticism.

We found that opinions on destructive criticism vary. Women perceive destructive criticism as less appropriate and are less motivated to continue working with the developer after receiving destructive criticism. Destructive criticism is fairly common with more than half of respondents having received nonspecific negative feedback and nearly a quarter having received inconsiderate negative feedback in the past year. Our results suggest that destructive criticism in code review could be a contributing factor to the lack of gender diversity observed in the software industry.

CCS Concepts: • **Human-centered computing → Empirical studies in collaborative and social computing**; • **Software and its engineering → Collaboration in software development**; • **Social and professional topics → Gender**.

Additional Key Words and Phrases: Software Code Review, Destructive Criticism, Diversity and Inclusion, Software Engineering

ACM Reference Format:
Sanuri Dananja Gunawardena, Peter Devine, Isabelle Beaumont, Lola Garden, Emerson Murphy-Hill, and Kelly Blincoe. 2022. Destructive Criticism in Software Code Review Impacts Inclusion. In *Proceedings of the ACM on Human Computer Interaction.* ACM, New York, NY, USA, 29 pages.

1 INTRODUCTION

The software industry is in a diversity crisis. Despite the recent push to increase gender diversity, tech jobs remain dominated by men. Women represent less than 25% of the tech workforce at companies like Google and Facebook[1]. In Open Source Software (OSS), less than 10% of developers are women [80]. People who identify as non-binary or gender diverse are also under-represented at less than 1% of the tech workforce in big tech companies[1]. We know that gender diversity on

*Corresponding author
[1]see https://diversity.google/annual-report/ and https://diversity.fb.com/read-report/

CSCW'22, November 12–16, 2022, Taipei, Taiwan

software teams brings many benefits (e.g. improved productivity [80], financial performance [9], innovation [57], problem solving [14], and user experience [20]). It is, therefore, critical to improve gender diversity in tech. Yet, there are reports of a culture problem in the software industry that hampers diversity and inclusion [26]. Analysis of data from StackOverflow's Developer survey found that women, non-binary, and transgender software developers are less confident in their development skills compared to men, possibly due to stereotypes and unconscious bias [70]. Recent studies of OSS projects have found evidence of gender biases and discrimination (e.g. [50, 55, 75]). There are also reports that a similar discriminatory, toxic culture exists in software companies like Apple [51] and Facebook [4]. Studies have also found that both sexual harassment and gender discrimination are reported in the software industry at alarming rates [29, 84]. While some developers have spoken out against this culture, a better understanding is needed of the impact of specific negative behaviours that contribute to this culture to enable interventions that can lead to a more positive culture and increased diversity and inclusion in the software industry.

One attributing factor could be "destructive criticism", which is defined as negative feedback that is both nonspecific and inconsiderate [10]. Here, inconsiderate is defined as negative feedback that is delivered in a harsh or sarcastic tone, includes threats, or attributes poor task performance to flaws of the individual [10]. Research in other domains found that destructive criticism often provokes negative reactions in the recipient, such as reduced productivity and motivation [10]. Studies have also found that the impact of destructive criticism can be higher for women for some types of tasks [10]. If this is true for destructive criticism for software tasks, it could have important implications for diversity and inclusion in the software industry.

One particular software task where destructive criticism might occur is in code review, where software code contributions are examined by other members of the team. Code review is a common practice on software teams, and it has long been used to improve software quality [39]. Code review also brings many other benefits including improving knowledge sharing in teams and providing learning opportunities [7, 17, 64]. Bosu et al. recently found that code review also impacts impression formation between developers, including perceptions of trust, reliability, and expertise [17]. Developers have reported that friendships can even be developed through the informal discussions that occur through code review [16]. Thus, code review is important not just for its technical benefits, but also to foster community building and to provide a sense of belonging in software teams.

Developers report that code review can also create toxic, unsupportive environments (e.g. [3, 65]). Anecdotal evidence points to the existence of destructive criticism in code review in the form of judgmental questions, sarcasm, or comments that contain only negative emojis [65]. However, no studies have examined the impact of destructive criticism during software code review.

In this study, we examine the perceived impact of destructive criticism in software code review through an online questionnaire. Scenario-based questions, or vignettes, are used where participants are presented with a scenario where they receive code review comments containing either constructive or destructive criticism. We investigate the impact destructive criticism has on the participants' mood, willingness to continue working, and written responses to the criticism. We also explore participants' perceptions of the helpfulness, validity, and appropriateness of destructive criticism. We examine the differences in reported impact and perceptions across genders and other reported demographics (e.g., years of experience, self-reported competency, self-efficacy). Our study was guided by the following research questions:

RQ1 What are the perceptions of destructive criticism (compared to constructive criticism) and are there differences across genders or other demographics?

RQ2 How often do respondents report giving and receiving destructive criticism and are there differences across genders or other demographics?

RQ3 What is the impact of destructive criticism (compared to constructive criticism) and are there differences across genders or other demographics?

The results suggest that destructive criticism is common in software code review, with more than half of respondents reporting they have received nonspecific negative feedback and nearly a quarter reporting that they have received inconsiderate negative feedback in the past year. When writing code review comments, most respondents believed that improving code quality is equally as important as considering how the code author will react to the negative feedback. However, nearly a quarter of respondents believe that improving code quality is more important. The open-ended responses illustrated these conflicting views, with some participants noting that harsh feedback is sometimes necessary and easier to parse, while others explained that inconsiderate feedback is never acceptable. When receiving code review comments, many respondents believe it is acceptable to receive inconsiderate feedback if code quality will be improved. However, we found significant gender differences in these results with women being significantly less likely to approve of receiving inconsiderate code reviews, regardless of code quality benefits. Women also reported being less motivated to continue working with the developer after receiving destructive criticism. Non-binary participants also found destructive criticism less appropriate and less helpful in improving code quality (note that our sample size of non-binary participants was very small ($n = 3$), and we do not know if these responses are representative). Thus, our results show that destructive criticism could contribute to the lack of gender diversity on software teams.

The remainder of this paper is organized as follows: Related work is described in Section 2. The methodology and results are presented in Sections 3 and 4. In Section 5, we further discuss the implications of the results, future research directions, and threats to validity of our study. Finally, we offer a brief conclusion in Section 6.

2 RELATED WORK

In this section, we first discuss the literature on negative performance feedback and destructive criticism from other domains, then we discuss related literature in the field of software engineering.

2.1 Negative Performance Feedback and Criticism at Work

Feedback on task performance has been studied in many domains. While the intention of negative task performance feedback is often to improve future task performance, it has been found to be associated with many adverse effects. Negative performance feedback is related with lower self-efficacy and inferior future task performance [49]. It has also been found to be associated with higher turnover intentions [13] and negative emotions in the recipient [53, 54, 59]. However, code review feedback, which is the subject of this study, often must deliver negative task performance feedback by the very nature of the process. Thus, this literature indicates that code review comments have the potential to cause negative reactions in the recipients.

There are different manners in which negative task performance feedback, and negative feedback in general, can be conveyed. One differentiation is whether the feedback is constructive or destructive. Studies of destructive criticism originate in the field of applied psychology. Baron defined destructive criticism as both inconsiderate and nonspecific [10]. Conversely, he defined constructive criticism as negative feedback that was still considerate and provided specific improvement suggestions. Destructive feedback has been found to have detrimental effects on factors such as mood, productivity, and motivation [10, 13]. Other forms of harsh criticism at work, like insults or bullying from a supervisor, have been found to have similar adverse impacts including higher turnover intentions, reduced self-efficacy, and reduced work performance [34, 36]. In this study, we use the destructive criticism lens.

It is currently unknown how common destructive criticism occurs in software code review. Anecdotal evidence suggests that both nonspecific and inconsiderate code review comments exist (e.g. [3, 65]), but there have not been comprehensive studies to show how common they are. It is important to understand how often destructive criticism occurs in software code review given the adverse impacts that are associated with this type of criticism in other domains.

2.2 Gender Differences in Destructive Criticism

Some studies on destructive criticism have also considered gender as a confounding variable in their analysis and reported on gender differences [2, 10, 62]. Baron found gender differences in task performance after destructive criticism for clerical tasks that involved finding prices of items in a book [10]. Baron found that women were more strongly impacted by destructive criticism in that task. However, in the same study, there were no significant gender differences in a proofreading task, which involved identifying spelling and grammar errors in text. Similarly, Raver et al. did not find any gender differences in response to destructive criticism in either of the tasks in their study, which involved giving a presentation and a writing task where participants created sentences from a set of words [62]. In a study that did not involve work-related tasks, but examined destructive criticism from relatives, Allred and Chambless found that women were more upset by this criticism than men [2]. Thus, the gender differences identified in these studies have varied, with differences being identified for only a subset of the tasks in the studies. This could indicate that the type of task is a factor in whether the impact of destructive criticism will vary with gender.

No studies have investigated the impact of destructive criticism specifically in a software engineering setting. Given the lack of diversity in software engineering, it is important to understand both how common destructive criticism occurs and potential gender differences in the impacts of destructive criticism in a software engineering setting. This is the first study to apply the destructive criticism lens to software code review.

2.3 Communication, Criticism, and Emotions in Software Engineering

Software Engineering is a socio-technical activity [33]. Software developers must work together to create and maintain large, complex software systems [15]. Given this, much research has investigated the social aspects of software engineering [74]. Research in this area has evolved from studies examining when developers coordinate or share knowledge with each other to studies of *how* these social activities are done, including studies on emotion, politeness, and sentiment (positive, negative, or neutral polarity of text).

2.3.1 Analysis of Sentiment and Toxicity in Communications. Many studies have shown that not all conversations between software developers are purely technical in nature. Natural language processing techniques have been used to identify sentiment and emotion in communications. Studies have found that developers convey both positive and negative sentiment in their technical communications, including in mailing list messages [79], commit messages [48, 71], code review comments [58], and issue comments [52]. Paul et al. reported that the ratio of negative sentiments is higher than that of positive sentiments for code review comments [58]. Tourani et al. classified the sentiment in mailing list messages and showed that developers expressed sentiments such as aggression, dissatisfaction, and sadness in their messages [79]. Several studies have also found that anger is commonly expressed in software issue reports [43, 56].

Studies have since investigated the impact of negative sentiment in software engineering communications. Ortu et al. found that negative sentiment in issue comments was associated with slower resolution times [56]. Conversely, issues with more polite comments were correlated with faster resolution times [32]. Code reviews that receive negative comments have also been found to take longer to complete [37]. These results are in line with studies in other domains which found that negative sentiment is associated with negative outcomes such as reduced productivity, quality, and job satisfaction [5, 31]. While these studies have shown that negative sentiment or emotion in text can be associated with negative outcomes, negative sentiment can be expressed for many reasons like healthy disagreements and is not in itself indicative of unhealthy communications.

Other research has focused on unhealthy interactions on software projects. Squire and Gazda found that profanity and insults are common in communications on open source software projects [72]. Cheriyan et al. found a toxic culture on Stack Overflow, a community forum popular with software developers [22]. Recent research has found relatively low levels of toxic discussions, such as insults or personal attacks, in software development discussions in general [61, 67]. These studies have been enabled by recent advances in natural language processing, which can automatically detect toxicity in text (e.g. Google's Perspective API). Studies of toxicity in software engineering have looked at communication on software teams in general. Given that negative performance feedback has been found to have many detrimental effects as described in Section 2.1, there may be differences in the impact of toxic communications depending on their context. Our study specifically focuses on negative feedback received during software code review, a practice which often requires negative feedback to be delivered, rather than all software engineering communications.

The study closest to ours is a recent study at Google, which investigated "pushback" during software code review, which they defined as "negative interpersonal interactions" with peers during a code review. They found that negative experiences, including personal attacks and harsh communications, are relatively rare in code review at Google, but that they have negative repercussions when they occur [35]. However, pushback has only been examined in a corporate context thus far. In contrast, we survey developers from industry, developers from open source software, and students to learn about the perceptions and perceived impacts of destructive criticism in software code review.

2.3.2 Gender Differences in Communications. Studies have examined how written communications on software projects vary across genders, and gender differences have been found. Imtiaz et al. found that, compared to men, women express more neutral sentiment and use less profanity in their conversations on GitHub [50]. Similar results were obtained in a study by Paul et al. which examined code review comments of six open source software projects and also found that women used more neutral sentiment and less expletives compared to men. This is in contrast to other domains, such as social media, where women are far more likely to use positive sentiment in communication channels and negative sentiment is much rarer and not associated with gender [76]. Imtiaz et al. hypothesize that women may stay more neutral in their conversations and avoid expletives due to a "tightrope" bias, where women, who are underrepresented in the software industry, feel they have a narrower band of socially acceptable behaviour. In addition to women being less likely to express positive or negative sentiment, it has also been found that comments directed towards women offer less positive encouragement and more negative criticism compared to those directed at men [58]. This indicates potential bias and discrimination towards women, particularly if this feedback is destructive. In this study, we examine in more depth the frequency of and impacts of destructive criticism during software code review with a lens on gender differences.

The studies of gender differences in software developer communications have only considered binary genders. This is likely because in these studies gender has been approximated, since gender information of software developers is not available in large-scale repository analysis. In our study, we survey software developers about their perceptions of destructive criticism. Thus, we do not need to approximate gender. However, we received only 3 responses from developers who identify as non-binary.

2.3.3 Software Developers' Perceptions of Emotions. In addition to studies that have examined communications in software projects, studies have also investigated the perceptions of software developers around emotions. Wrobel et al. found that positive emotions during software development tasks are associated with higher productivity [83]. Ford and Parnin found that developers often experience severe frustration in their tasks [42]. This is particularly problematic, as frustration has been found to have a dramatic negative impact on developer productivity [83]. Graziotin et al. found that developer unhappiness can negatively impact their own mental well-being and can also have detrimental effects to the software development process and the software quality [45, 46]. Notably, developer productivity and the quality of the software produced are impacted the most negatively from developer unhappiness. These results are in line with literature from psychology which has found that negative emotions interfere with cognitive functioning [24]. These studies show the importance of considering developer well-being given the negative consequences when developers experience strong negative emotions during work tasks. Since negative emotion is also associated with higher turnover rates and intention to quit [18], ensuring a positive working environment is also important for developer retention. In these studies, perceptions of emotions and their impact was the primary goal. In our study, we investigate the impacts of destructive criticism received during software code review, with developer mood being one of the factors studied.

3 METHODOLOGY

This study used an anonymous online questionnaire to gather data about how developers perceive destructive feedback in software code review with a focus on the impact of destructive criticism.

3.1 Questionnaire Design

The questionnaire contained five sections: a preparation question, vignette-style questions, questions about experiences with destructive criticism, questions about opinions of destructive criticism in general, and demographic questions including a standard self-efficacy assessment. All questions were optional. A complete list of questions and associated answer options is available in our replication pacakge [47].

The questionnaire was piloted with three fourth year Bachelor of Engineering students at the University of Auckland specializing in software engineering in order to ensure it was understandable and could be completed in less than 15 minutes. All pilot study participants had some working experience as software developers and experience with code review. Two of the researchers discussed the questionnaire and potential improvements with the participants after they had all completed the pilot study. Several changes were made based on this pilot including a reordering of questions to improve the flow and better presentation of several of the questions to ensure the clarity of the questions. The participants in the pilot study pointed out that the flow of logic between questions associated with the two Vignette-style Questions was confusing. In response to this feedback, we altered the order of these questions for a more logical flow. Additionally, within each Vignette, the code snippet was moved to be integrated into the vignette description (rather than after the description) due to suggestions made by the pilot study participants. These changes were discussed and agreed with the pilot study participants. The responses from the pilot study were not included in the analysis. The five sections of the final questionnaire are described in the following subsections.

3.1.1 Preparation Question. The first question of the questionnaire aimed to prepare the participant to answer questions about code review. The participant was presented a snippet of code where another developer had made an obvious, novice-level mistake. The participant was asked to review the code and write a code review comment.

3.1.2 Vignette-style Questions. This section of the questionnaire contained vignettes, which are descriptions of hypothetical situations [1]. Each vignette had a series of associated questions about the perceived impact that the hypothetical situation would have on the participant. Vignettes are an alternative to lab experiments and enable participants to consider real-world scenarios without subjecting participants to the treatment under study. They also allow for a standardized scenario to be shown to all study participants. By creating detailed and concrete vignettes, they have been shown to be able to approximate real-world decision making and produce more valid and more reliable opinions in questionnaires compared to abstract questions [1]. Vignettes have been used successfully in studies on the impacts of destructive criticism in other domains (e.g., [62, 77]).

The questionnaire contained two vignettes which contained hypothetical scenarios where the participant was asked to imagine they had submitted a piece of code for review and had just received feedback. While all participants saw the same hypothetical scenario description and the same code snippet, the hypothetical feedback on the code snippet was randomised so that some participants saw a version of the feedback that was constructive and some participants saw a version of the feedback that was destructive. Across the two vignettes, each participant saw one with destructive feedback and one with constructive feedback, with approximately half of the participants seeing the destructive feedback in the first vignette and the remainder seeing the destructive feedback in the second vignette.

The destructive feedback was nonspecific and inconsiderate (e.g., "This is seriously obtuse"). In contrast, the constructive feedback contained specific suggestions framed in a more considerate manner (e.g., "You don't need to protect instanceOf against null pointers"). The feedback provided in the vignettes was obtained from actual code review comments made on open source software projects to ensure the scenarios mimic real-world code review interactions. Similarly, the code snippets presented in the vignettes were snippets of real code from open source software projects.

After reading the vignette and seeing the associated code snippet and feedback, participants were then asked a series of questions to understand the participants perceived reaction to the feedback. Questions probed the participant about their mood (using a subset of the multidimensional mood questionnaire [73]), their motivation to continue working, and their perceptions on the helpfulness, validity, and appropriateness of the feedback. The participant was also asked what they would write if they were to write a response to the feedback. The questions and associated answer options are shown in Table 1. The responses to these questions were used to answer RQ1 (perceptions of destructive criticism) and RQ3 (impact of destructive criticism).

3.1.3 Experience Questions. This part of the questionnaire asked participants to reflect on their past experiences of giving and receiving negative feedback during software code review. Questions asked about the frequency the participant both gave and received code review comments in general (to establish a baseline of activity) and two types of destructive criticism: nonspecific and inconsiderate negative feedback. The questions in this section were:

- In the past year, how often have you *received* code review comments?
- In the past year, how often have you *received* comments that have contained nonspecific negative feedback? E.g. "This is a hack"
- In the past year, how often have you *received* comments that have contained inconsiderate negative feedback? E.g. "You have to be stupid to do this"
- In the past year, how often have you *given* code review comments?
- In the past year, how often have you *given* comments that have contained nonspecific negative feedback? E.g. "This is a hack"
- In the past year, how often have you *given* comments that have contained inconsiderate negative feedback? E.g. "You have to be stupid to do this"

Question	**Response Options**
How do you think this feedback would make you feel? (select all that apply)	good; bad; nervous; tense; great; composed; uncomfortable; at ease
Please rate your agreement with the following statements: This feedback would help me to improve the code This feedback is valid This feedback is appropriate	Strongly disagree; Somewhat disagree; Neither agree nor disagree; Somewhat agree; Strongly agree; I don't know
How motivated do you think you would be to: Continue working on this task? Continue working with this developer? Continue working on this project?	Very unmotivated; Somewhat unmotivated; Neither motivated nor unmotivated; Somewhat motivated; Very motivated; I don't know
If you were to write a response to this feedback, what would you write? (Leave blank if you would not respond)	n/a

Table 1. Questions asked of the study participants for the vignettes where the participant receives constructive or destructive feedback

Question	**Response Options**
Please rate your agreement with the following statements: Destructive criticism is harmful Destructive criticism will cause a negative reaction for the recipient	Strongly disagree; Somewhat disagree; Neither agree nor disagree; Somewhat agree; Strongly agree; I don't know
When writing code review comments, I believe it is more important to write:	Feedback that will improve code quality; Feedback that will not cause a negative reaction for the code author; Both are equally important; I don't know
When receiving code review comments, I don't mind getting inconsiderate feedback as long as the feedback helps to improve the code quality.	Strongly disagree; Somewhat disagree; Neither agree nor disagree; Somewhat agree; Strongly agree; I don't know
Is there anything else you want us to know about how you feel about negative feedback in software code review?	n/a

Table 2. Questions about opinions of destructive criticism in general

For each question, the participant was asked for the frequency across three types of projects: academic, open source, and commercial software projects. The response options for these questions were: Never; Once or more a year; Once or more a quarter; Once or more a month; Once or more a week; N/A. These responses were used to answer RQ2 (frequency of destructive criticism).

3.1.4 Opinion Questions. The next section of the questionnaire asked participants for their opinion on negative feedback on software projects and on the appropriateness of destructive criticism in general. The opinion questions are shown in Table 2. The responses to these questions were used to answer RQ1 (perceptions of destructive criticism).

3.1.5 Demographics Section. The final section of the questionnaire collects demographics of the participants, including a self-efficacy assessment using questions from a standard self-efficacy assessment tool [21]. Participants were asked their gender (open-ended), years of work experience, highest level of education, self-reported competence in software development, current profession, age, and number of open source software projects contributed to. The demographic questions are used to enable the analysis of differences in perceptions across genders, account for potential confounding variables in our analysis, and to provide context of the study setting.

3.2 Participant Recruitment

Convenience sampling was used to recruit participants [38]. The survey was available online through the Qualtrics platform[2]. Participants were recruited by posting advertisements on social media (e.g., Twitter and LinkedIn) and online communities (e.g., local software meetup groups including one for Software Engineering students at the University of Auckland). It should be noted that at the University of Auckland, Software Engineering students regularly develop software projects in teams and also have a requirement of 800 hours of professional work experience before graduation. Our advertisement elicited responses from software developers. While there was no enforced cut-off criteria for participating, all participants reported to have at least some working experience in software development (more details provided in Section 3.3). The questionnaire was open for two months before data was analysed.

3.3 Study Participants

We received 93 complete responses. Of these, 43 participants were women, 43 were men, 3 were non-binary, and 4 did not disclose their gender. Table 3 shows the number/percentage of respondents that belong to different demographic groups. Note that since all questions were optional, some participants did not provide all demographic information. Table 3 reports only those participants who responded for each question. Years of working experience was an open-ended question, so additional details are available beyond the coarse groupings shown in Table 3. The minimum response was 1 internship (<1 year), the maximum response was 28 years, and the median and mean were 5 years and 7.6 years, respectively. Note that the current occupation question allowed participants to select multiple options. There were six participants who selected both professional and student.

3.4 Ethics

The survey had ethics approval from the University of Auckland's Human Participants Ethics Committee.

3.5 Data Analysis

The data analysis was done using R. To examine differences in perceptions, frequency, and impact of destructive criticism across demographics, we developed regression models using the 'MASS' R package to model the relationship between each variable of interest and various demographics of the participants collected in the questionnaire [81]. Ordinal logistic regression was used since the dependent variables were the responses to Likert-scale questions, thus they are ordered categorical variables. However, in one case (when the dependent variable was the frequency of inconsiderate feedback received in code review), ordinal logistic regression was not suitable due to the non-normally distributed dependent variable. Thus, for this dependent variable, a log-log model was used.

[2]https://www.qualtrics.com/

Demographic Type	Group	Respondents
Gender	Men	43 (46.2%)
	Women	43 (46.2%)
	Non-binary	3 (3.2%)
Self-rated Competence	Low	5 (5.4%)
	Average	22 (23.7%)
	Moderately High	40 (43.0%)
	High	22 (23.7%)
Current Occupation*	Professional	76 (81.7%)
	Student	19 (20.4%)
Highest level of education	High School	1 (1.1%)
	Some Undergraduate	13 (14.0%)
	Undergraduate	45 (48.4%)
	Post Graduate	32 (34.4%)
Age	18 - 24	25 (26.9%)
	25 - 34	38 (40.9%)
	35 - 44	18 (19.4%)
	45 - 54	8 (8.6%)
Past Contributions to Open Source Software projects	Never contributed	47 (50.5%)
	1 project	13 (14.0%)
	2-5 projects	23 (24.7%)
	5-10 projects	4 (4.3%)
	10+ projects	4 (4.3%)
Years of Experience	<= 5 years	49 (52.7%)
	>5 years	40 (43.0%)

* This question allowed participants to select more than one answer option.

Table 3. Respondent Demographics. Participants who did not provide a response for a particular question are not reported.

While our primary interest is examining gender differences, we collected additional demographic information to be used as control variables in our models to account for potential confounding variables. The independent variables in each of the regression models are:

- gender (categorical: man, woman, non-binary)
- highest education level (categorical: less than undergraduate, undergraduate, postgraduate) *Note:* high school and some undergraduate were combined as a single category for this analysis given the small number of participants that selected high school (see Table 3).
- experience in years (numeric)
- self-efficacy score, computed as average response across the eight self-efficacy questions (numeric)
- self-reported competence (categorical: low/average, moderately high, high) *Note:* low and average competency were combined as a single category for this analysis given the small number of participants that selected low competency (see Table 3).
- has contributed to one or more OSS projects (binary)
- is a student (binary)

Variable	**VIF**
highest educational level	1.66
experience (years)	1.67
self-efficacy score	1.24
self-reported competence	1.86
has contributed to OSS	1.21
is student	1.63
gets feedback once or more a year	1.14
gets feedback once or more a quarter	1.55
gets feedback once or more a month	1.71
gives feedback once or more a year	1.26
gives feedback once or more a quarter	1.72
gives feedback once or more a month	1.70

Table 4. Variance inflation factors (VIF) for each of the independent variables from all models.

In addition, for the regression models where frequency of giving or receiving destructive criticism was the variable of interest, the reported frequency of giving or receiving any code review feedback were used as independent variables.

While we also collected the age of the participants, this was highly correlated with years of experience (Pearson correlation 0.84). Since exact years of experience was collected (compared to age brackets), age was excluded from the models to prevent multicollinearity issues. Variance inflation factors (VIF) for each of the remaining independent demographic variables (including the participants' responses related to frequency of giving and getting code review feedback, since these are used as independent variables in one of our models) show no multicollinearity issues (see Table 4).

For the regression models, goodness-of-fit was measured using the Pulkstenis-Robinson chi-squared test with the 'generalhoslem' R package, which has been found to be suitable for testing goodness-of-fit for regression models with categorical dependent variables [60]. Results are shown in Table 5. A small p-value indicates that the model does not fit the data well. Thus, for most of our models, there is no evidence of lack of fit. However, four of the models have evidence of a lack of fit (destructive criticism is valid; destructive criticism is appropriate; freq. receiving nonspecific criticism; and motivation to continue working on the project), indicating that while we have found some significant predictors in the demographic variables, there are potentially other factors that influence perceptions, frequency, and impact of destructive criticism that are not included in our models. It is also possible that non-linear relationships exist between our variables. This is further discussed in Section 5.1.

When reporting results for the regression models, significant results are bolded and asterisks are used to denote p-values where * $p<0.05$, ** $p<0.01$, and *** $p<0.001$. The model coefficients are reported in the Value columns and standard errors are reported in the SE columns.

To examine differences between responses for constructive and destructive criticism, Chi-squared tests and one-sided Mann–Whitney U-tests were used. Effect sizes were calculated using Cohen's D [25]. Cohen's D reports the standardized mean difference between two groups (computed by subtracting the mean of one group from the mean of the other group and dividing this by the standard deviation). Thus, a d score tells you by how many standard deviations the means of two groups differ. With this measure, $d<0.2$ is considered a negligible effect, $d>=0.2$ is considered a small effect, $d>=0.5$ is considered a medium effect, and $d>=0.8$ is considered a large effect [25].

Model	χ^2	p
Table 6: destructive criticism helps improve code	324.62	0.66
Table 6: destructive criticism is valid	359.34	0.02
Table 6: destructive criticism is appropriate	400.65	0.006
Table 7: don't mind receiving destructive criticism	321.61	0.76
Table 8: freq. give nonspecific criticism	217.07	0.93
Table 8: freq. receive nonspecific criticism	397.78	0.004
Table 8: freq. receive inconsiderate criticism	184.92	0.99
Table 9: motivation continue working on task	323.47	0.56
Table 9: motivation continue working with developer	355.19	0.23
Table 9: motivation continue working on project	448.5	<0.001

Table 5. Pulkstenis-Robinson chi-squared goodness-of-fit test results for the regression models

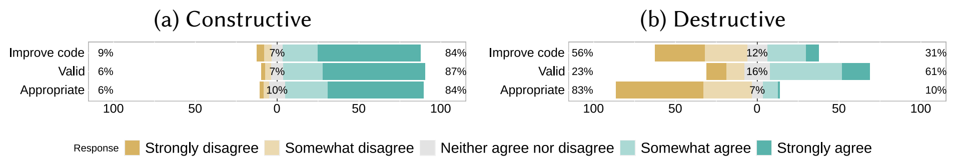

Fig. 1. Reactions to the statements "This feedback would help me to improve the code", "This feedback is valid", and "This feedback is appropriate" in regards to the feedback given in the vignette questions, either constructive or destructive.

To compute average responses, Likert scale responses were converted to integers from -2 (strongly disagree / very unmotivated) to +2 (strongly agree / very motivated).

To analyze open-ended responses, Thematic Analysis was used [19]. This analysis was performed independently by two of the authors, who both have experience in the software industry and are current Software Engineering PhD students. Both authors went through all responses and coded them individually. Final themes were developed through iterative discussions between the two authors and regular discussions with a third author (an experienced Software Engineering researcher who has also worked as a software engineer in industry). Conflicts were discussed between these three authors until a consensus was reached. This analysis was performed blind to who made the comment and their reported demographic information (e.g. gender and years of experience). However, after the themes were finalized and all responses were categorized, the demographics of the participants were qualitatively examined to identify trends in the responses. Since years of experience was a numeric response, we considered two levels of experience for this analysis. Those with more years experience than the median (5 years) were considered more experienced participants. Participants with less than or equal to the median, were considered less experienced.

4 RESULTS

4.1 Perceptions of Destructive Feedback (RQ1)

To answer this RQ, we asked participants both about their perceptions of the specific destructive feedback they received in the vignette questions and about their perceptions of destructive criticism in general. For the vignette questions, participants rated their agreement that the feedback would help improve their code, was valid, and was appropriate.

	would help improve code		is valid		is appropriate	
	Value	SE	Value	SE	Value	SE
gender (non-binary)	-2.02	1.34	-0.10	1.11	-1.45	1.46
gender (woman)	-0.68	0.45	-0.37	0.46	**-1.13** *	0.48
experience in years	-0.02	0.04	-0.04	0.04	0.03	0.04
self-efficacy score	0.06	0.47	0.37	0.50	-0.49	0.52
education (undergraduate)	-0.60	0.75	**-2.03** **	0.82	0.34	0.84
education (postgraduate)	-0.10	0.75	-1.54	0.83	-0.45	0.86
has contributed to OSS	0.05	0.44	0.23	0.47	0.52	0.47
competence (mod. high)	-0.72	0.54	0.02	0.57	0.11	0.63
competence (high)	0.20	0.69	1.02	0.78	1.03	0.84
is a student	-0.26	0.65	**-1.74** *	0.75	0.79	0.74

Table 6. Ordinal regression models for responses related to perceptions of destructive criticism received in vignette questions.
(* $p<0.05$, ** $p<0.01$, *** $p<0.001$)

4.1.1 Constructive vs Destructive. Destructive feedback in the vignettes was viewed as less likely to help to improve the code, less valid, and less appropriate compared to constructive feedback (see Figure 1). These differences were all statistically significant with a large effect. The average Likert agreement for the feedback would help to improve the code was -0.47 for destructive and 1.34 for constructive (Mann-Whitney U=1233.5, $p<0.001$; Cohen's D=-1.48). For the feedback is valid, average agreement was 0.43 for destructive criticism and 1.41 for constructive criticism (Mann–Whitney U=1699.5, $p<0.001$; Cohen's D=-0.89). For the feedback is appropriate, the average agreement was -1.25 for destructive criticism, while it was 1.35 for constructive criticism (Mann-Whitney U=462, $p<0.001$; Cohen's d=-2.65). This attitude towards destructive criticism was also reflected in participants' written responses to the feedback, where it was more common for responses to constructive criticism to agree with the criticism while destructive criticism often received comments disagreeing with the manner of discourse (see Section 4.3.3).

4.1.2 Destructive Criticism: Gender and other demographic differences. The results of the ordinal logistic regression for the three questions (would help improve code, is valid, is appropriate) are shown in Table 6. As can be seen, compared to men, women report stronger disagreement with the statement that the destructive criticism received in the vignettes is appropriate. The average agreement to this statement from men was -0.95, while women's agreement averaged -1.55. Despite this, women were not more likely than men to disagree with the manner of discourse in the criticism in their written responses to the feedback (see Section 4.3.3).

Participants who are current students more strongly disagreed with the statement that the destructive criticism received was valid. For helping to improve the code, there were no statistically significant differences across any of the demographic variables.

4.1.3 Attitudes towards destructive criticism in general. Participants were also asked about their opinions on destructive criticism in general. Figure 2 shows responses to three statements: "Destructive criticism is harmful"; "Destructive criticism will cause a negative reaction for the recipient"; "When receiving code review comments, I don't mind getting inconsiderate feedback as long as the feedback helps to improve the code quality". While there is a relative consensus that destructive feedback is harmful and causes a negative reaction in a recipient, opinions are more varied on

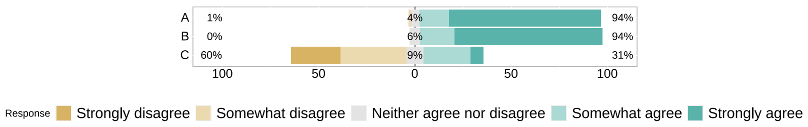

Fig. 2. Agreement to general statements about destructive criticism. A) Destructive criticism is harmful; B) Destructive criticism will cause a negative reaction for the recipient; and C) When receiving code review comments, I don't mind getting inconsiderate feedback as long as the feedback helps to improve the code quality

	Value	**SE**
gender (non-binary)	0.36	1.32
gender (woman)	**-1.54** ***	0.46
experience in years	-0.05	0.04
self-efficacy score	0.54	0.49
education (undergraduate)	0.42	0.80
education (postgraduate)	0.24	0.79
has contributed to OSS	0.45	0.44
competence (mod. high)	-0.29	0.59
competence (high)	-0.07	0.75
is a student	0.98	0.74

Table 7. Ordinal regression model for responses on whether the participant would mind receiving inconsiderate feedback if the feedback helped to improved code quality.
(* p<0.05, ** p<0.01, *** p<0.001)

whether respondents would mind receiving destructive criticism if the feedback helped to improved code quality. For this question, the mean response for women was -0.98, while it was -0.05 for men, showing that women more strongly disagreed with the statement "When receiving code review comments, I don't mind getting inconsiderate feedback as long as the feedback helps to improve the code quality". The ordinal logistic regression results (Table 7) show that this difference is statistically significant. The variations in opinions around the appropriateness of destructive criticism were also evident in participant's open-ended responses (see Section 4.1.4).

Participants were also asked about their opinions on writing code review comments, where they were asked which of the following were most important: "Feedback that will improve code quality"; "Feedback that will not cause a negative reaction for the code author"; "Both are equally important". Most participants, 67 of 88 (76%), responded that both improving code quality and preventing a negative reaction in the recipient were equally important when writing code review comments. However, there was still a significant number of participants, 20 of 88 (23%), who said that feedback that will improve code quality is more important than feedback that will not cause a negative reaction for the code author. Again, this is in line with some of the open-ended responses that indicate that some developers do not mind receiving destructive criticism (see Section 4.1.4). For example, one participant said, *"In technical discussions, I prefer direct wording to have-a-good-day sugar-coated expressions. It's just easier to parse and act upon, though may sometimes appear harsh or inconsiderate."*. Only one participant answered that not causing a negative reaction in the recipient is more important.

4.1.4 Additional Comments from participants.

At the end of the questionnaire, participants were given the opportunity to provide any additional comments related to negative feedback during software code review in an open-ended question. Four themes emerged in these responses.

Some developers do not mind destructive criticism. Some participants made comments indicating that destructive criticism was acceptable and sometimes even necessary. One participant said *"'this is a hack' is not negative at all in my books. In technical discussions, I prefer direct wording to have-a-good-day sugar-coated expressions. It's just easier to parse and act upon, though may sometimes appear harsh or inconsiderate."* One participant justified destructive criticism saying *"I think that sometimes people get angry because they really care about a project, and I can excuse it every now and then but I understand that it can discourage people and I think most people are just doing their best."* Other participants said that they do not react to destructive criticism. Instead, they *"extract the added value from the comment (i.e. the valid part) while ignoring the destructive part"*. One respondent indicated that destructive criticism is sometimes necessary, saying, *"If a developer continues to make the same mistake over and over again, sometimes leaving negative [feedback] is the only way to get through."*

Others are against destructive criticism. Many participants made comments against destructive criticism in general. Some participants said that they are fine with blunt comments as long as they provide useful information, but that inconsiderate language was problematic. For example, one participant said *"It is definitely possible to criticise someone's code without doing so in a rude way. Even when someone gets straight to the point it's fine, the main issue is when people add in negative emotive language/insults or generalise the criticism."* Other participants pushed back against nonspecific negative feedback. For example, *"Negative is alright - but not both negative and nonspecific. It doesn't help either party."* Participants noted that code review comments should be both considerate and detailed to enable learning, *"I think it's important to indicate when the code is bad, but it should be communicated in a considerate way and the reviewer should explain why the code is bad so that the person can learn."* Some participants described the importance of being polite when providing feedback. For example, *"I think pointing out the code is wrong or can be improved is important, but that has nothing to do with the author. You don't need to insult someone to tell them they're wrong."* Others just noted disagreement with destructive criticism in general. One participant said, *"I think it (destructive criticism) is extremely disrespectful and unproductive."*

However, some of the comments point out that writing good code review comments is difficult, showing that destructive criticism may not always be intentional. For example, one participant said, *"Sometimes no matter how carefully you word your feedback a person will have a negative reaction to it."* Another said, *"I assume I probably give unhelpful feedback on PRs [pull requests] at least once a year, despite my best intentions."*

Differences in response to destructive criticism. Some participants noted that they preferred speaking to the code reviewer directly if destructive criticism was received. For example, one participant said, *"I have always found speaking to reviewers the easiest way to resolve conflicts in code reviews"*. Another participant said, *"If I received inconsiderate feedback at work I would feel both comfortable and motivated enough to address the issue with the person (face to face if possible)..."*. In contrast, one participant noted *"I am quite conflict-adverse so I would never respond to an inconsiderate piece of feedback with any sort of confrontation, instead I would probably just try my best to get useful information out of them."*

Consequences of destructive feedback. Some participants wrote about the negative impacts they have faced due to receiving destructive criticism. Many of these responses described the negative impacts inconsiderate negative feedback can have on team dynamics. For example, one participant wrote, *"Receiving inconsiderate feedback makes you question how the other person values you and your contribution and your skills and therefore decreases team unity, team work and the* *ability of a team to build strong systems together."* Another participant said, *"It creates an unsafe place where people are not confident enough to ask for help or how they can improve."* Other participants described the effects on individuals. For example, *"It can really impact confidence in the future, even when working on unrelated code with different people."* and *"I withdraw myself and become less motivated and less enthusiastic about the work."*

Some participants noted the potential impact of inconsiderate feedback on diversity and inclusion saying, *"I think that unfortunately this is one of the things that turns away minority groups from tech".* Another participant said, *"I use a non-gendered alternate account for open source contributions as it's helped me getting things through, unlike my main account which clearly IDs me as a woman and gets more snark."*

Answer to RQ1: In the vignette questions, destructive feedback was perceived to be less appropriate, less valid, and less likely to help improve the code quality compared to constructive criticism. Compared to men, women were more likely to perceive destructive criticism as inappropriate. Open-ended comments from participants demonstrate that there is disagreement between software developers on how code review comments should be written, with some advocating for polite comments while others saying that direct feedback that may appear harsh or inconsiderate is easier to parse and act on.

4.2 Frequency of Destructive Criticism (RQ2)

Participants were asked both how often they gave and received destructive criticism during software code review. To enable more in-depth analysis, these questions distinguished between nonspecific negative feedback and inconsiderate negative feedback, which are the two main aspects of destructive criticism [10].

4.2.1 Giving destructive feedback. Most respondents (86 of 88) give code review feedback at least once a year. Very few respondents believe they have given inconsiderate feedback in the past year (only 1, who reports to give inconsiderate feedback quarterly). More respondents reported that they have given nonspecific negative feedback, with 23 of the 86 (27%) respondents who give feedback at least once a year reporting to have given nonspecific feedback at least yearly.

Gender and other demographic differences: For giving nonspecific feedback, there were no statistically significant differences across genders. More experience was associated with giving less nonspecific feedback, while having contributed to one or more OSS projects was associated with giving more nonspecific feedback (see Table 8). Unsurprisingly, the frequency of giving any code review feedback was associated with the frequency of giving nonspecific feedback, with those who give code review feedback more regularly also reporting to give nonspecific feedback more regularly. Given only one participant reported giving inconsiderate feedback, we cannot examine differences across demographics for giving inconsiderate feedback.

4.2.2 Receiving destructive criticism. More respondents reported receiving both inconsiderate and nonspecific negative feedback compared to those who reported giving such feedback. There were 87 respondents who reported receiving any code review feedback at least once a year. Of those, 19 (22%) reported receiving inconsiderate feedback and 48 (55%) reported receiving nonspecific negative feedback at least once a year.

Gender and other demographic differences: Non-binary participants reported receiving more inconsiderate feedback than other participants (see Table 8). However, given the small sample size for non-binary participants, we do not know if this is representative. There were no other statistically significant differences across gender or any other demographic variables for frequency of receiving nonspecific or inconsiderate feedback. However, the frequency of receiving any code

	Give nonspec.			Get nonspec.			Get inconsiderate		
	Value		SE	Value		SE	Value		SE
gender (non-binary)	0.94		1.26	-0.15		1.60	2.52	*	1.27
gender (woman)	-0.87		0.64	0.14		0.49	0.62		0.55
experience in years	-0.13	*	0.06	-0.05		0.04	-0.06		0.05
self-efficacy score	-0.03		0.75	0.00		0.52	0.41		0.68
education (undergraduate)	-0.56		1.45	1.35		0.92	-0.06		1.38
education (postgraduate)	-0.49		1.42	1.52		0.98	0.31		1.41
has contributed to OSS	1.27	*	0.65	0.32		0.48	0.97		0.56
competence (mod. high)	-0.83		0.85	0.15		0.62	-0.09		0.84
competence (high)	1.21		1.07	1.17		0.80	1.52		0.89
is a student	-0.53		1.43	0.88		0.84	0.16		1.37
give/get any feedback (yrly)	-1.48	***	0.00	11.56	***	1.06	-10.73	***	0.00
give/get any feedback (qtrly)	19.67	***	1.45	-1.04	***	0.00	-5.07	***	0.00
give/get any feedback (mthly)	0.41	***	0.00	12.81	***	0.75	9.48	***	1.21
give/get any feedback (wkly)	19.64	***	1.12	13.06	***	0.65	10.44	***	1.15

Table 8. Ordinal regression models for frequency of giving/receiving destructive criticism. (* p<0.05, ** p<0.01, *** p<0.001)

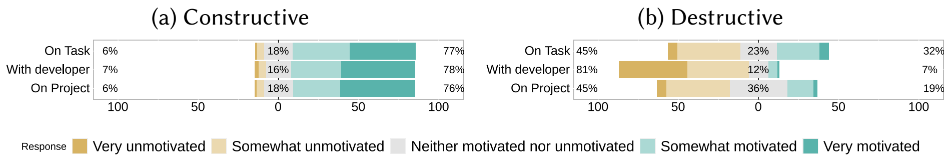

Fig. 3. Reactions to the statements "How motivated do you think you would be to: Continue working on this task?", "Continue working with this developer?", and "Continue working on this project?" in regards to the feedback given in the vignette questions, either constructive or destructive.

review feedback was associated with the frequency of receiving destructive feedback in both the nonspecific and inconsiderate regression models, with those receiving more code review feedback also reporting to receive more of both types of destructive feedback.

Answer to RQ2: Respondents report receiving more destructive criticism than giving such criticism. Of respondents who receive code review comments, 22% of respondents report receiving inconsiderate negative feedback at least once a year and 55% report receiving nonspecific negative feedback at least once a year. There were no differences in likelihood of receiving destructive feedback across genders or any other demographics.

4.3 Impact of Destructive Criticism (RQ3)

Through the two vignette questions, we examined the impact of destructive criticism (compared to constructive criticism) on the participant's perceived motivation to continue working, mood, and written responses to the feedback.

4.3.1 Impact on Motivation. Using vignette questions, we investigated the impact the destructive code review comments had on reported motivation to continue working on the task, to continue

	on the task		with the developer		on the project	
	Value	SE	Value	SE	Value	SE
gender (non-binary)	0.22	1.14	0.14	1.37	-0.80	1.65
gender (woman)	-0.80	0.46	**-1.24** **	0.47	-0.42	0.45
experience in years	-0.01	0.04	0.01	0.04	0.02	0.04
self-efficacy score	0.08	0.47	-0.16	0.51	0.59	0.51
education (undergraduate)	0.29	0.76	0.95	0.87	0.58	0.78
education (postgraduate)	0.77	0.76	1.26	0.84	0.85	0.76
has contributed to OSS	-0.10	0.45	-0.41	0.47	-0.24	0.45
competence (mod. high)	0.44	0.54	0.43	0.58	-0.06	0.55
competence (high)	0.36	0.73	-0.40	0.79	-0.95	0.76
is a student	-0.38	0.68	0.48	0.72	-1.13	0.71

Table 9. Ordinal regression models for responses related to motivation to continue working when destructive criticism received in vignette questions.
(* p<0.05, ** p<0.01, *** p<0.001)

working with the developer, and to continue working on the project as a whole. In general, all participants reported to be less motivated in all three categories when the vignette included destructive feedback (compared to constructive feedback). This trend can be seen on the Likert scale graphs in Figure 3. These differences were all statistically significant with a large effect. The average Likert value for motivation to continue working on the task was -0.13 for destructive and 1.11 for constructive (Mann-Whitney U=1573, p<0.001; Cohen's D=-1.25). For motivation to continue working with the developer, average responses were -1.16 for destructive criticism and 1.14 for constructive criticism (Mann–Whitney U=555.5, p<0.001; Cohen's D=-2.39). For motivation to continue working on the project, the average response was -0.30 for destructive criticism, while it was 1.16 for constructive criticism (Mann-Whitney U=1118.5, p<0.001; Cohen's D=-1.58). The largest effect size is seen in the motivation to continue working with the developer who gave the negative feedback. In an open-ended response, one participant mentioned, *"I withdraw myself and become less motivated and less enthusiastic about the work"* in reference to receiving destructive criticism.

Gender and other demographic differences: The results of the ordinal regression models (see Table 9) show that women, compared to men, were less motivated to continue working with the developer after receiving destructive criticism. There are no other statistically significant demographic differences.

4.3.2 Impact on Mood. Respondents were also asked how the feedback would make them feel after receiving the constructive and destructive criticism in the vignette questions. Table 10 shows that positive moods (*great*, *at ease*, *good*, *composed*) were reported with a much higher prevalence when respondents received constructive feedback compared to destructive feedback. Conversely, negative moods (*bad*, *tense*, *nervous*, *uncomfortable*) were reported more often when respondents received destructive feedback. Negative moods were reported by 18 out of 89 (20.2%) participants when the feedback was constructive, compared to 84 out of 89 (94.3%) participants when given destructive feedback. This difference is statistically significant (χ^2=100.02, p<0.001). This indicates that receiving destructive feedback has a negative effect on developers' moods. This is in line with the tone of the participants' written responses to the criticism received, where negative tones were used only in responses to destructive criticism (see Section 4.3.4). Gender and other demographic

	Constructive		**Destructive**	
	Count	%	Count	%
Great	16	17.20%	0	0.00%
At ease	30	32.26%	5	5.38%
Good	38	40.86%	3	3.23%
Composed	41	44.09%	5	5.38%
Bad	5	5.38%	50	53.76%
Tense	5	5.38%	52	55.91%
Nervous	7	7.53%	28	30.11%
Uncomfortable	7	7.53%	59	63.44%

Table 10. Participants' perceived mood after receiving constructive and destructive criticism.

differences were not examined given that nearly all participants reported negative moods after receiving destructive criticism.

4.3.3 Written Response to Feedback. Respondents were also given an option to provide a written response to the feedback using an open-ended question. They were instructed not to write anything if they would not respond to the code review feedback. There was no difference in response rates between the constructive and destructive criticism with 64 of 93 (69%) participants responding to constructive criticism and 66 of 93 (71%) responding to destructive criticism (χ^2=0.026, p=0.87).

We also did a deeper analysis of the content of responses. For participants who did provide a written response, responses were analyzed using Thematic Analysis [19] as described in section 3.5. Across all participants for the two vignette questions, 129 responses were provided to this open-ended question. Of these, 15 commented on the feedback or their opinion of the feedback rather than writing a written response to the feedback (e.g. "*They were unnecessarily rude and could have just pointed out it wasn't necessary and the reasons for it.*"). These were excluded from the analysis, leaving 114 written responses (61 in response to constructive criticism and 62 in response to destructive criticism). From these, we identified 5 types of responses. While we did not perform any statistical analysis of demographic differences across these response types, given the small sample sizes for each response type, we note any demographic trends for each response type if relevant.

- *Ask for more information.* 50 of 114 (43.9%) responses requested more information on the feedback. For example, *"could you add some explanations on the comments?"* This type of response was more common in response to destructive criticism (compared to constructive criticism) with 41 of the 50 responses of this type being in response to destructive criticism. Most of these responses (34 of 50) were from participants with less experience (<=5 years).
- *Agree with the criticism.* 50 of 114 (43.9%) responses accepted the code review comment as valid. For example, *"I guess we don't need a null check, will take it out."* This type of response was more common for constructive criticism (compared to destructive criticism) with 42 of the 50 responses of this type being in response to constructive criticism. When the feedback was destructive, most of the responses of this type (7 of 8) came from participants with less experience (<=5 years).
- *Disagree with the criticism on a technical level.* 13 of 114 (11.4%) responses rejected the code review comment as technically invalid. For example, *"The 'instanceOf' will only be triggered if there is no null pointer present thus this is a valid way of checking."* This response occurred

at similar rates for both constructive and destructive criticism. Most responses of this type (10 of 13) came from men.

- *Disagree with the manner of discourse in the criticism.* 8 of 114 (7%) responses stated disapproval on the way the code review comment was written. For example, *"If you disagree with my code you're welcome to provide constructive feedback in accordance with the code of conduct."* This type of response was only made in response to destructive criticism. Most of these responses (7 of 8) came from participants with more experience (> 5 years).
- *Acknowledge comment reception.* 2 of 114 (1.8%) responses simply confirmed the reception of the code review comment by replying similar to *"Will review and get in touch if any discussion is required. Thanks for the feedback."* This type of response was only given in response to constructive criticism.

It should be noted that some responses were categorized under multiple themes. For example, one participant wrote *"Null check is a good practice. Why do you think, null check should not be here?"* This comment was categorized as both *Disagree with the feedback at a technical level* and *Ask for more information.*

4.3.4 Written Response Tone. During the thematic analysis, we also categorized the tone of the written responses. Each response was categorized under a single tone. We found six distinct tones evident in the written responses:

- *Neutral.* 58 of 114 (50.9%) of responses were neutral in tone. For example, *"Could you please elaborate? Do you mean this logic should be simpler or to be extracted to a separate method?"* Neutral responses were more common in response to destructive criticism (43 of 58 made in response to destructive criticism).
- *Gratitude.* 42 of 114 (36.8%) of responses conveyed a thankfulness for the feedback. For example, *"Thanks for your feedback, xxx. It makes the code cleaner and more readable. I will update the pull request."* This tone was used more commonly in response to constructive criticism (39 of 42 were in response to constructive criticism). The majority of responses showing gratitude (27 of 42) were made by participants with less experience (<=5 years).
- *Apology.* 5 of 114 (4.4%) of responses apologized for the problem in the code. For example, *"Oh okay sorry, I'll take it out."* This tone was only used in response to destructive criticism. Four of the five responses that were apologetic came from women. All responses that were apologetic came from participants with <= 5 years of experience.
- *Praise.* 5 of 114 (4.4%) conveyed an admiration for the problem being detected by the code reviewer. For example, *"Good catch, removed"*. This tone was used more commonly in response to constructive criticism (4 of 5 were in response to constructive criticism). Four of the five responses that conveyed praise were made by men.
- *Sarcasm.* 2 of 114 (1.8%) of the responses were sarcastic. For example, *"Please, enlighten me with the improvements that could be made to this code snippet and an explanation on why you think this manner of solving the bug is 'obtuse'."* This tone was only used in response to destructive criticism.
- *Anger.* 2 of 114 (1.8%) of the responses expressed annoyance or hostility towards the feedback. For example, *"what the heck?? wanna explain why?"* This tone was only used in response to destructive criticism.

Answer to RQ3: Compared to constructive criticism, destructive criticism negatively impacts motivation to continue working and mood of the participants. Women were less motivated to continue working with the developer who gave the feedback.

Participants responded to destructive criticism to ask for more information and disagree with the manner of discourse. For constructive criticism, participants most commonly responded to agree with the criticism. The majority of responses were neutral in tone. However, participants also showed anger, sarcasm, and apology in response to destructive criticism. For constructive criticism, gratitude and praise were more common.

5 DISCUSSION

In this section, we discuss the threats to validity and implications of this research.

5.1 Threats to Validity

The validity and reliability of the questionnaire itself is a potential threat to validity, since the questionnaire was designed by the research team. We used Cronbach's Alpha [27] to measure the internal consistency reliability of our questionnaire (excluding the open-ended response questions). The Cronbach's Alpha score for all items was 0.702, indicating good overall internal reliability. To improve the validity of the questionnaire, we performed a pilot study to ensure respondents interpreted our questions how we intended. Future work can replicate our study to further validate the reliability and validity of the questionnaire.

Related to the reliability and validity of the questionnaire, we used vignettes to examine the impact of constructive and destructive criticism. This allowed us to isolate the effect of constructive versus destructive criticism since we were able to present two versions of the same scenario-based question where all conditions (e.g. same scenario, same code snippet) are fixed except for the type of criticism received. This enabled us to clearly examine the differences between the two types of criticism. However, this also introduces a threat in that we do not know if our results will hold in more realistic settings. To mitigate this bias, we used real code snippets and code review comments from open source software projects to make the vignettes as close as possible to reality. The scenarios used were also designed to be as concrete and detailed as possible to minimize the bias [1]. Further, vignette-based questions have been shown to produce more reliable opinions than more abstract opinion-based questions [1], and they provide the benefit of not having to subject participants to the negative treatment of unexpected destructive criticism. Despite these mitigation techniques, the threat remains that participants may respond differently in a realistic scenario, and we encourage future field research of destructive criticism in software code review.

The vignettes also introduce a threat that we are asking participants to imagine how they would react to feedback rather than studying their actual reactions. Research has found that people are generally good at assessing their own intentions in hypothetical situations, but this may introduce affective forecasting biases where participants overestimate the intensity of their reactions in the hypothetical situations [82]. To mitigate this bias, we compared responses between the two types of criticism. We saw significant differences in anticipated reactions between the two types of criticism, with significantly more negative reactions being reported for destructive criticism. Even so, it is possible that the intensity of the reported reactions is overestimated. Further, in the vignette-style questions, participants were asked to consider how receiving feedback in a hypothetical situation would make them feel. We did not ask participants about their current mood before reading the vignettes, which could impact their perceived moods in the hypothetical situations. However, this threat also exists in real world settings where the mood of software developers can impact how they react to feedback. Future work could investigate specifically how someone's mood impacts their reactions to destructive criticism.

The order in which each type of feedback is presented could also introduce a bias (question order bias). To mitigate this, the order in which participants received each type of feedback was randomized, with roughly half of the participants receiving the destructive criticism first and the remainder receiving constructive criticism first. Nonetheless, question ordering may have affected responses for questions where we did not randomize orderings.

Another potential threat is the social desirability bias, where participants answer questions based on what they believe will make them appear more favorable. This was mitigated by having an anonymous questionnaire, where participant identity was unknown to the researchers. Despite this, a threat still exists that responses were impacted by social desirability bias.

Convenience sampling, used in this study to recruit participants, is a non-probabilistic sampling method and a possible source of bias [38]. The target population of this study were software practitioners. We recruited participants by advertising the questionnaire on social media such as Facebook and Twitter and online software developer communities (e.g. local software meetup groups). Additionally, all respondents who completed the survey were self-selected, which introduces a self-selection bias. We received an equal number of responses from men and women and three responses (3.2%) were from non-binary people. This gender breakdown is not representative of the software industry as a whole, but it did enable us to examine gender differences, which was the goal of this research. Nevertheless, the sample may not be representative of all software developers, and the code review habits and perceptions of the participants may not generalize to all software practitioners. To mitigate this bias, we advertised the questionnaire invitation on a variety of channels to encourage a diverse set of participants. Also, when advertising the questionnaire, the goal was described as understanding the impacts of feedback in software code review, without revealing the researchers focus on destructive criticism, to aim for a broader set of participants not limited to only those who care about the impact of destructive criticism. However, our results might not generalize outside the sample. Replications of this study in the future can further validate our findings.

Another threat to validity is the potential exclusion of some confounding variables. There are many variables that could potentially impact how software developers perceive destructive criticism. While we made an effort to include potential confounding factors, such as competence in software development and level of education, it is possible that there are other confounding variables that we have missed. For example, the *existence of a code of conduct*, which describes expectations around code review comments, could impact how participants perceive criticism received during code review. Codes of conduct are designed to prevent destructive criticism, and they typically describe mechanisms for reporting violations [63, 78]. Therefore, it is possible that a code of conduct could reduce the impact of destructive criticism since the recipient would be aware that such behavior should not be tolerated. Another potential confounding variable is the participants' *sensitivity to criticism*. Prior research found that people have varying levels of sensitivity to criticism and that sensitivity to criticism can affect motivation [6]. Differing levels of sensitivity of criticism could possibly explain some of the contrasting opinions on the appropriateness of destructive criticism identified in this study. Future studies can continue to investigate additional factors that impact perceptions of destructive criticism.

Finally, the statistical analysis performed on the survey responses was done using ordinal logistic regression, using linear variables. Therefore, non-linear effects could potentially exist between variables, which have not been discovered in this analysis. Due to the number of participants and the number of variables considered, it is not possible to identify potential non-linear effects with any statistical significance. Therefore, only linear models were used for the statistical analysis of responses. Future work can explore any potentially more complex non-linear effects between the variables studied with a larger sample size of participants.

5.2 Implications and Future Work

Opinions on appropriate software code review are varied. While participants at an abstract level agreed that destructive criticism was harmful, opinions were mixed on whether destructive criticism was acceptable if the outcome was improved code quality. In open-ended responses, some participants voiced that inconsiderate feedback was always unacceptable while others believed it was necessary at times. This split is also evident in the open source software community. While many projects have begun to adopt codes of conduct, there are vocal opponents to codes of conduct who believe that being expected to conform to certain standards is against the democratic principles of open source software and that source code quality will suffer without brutal honesty. Some have gone as far as starting an opposition "No Code of Conduct" movement, boycotting projects with codes of conduct[3]. Future work could investigate whether these differences in opinions can be attributed to differences in sensitivity to criticism [6]. If sensitivity to criticism is an important factor, this could have important implications for code of conduct design. In this case, it would be important to convey the importance of not giving destructive criticism to those with less sensitivity who might not understand the impacts of such criticism on those with greater sensitivity. It is also possible that such variations in perceptions around the acceptableness of destructive criticism also exist for workers in other domains. Thus, future work could study perceptions of destructive criticism and its relation to sensitivity to criticism more broadly.

In our study, significantly more participants reported receiving destructive criticism compared to those who reported giving destructive criticism. It is possible that responses were impacted by social desirability bias, and participants did not want to admit to giving destructive criticism to look better. It is also possible that it was difficult for some participants to admit to giving destructive criticism due to cognitive dissonance. Cognitive dissonance theory states that people have a desire to align their actions and their beliefs [40], so someone who believes that destructive criticism is wrong may not be able to admit, even to themselves, that they give such criticism. Thus, it is possible that the amount of destructive criticism given was under-reported. Of course, this disparity could also be due to the fact that only a small number of individuals give a majority of the destructive criticism. Future work can investigate through observational studies whether most destructive criticism originates from a small number of people. It would also be interesting to understand whether the number of people giving destructive criticism varies across domains.

Giving destructive criticism could be unintentional. Some participants noted that, even with good intentions, feedback can result in a negative reaction in the recipient and that it is difficult to always get it right when providing criticism. The consequences of inconsiderate feedback can be quite severe with respondents noting that it can shake their confidence, lower their motivation and enthusiasm, and cause conflict in teams. While improved guidelines could be useful for some projects, many guidelines already exist that encourage reviewers to write both specific and considerate code review feedback (e.g. Google's code review guidelines[4]). However, despite such guidelines, participants in this study still report receiving both nonspecific and inconsiderate code review feedback. What is considered to be specific and considerate could also be subjective, making it difficult for code reviewers to know how their feedback will be received. Another possibility for the disparity between the number of participants reporting giving and the number reporting receiving destructive criticism could be a lack of awareness of how feedback is being interpreted by the recipient. This lack of awareness on how criticism will be received could potentially span across domains. Thus, these findings could be applicable more broadly and future work could investigate better ways to improve awareness more generally.

[3]https://github.com/domgetter/NCoC

[4]https://google.github.io/eng-practices/review/reviewer/comments.html

In the software industry, one potential solution would be additional tool support for code review comments. Tools that automatically detect toxic comments, such as the Sentiment Bot[5] and the Safe Space Bot[6] available on GitHub, could be used prevent toxic language in code review comments. Additionally, since code review feedback without explicitly toxic language can still be taken negatively by the receiver, code review tools could also integrate a mechanism to give feedback on code review comments. This would enable code reviewers to be aware of how their feedback is being received. Very few participants provided written feedback in response to the criticism they received in the questionnaire that disagreed with the manner of discourse in the destructive criticism. While some participants noted that they would directly address inconsiderate feedback with the giver, others said they would avoid confrontation and would not vocalize their concerns. One participant complained that it is currently not possible to know how code authors react to their feedback. *"I'm very aware of what comments can do and constantly try to not criticise in an unfriendly way, formulating everything as a question or similar. Also there is sadly never a conversation later on how the review was received, no feedback to me on whether it was appropriate or hurt the recipient. I'd like such feedback."* Tool designers could consider integrating this functionality to encourage code reviewers to better consider the impact of their feedback delivery. Given that some code authors do not feel comfortable calling out inappropriate criticism, this feature would need to be designed to enable a safe way to provide this feedback. This may be in contrast to the "everything is open" philosophy of open source software. Thus, future research should investigate how to best design this feature for it to be adopted by open source software teams. Similarly, tools could be devised for other domains to promote constructive feedback.

Given the transparency in many collaborative software development platforms [28], inconsiderate criticism can be even more problematic given its public nature. This was noted by a participant who said, *"It becomes a much bigger problem if the reviewer puts the person on the spot in front of other people (or shaming in the public channel) about the quality of code in the review - then things would start to get personal."* It would also be important to include in code review guidelines what type of feedback is inappropriate in a public venue.

Another possible reason destructive criticism could be given during software code review is external pressure or stress of the developers doing the code review. Modern software development methodologies have moved towards frequent or even continuous delivery [11, 12]. While more frequent releases bring many benefits, research has also found that they put pressure on software developers and cause stress [69]. Sarker et al. found that developers communications are both more negative and terser when developers are stressed [66]. Thus, it is possible that destructive criticism is an unintentional consequence of stress and pressure on software teams. The link between stress and destructive criticism should be further investigated by future research, and, if such a link is identified, additional support should be provided to help developers minimize stress to improve inclusion on software teams. The link between stress and destructive criticism would also be important in other domains.

Gender differences. Women were less likely than men to believe destructive criticism was appropriate, and they also reported they would be significantly less motivated to continue working with the developer after receiving destructive criticism. Previous studies on destructive feedback identified gender differences, but only in the case of some types of tasks [2, 10]. Compared to the tasks in these previous studies (e.g., clerical work and proofreading), writing software code involves more knowledge creation. Thus, our findings are important to understand gender differences in the effects of destructive criticism. Specifically, gender differences do exist when considering

[5]https://probot.github.io/apps/sentiment-bot
[6]https://github.com/charliegerard/safe-space

destructive criticism in software code review. This gender difference could be due to women being under-represented in the software industry and already feeling like they don't fit in as found by more recent studies [30, 68, 80]. Thus, receiving destructive, inconsiderate criticism could reinforce these feelings of lack of belonging. In their written responses to feedback, women apologized more when receiving destructive criticism and less commonly disagreed technically with the feedback, compared to men. This could be due to a tightrope bias, where women are perceived to have a more narrow band of socially acceptable behavior [50]. Yet, a similar number of men and women wrote responses to destructive criticism which disagreed with the discourse of the feedback. However, the number of respondents who vocalized their disagreement with destructive criticism in a written response to the feedback was rather small. Thus, we recommend future studies of destructive criticism to consider the gender balance of the domain of study when interpreting their results.

Outside of the destructive criticism literature, our results are also in line with findings from education research, particularly studies that examine girls learning how to code. Studies have found that girls' motivations are influenced by both positive and negative emotions [23, 44]. Since our results found that destructive criticism was associated with negative moods while constructive criticism was associated with positive moods, this may help to explain why women in our study were less motivated to continue working with the developer after receiving destructive criticism.

Non-binary participants. Due to the low number of non-binary participants, we do not know if their responses are representative. We discuss their responses here at a high-level, but caution against drawing conclusions from the responses. We can see from Table 6 that the three non-binary participants reported less agreement with the statements that destructive criticism received in the vignette questions would help to improve their code and that it was appropriate. The coefficients for both of these models are highest for the non-binary gender independent variable, showing their disagreement with these statements was more extreme compared to other participants. Table 7 shows that non-binary participants tended to slightly agree that they do not mind receiving inconsiderate feedback as long as the feedback would help to improve code quality. In terms of frequency of destructive feedback, 2 of the 3 non-binary respondents reported receiving inconsiderate negative feedback in the past year, but none report giving inconsiderate feedback. The non-binary participants reported receiving more inconsiderate feedback (see Table 8). Non-binary participants did report giving nonspecific feedback in the past year. Only one non-binary participant reported to receive nonspecific feedback.

Table 9 shows non-binary participants also reported being less motivated to continue working on a project after receiving destructive criticism. However, they reported being still slightly motivated to continue working on the task and with the developer. This is in contrast to women, who were most unmotivated to continue working with the developer who gave the destructive criticism. The only open-ended survey response from a non-binary participant indicated that destructive criticism impacts an individual's confidence in the future, even if working on different code with different people. Thus, there could be important implications for retention of non-binary software developers that warrants further investigation. Recent calls have been made for software engineering research to consider the full gender spectrum [41, 70]. We suggest using purposive or respondent-driven sampling techniques in future research studies to obtain responses across the gender spectrum [8]. Similarly, future studies of destructive criticism in other domains should consider the full gender spectrum.

6 CONCLUSION

Destructive criticism is known to cause negative effects on its receiver [10, 13]. This has been observed in many contexts outside software engineering. However, this is the first study to apply the destructive criticism lens to software code review. In a survey of 93 software practitioners, we found that destructive criticism had a negative impact on participants' moods and motivation to continue working. We found that the impact of destructive criticism was greater for women, who reported to be more unmotivated to continue working after receiving destructive criticism. Women were also more likely to believe destructive criticism was inappropriate.

The findings presented in this paper show that there are conflicting opinions on destructive criticism. Yet, given that the impacts of destructive criticism can be large, we believe it is important to work towards shifting the culture of code review in the software industry. We have discussed potential future work that can move in this direction including the creation of mechanisms to allow code reviewers to become aware when they provide destructive criticism, guidelines on when code review feedback on public channels is inappropriate, and further research on the association between stress and time pressure and destructive criticism.

ACKNOWLEDGEMENT

We thank the study participants for their time. This work was supported by a Google Faculty Research Award (KB).

REFERENCES

[1] Cheryl S Alexander and Henry Jay Becker. 1978. The use of vignettes in survey research. Public opinion quarterly 42, 1 (1978), 93–104.

[2] Kelly M Allred and Dianne L Chambless. 2018. Racial differences in attributions, perceived criticism, and upset: A study with Black and White community participants. Behavior therapy 49, 2 (2018), 273–285.

[3] Anonymous. 2014. Leaving Toxic Open Source Communities. https://modelviewculture.com/pieces/leaving-toxic-open-source-communities

[4] Anonymous. 2016. I worked on Facebook's Trending team – the most toxic work experience of my life. https://www.theguardian.com/technology/2016/may/17/facebook-trending-news-team-curators-toxic-work-environment

[5] Neal M Ashkanasy, Charmine EJ Härtel, and Wilfred J Zerbe. 2000. Emotions in the workplace: Research, theory, and practice. Quorum Books/Greenwood Publishing Group.

[6] Gordon D. Atlas. 1994. Sensitivity to Criticism: A New Measure of Responses to Everyday Criticisms. Journal of Psychoeducational Assessment 12, 3 (1994), 241–253.

[7] Alberto Bacchelli and Christian Bird. 2013. Expectations, outcomes, and challenges of modern code review. In in Proc. 2013 35th International Conference on Software Engineering (ICSE). 712–721.

[8] Sebastian Baltes and Paul Ralph. 2020. Sampling in software engineering research: A critical review and guidelines. arXiv preprint arXiv:2002.07764 (2020).

[9] Lecia Barker, Cynthia Mancha, and Catherine Ashcraft. 2014. What is the impact of gender diversity on technology business performance. Research Summary [Internet] (2014).

[10] Robert A Baron. 1988. Negative effects of destructive criticism: Impact on conflict, self-efficacy, and task performance. Journal of Applied Psychology 73, 2 (May 1988), 199.

[11] Len Bass, Ingo Weber, and Liming Zhu. 2015. DevOps: A software architect's perspective. Addison-Wesley Professional.

[12] Kent Beck, James Grenning, Robert C. Martin, Mike Beedle, Jim Highsmith, Steve Mellor, Arie van Bennekum, Andrew Hunt, Ken Schwaber, Alistair Cockburn, Ron Jeffries, Jeff Sutherland, Ward Cunningham, Jon Kern, Dave Thomas, Martin Fowler, and Brian Marick. 2001. Manifesto for Agile Software Development.

[13] Frank D Belschak and Deanne N Den Hartog. 2009. Consequences of positive and negative feedback: The impact on emotions and extra-role behaviors. Applied Psychology 58, 2 (Apr 2009), 274–303.

[14] Amel Bennaceur, Ampaeli Cano, Lilia Georgieva, Mariam Kiran, Maria Salama, and Poonam Yadav. 2018. Issues in Gender Diversity and Equality in the UK. In in Proc.1st International Workshop on Gender Equality in Software Engineering. 5–9.

[15] Kelly Blincoe, Giuseppe Valetto, and Daniela Damian. 2015. Facilitating coordination between software developers: A study and techniques for timely and efficient recommendations. IEEE Transactions on Software Engineering 41, 10 (2015), 969–985.

[16] Amiangshu Bosu and Jeffrey C Carver. 2013. Impact of peer code review on peer impression formation: A survey. In 2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. 133–142.

[17] Amiangshu Bosu, Jeffrey C Carver, Christian Bird, Jonathan Orbeck, and Christopher Chockley. 2016. Process aspects and social dynamics of contemporary code review: Insights from open source development and industrial practice at microsoft. IEEE Transactions on Software Engineering 43, 1 (Jun 2016), 56–75.

[18] Dave Bouckenooghe, Usman Raja, and Arif Nazir Butt. 2013. Combined effects of positive and negative affectivity and job satisfaction on job performance and turnover intentions. The Journal of psychology 147, 2 (2013), 105–123.

[19] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. Qualitative research in psychology 3, 2 (Jan 2006), 77–101.

[20] Margaret Burnett, Anicia Peters, Charles Hill, and Noha Elarief. 2016. Finding gender-inclusiveness software issues with GenderMag: a field investigation. In in Proc. 2016 Conference on Human Factors in Computing Systems. 2586–2598.

[21] Gilad Chen, Stanley M Gully, and Dov Eden. 2001. Validation of a new general self-efficacy scale. Organizational research methods 4, 1 (Jan 2001), 62–83.

[22] Jithin Cheriyan, Bastin Tony Roy Savarimuthu, and Stephen Cranefield. 2020. Norm violation in online communities–A study of Stack Overflow comments. arXiv preprint arXiv:2004.05589 (Apr 2020).

[23] Jacqui Chetty and Glenda Barlow-Jones. 2018. Coding for girls: dismissing the boys club myth. In the 18th International Conference on Information, Communication Technologies in Education (ICICTE 2018).

[24] Lee Anna Clark and David Watson. 1988. Mood and the mundane: Relations between daily life events and self-reported mood. Journal of personality and social psychology 54, 2 (1988), 296.

[25] Jacob Cohen. 2013. Statistical power analysis for the behavioral sciences. Academic press.

[26] Katy Cook. 2020. Culture & Environment. In The Psychology of Silicon Valley. Springer, 37–64.

[27] Lee J Cronbach. 1951. Coefficient alpha and the internal structure of tests. psychometrika 16, 3 (1951), 297–334.

[28] Laura Dabbish, Colleen Stuart, Jason Tsay, and Jim Herbsleb. 2012. Social coding in GitHub: transparency and collaboration in an open software repository. In Proceedings of the ACM 2012 conference on computer supported cooperative work. 1277–1286.

[29] Aniruddha Das. 2009. Sexual harassment at work in the United States. Archives of sexual behavior 38, 6 (2009), 909–921.

[30] Paul A David and Joseph S Shapiro. 2008. Community-based production of open-source software: What do we know about the developers who participate? Information Economics and Policy 20, 4 (Dec 2008), 364–398.

[31] Munmun De Choudhury and Scott Counts. 2013. Understanding affect in the workplace via social media. In Proceedings of the 2013 conference on Computer supported cooperative work. 303–316.

[32] Giuseppe Destefanis, Marco Ortu, Steve Counsell, Stephen Swift, Michele Marchesi, and Roberto Tonelli. 2016. Software development: do good manners matter? PeerJ Computer Science 2 (2016), e73.

[33] Nicolas Ducheneaut. 2005. Socialization in an open source software community: A socio-technical analysis. Computer Supported Cooperative Work (CSCW) 14, 4 (2005), 323–368.

[34] Michelle K Duffy, Daniel C Ganster, and Milan Pagon. 2002. Social undermining in the workplace. Academy of management Journal 45, 2 (2002), 331–351.

[35] Carolyn D Egelman, Emerson Murphy-Hill, Elizabeth Kammer, Margaret Morrow Hodges, Collin Green, Ciera Jaspan, and James Lin. 2020. Predicting developers' negative feelings about code review. In 2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE). 174–185.

[36] Staale Einarsen, Helge Hoel, and Guy Notelaers. 2009. Measuring exposure to bullying and harassment at work: Validity, factor structure and psychometric properties of the Negative Acts Questionnaire-Revised. Work & stress 23, 1 (2009), 24–44.

[37] Ikram El Asri, Noureddine Kerzazi, Gias Uddin, Foutse Khomh, and MA Janati Idrissi. 2019. An empirical study of sentiments in code reviews. Information and Software Technology 114 (2019), 37–54.

[38] Ilker Etikan, Sulaiman Abubakar Musa, and Rukayya Sunusi Alkassim. 2016. Comparison of convenience sampling and purposive sampling. American journal of theoretical and applied statistics 5, 1 (2016), 1–4.

[39] Michael Fagan. 1976. Design and code inspections to reduce errors in program development. Vol. 15. 182–2011.

[40] Leon Festinger. 1957. A theory of cognitive dissonance. Vol. 2. Stanford university press.

[41] Denae Ford, Reed Milewicz, and Alexander Serebrenik. 2019. How remote work can foster a more inclusive environment for transgender developers. In 2019 IEEE/ACM 2nd International Workshop on Gender Equality in Software Engineering (GE). IEEE, 9–12.

[42] Denae Ford and Chris Parnin. 2015. Exploring causes of frustration for software developers. In 2015 IEEE/ACM 8th International Workshop on Cooperative and Human Aspects of Software Engineering. IEEE, 115–116.

[43] Daviti Gachechiladze, Filippo Lanubile, Nicole Novielli, and Alexander Serebrenik. 2017. Anger and its direction in collaborative software development. In 2017 IEEE/ACM 39th International Conference on Software Engineering: New Ideas and Emerging Technologies Results Track (ICSE-NIER). IEEE, 11–14.

[44] Michail N Giannakos, Letizia Jaccheri, and Ioannis Leftheriotis. 2014. Happy girls engaging with technology: Assessing emotions and engagement related to programming activities. In International Conference on Learning and Collaboration Technologies. Springer, 398–409.

[45] Daniel Graziotin, Fabian Fagerholm, Xiaofeng Wang, and Pekka Abrahamsson. 2017. Unhappy developers: Bad for themselves, bad for process, and bad for software product. In 2017 IEEE/ACM 39th International Conference on

Software Engineering Companion (ICSE-C). 362–364.
[46] Daniel Graziotin, Fabian Fagerholm, Xiaofeng Wang, and Pekka Abrahamsson. 2018. What happens when software developers are (un) happy. Journal of Systems and Software 140 (2018), 32–47.
[47] Sanuri Dananja Gunawardena, Peter Devine, Isabelle Beaumont, Lola Garden, Emerson Murphy-Hill, and Kelly Blincoe. 2022. Replication Package for Destructive Criticism in Software Code Review Impacts Inclusion. https://doi.org/10.6084/m9.figshare.14378954.v2
[48] Emitza Guzman, David Azócar, and Yang Li. 2014. Sentiment analysis of commit comments in GitHub: an empirical study. In in Proc.11th working conference on mining software repositories. 352–355.
[49] Daniel Ilgen and Cori Davis. 2000. Bearing bad news: Reactions to negative performance feedback. Applied Psychology 49, 3 (Jul 2000), 550–565.
[50] Nasif Imtiaz, Justin Middleton, Joymallya Chakraborty, Neill Robson, Gina Bai, and Emerson Murphy-Hill. 2019. Investigating the effects of gender bias on GitHub. In in Proc. 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE). 700–711.
[51] Nathan Ingraham. 2016. Report: Apple is a sexist, toxic work environment. https://www.engadget.com/2016-09-15-apple-sexist-workplace-reports.html
[52] Md Rakibul Islam and Minhaz F Zibran. 2017. Leveraging automated sentiment analysis in software engineering. In 2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR). IEEE, 203–214.
[53] Avraham N Kluger, Shai Lewinsohn, and John R Aiello. 1994. The influence of feedback on mood: Linear effects on pleasantness and curvilinear effects on arousal. Organizational Behavior and Human Decision Processes 60, 2 (1994), 276–299.
[54] Richard S Lazarus and Richard S Lazarus. 1991. Emotion and adaptation. Oxford University Press on Demand.
[55] Dawn Nafus. 2012. 'Patches don't have gender': What is not open in open source software. New Media & Society 14, 4 (Jun 2012), 669–683.
[56] Marco Ortu, Bram Adams, Giuseppe Destefanis, Parastou Tourani, Michele Marchesi, and Roberto Tonelli. 2015. Are bullies more productive? Empirical study of affectiveness vs. issue fixing time. In in Proc.12th Working Conference on Mining Software Repositories. 303–313.
[57] Christian R Østergaard, Bram Timmermans, and Kari Kristinsson. 2011. Does a different view create something new? The effect of employee diversity on innovation. Research Policy 40, 3 (Apr 2011), 500–509.
[58] Rajshakhar Paul, Amiangshu Bosu, and Kazi Zakia Sultana. 2019. Expressions of sentiments during code reviews: Male vs. female. In 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER). 26–37.
[59] Philip M Podsakoff and Jiing-Lih Farh. 1989. Effects of feedback sign and credibility on goal setting and task performance. Organizational behavior and human decision processes 44, 1 (1989), 45–67.
[60] Erik Pulkstenis and Timothy J Robinson. 2004. Goodness-of-fit tests for ordinal response regression models. Statistics in medicine 23, 6 (2004), 999–1014.
[61] Naveen Raman, Minxuan Cao, Yulia Tsvetkov, Christian Kästner, and Bogdan Vasilescu. 2020. Stress and burnout in open source: toward finding, understanding, and mitigating unhealthy interactions. In Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: New Ideas and Emerging Results. 57–60.
[62] Jana L Raver, Jaclyn M Jensen, Junghyun Lee, and Jane O'Reilly. 2012. Destructive criticism revisited: Appraisals, task outcomes, and the moderating role of competitiveness. Applied Psychology 61, 2 (2012), 177–203.
[63] Neill Robson. 2018. Diversity and decorum in open source communities. In Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. 986–987.
[64] Caitlin Sadowski, Emma Söderberg, Luke Church, Michal Sipko, and Alberto Bacchelli. 2018. Modern code review: a case study at google. In Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice. 181–190.
[65] S. Sankarram. 2018. Unlearning Toxic Behaviors in a Code Review Culture. https://medium.com/@sandya.sankarram/unlearning-toxic-behaviors-in-a-code-review-culture-b7c295452a3c
[66] Farhana Sarker, Bogdan Vasilescu, Kelly Blincoe, and Vladimir Filkov. 2019. Socio-technical work-rate increase associates with changes in work patterns in online projects. In 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE). 936–947.
[67] Jaydeb Sarker, Asif Kamal Turzo, and Amiangshu Bosu. 2020. A Benchmark Study of the Contemporary Toxicity Detectors on Software Engineering Interactions. arXiv preprint arXiv:2009.09331 (2020).
[68] Alexander Serenko and Ofir Turel. 2021. Why are women underrepresented in the American IT industry? The role of explicit and implicit gender identities. Journal of the Association for Information Systems 22, 1 (2021), 8.
[69] Mojtaba Shahin, Muhammad Ali Babar, and Liming Zhu. 2017. Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices. IEEE Access 5 (2017), 3909–3943.

[70] Karina Kohl Silveira, Soraia Musse, Isabel H Manssour, Renata Vieira, and Rafael Prikladnicki. 2019. Confidence in programming skills: gender insights from StackOverflow developers survey. In 2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion). IEEE, 234–235.

[71] Vinayak Sinha, Alina Lazar, and Bonita Sharif. 2016. Analyzing developer sentiment in commit logs. In Proceedings of the 13th international conference on mining software repositories. 520–523.

[72] Megan Squire and Rebecca Gazda. 2015. FLOSS as a Source for Profanity and Insults: Collecting the Data. In 2015 48th Hawaii International Conference on System Sciences. IEEE, 5290–5298.

[73] R Steyer, P Schwenkmezger, P Notz, and M Eid. 1997. Multidimensional Mood Questionnaire (MDMQ). Göttingen: Hogrefe (1997), 33.

[74] Margaret-Anne Storey, Neil A Ernst, Courtney Williams, and Eirini Kalliamvakou. 2020. The who, what, how of software engineering research: a socio-technical framework. Empirical Software Engineering 25, 5 (2020), 4097–4129.

[75] Josh Terrell, Andrew Kofink, Justin Middleton, Clarissa Rainear, Emerson Murphy-Hill, Chris Parnin, and Jon Stallings. 2017. Gender differences and bias in open source: Pull request acceptance of women versus men. PeerJ Computer Science 3 (May 2017), e111.

[76] Mike Thelwall, David Wilkinson, and Sukhvinder Uppal. 2010. Data mining emotion in social network communication: Gender differences in MySpace. Journal of the American Society for Information Science and Technology 61, 1 (Jan 2010), 190–199.

[77] Gregory K Tortoriello and William Hart. 2019. Trait interpersonal vulnerability attenuates beneficial effects of constructive criticism on failure responses. British Journal of Psychology 110, 3 (Aug 2019), 594–613.

[78] Parastou Tourani, Bram Adams, and Alexander Serebrenik. 2017. Code of conduct in open source projects. In 2017 IEEE 24th international conference on software analysis, evolution and reengineering (SANER). IEEE, 24–33.

[79] Parastou Tourani, Yujuan Jiang, and Bram Adams. 2014. Monitoring sentiment in open source mailing lists: exploratory study on the apache ecosystem. In CASCON, Vol. 14. 34–44.

[80] Bogdan Vasilescu, Daryl Posnett, Baishakhi Ray, Mark GJ van den Brand, Alexander Serebrenik, Premkumar Devanbu, and Vladimir Filkov. 2015. Gender and tenure diversity in GitHub teams. In in Proc. 33rd annual ACM conference on human factors in computing systems. 3789–3798.

[81] W. N. Venables and B. D. Ripley. 2002. Modern Applied Statistics with S (fourth ed.). Springer, New York. https://www.stats.ox.ac.uk/pub/MASS4/ ISBN 0-387-95457-0.

[82] Timothy D Wilson and Daniel T Gilbert. 2003. Affective forecasting. (2003).

[83] Michal R Wrobel. 2013. Emotions in the software development process. In 2013 6th International Conference on Human System Interactions (HSI). IEEE, 518–523.

[84] Michal R Wrobel, Olga Springer, and Kelly Blincoe. 2019. Perceptions of Gender Diversity's impact on mood in software development teams. IEEE Software 36, 5 (2019), 51–56.