Toxic Talk, Lasting Harm: How Words Affect Productivity and Retention

Jinci Liu *

February 27, 2025

Preliminary and Incomplete Draft

Abstract

Workplace toxicity is widespread, but its economic consequences and the reasons behind toxic feedback remain understudied. I analyze 230 million comments to detect toxic feedback from code reviewers and examine economic consequences for developers and firms. This new approach provides a systematic way to identify toxicity. I find two main results. First, toxic comments flow disproportionately from senior to junior developers, reflecting hierarchical feedback patterns. Second, focusing on teams with random reviewer assignments, I construct reviewer toxicity IV and find that exposure to toxic comments reduces developer output by 44.8% over six months. Notably, these substantial effects appear to be specific to toxic comments and do not appear in either non-toxic negative or positive feedback.

*IIES, Stockholm University. Email: jinci.liu@iies.su.se.

I would like to thank Arash Nekoei and Mitch Downey for their continual guidance, encouragement and support. I am extremely grateful to Huen Tat Au-Yeung, Sydnee Caldwell, Stefano DellaVigna, Yu Ching Lam, Danielle Li, Patrizia Massner, Chloé Nibourel, Jens Oehlen, David Schönholzer, Mateusz Stalinski, David Strömberg, Horng Chern Wong, Yimei Zou and seminar participants at IIES Stockholm University and UC Berkeley. I thank Jakob Beuschlein for his valuable feedback on Judge IV. I acknowledge support from the Google Cloud Research Credits program (317889488, 216835289 and 381934723) and I thank the Mannerfelt Foundation and Institute for Evaluation of Labour Market and Education Policy (IFAU) for their generous financial support. All errors are my own.

1 Introduction

Workplace toxicity, unfortunately, is an issue that has become rather pervasive today. According to the 2024 Work in America Survey, 61% of workers report experiencing diminished psychological safety. Furthermore, an alarming 15% label their workplaces as somewhat or very toxic, citing behaviors such as intimidation, infighting, and demeaning communication (American Psychological Association, 2024). Studies conducted with victims of such toxic environments consistently demonstrate that exposure to these detrimental conditions is linked with an increase in mental issues such as depression, burnout and absenteeism (Zsila et al., 2022; McTernan et al., 2013; Nielsen and Einarsen, 2012; Fattori et al., 2015).

Despite the mental health costs of workplace toxicity, it remains an open question what its economic costs are and why toxic behavior exists. Indeed, the impact of toxic comments on workers' productivity, particularly in the short term, is ex-ante ambiguous. On one hand, toxic comments can elevate stress levels and diminish workers' well-being, ultimately lowering productivity. On the other hand, such comments may signal dissatisfaction with an employee's performance, potentially provoking workers to exert more effort out of fear of job loss or layoffs. Understanding these opposing forces is crucial for assessing the broader consequences of workplace toxicity.

Existing economic literature has primarily concentrated on toxic content in environments outside of the workplace, such as social media or online forums (Smirnov et al., 2023; Ederer et al., 2024). Research outside of economics relies largely on qualitative studies and surveys, which focus on workers' subjective perceptions of toxicity in their environment. While these studies offer valuable insights, quantifying the causal effect of toxic comments is challenging for three reasons. First, survey bias is a concern; workers who leave due to toxicity tend to be underrepresented in surveys. Second, there is the issue of misclassification. Employees may misclassify strict management as toxic, or they may fail to recognize toxic behavior because it is either not perceived as hostile or has become normalized within the workplace. Moreover, prolonged exposure to toxicity can lead workers to adopt and perpetuate such behaviors, a pattern previously documented in online gaming (Neto et al., 2017). Finally, self-selection into a work environment may lead to differences among workers, which raises endogeneity concerns. These limitations underscore the need for a more objective and systematic approach to studying workplace toxicity and its economic consequences.

In this paper, I address this important gap in the literature. Specifically, I detect toxic comments in daily workplace communication using state-of-the-art toxicity detection tools, enabling a more systematic assessment of toxic workplace environments. Using rich communication and activity data from GitHub, the world's largest web-based platform for software developers, I

provide the first empirical and theoretical evidence on the dynamic economic impacts of receiving toxic comments. By identifying when employees receive toxic comments from reviewers evaluating their code, I provide new insights into how seemingly minor yet harmful interactions can affect workers and firms.

To address concerns regarding selection bias, whereby toxic comments may correlate with workers abilities (e.g., lower-ability workers might receive more toxic feedback and face worse economic outcomes), I leverage GitHub's random reviewer assignment feature. This feature creates plausibly exogenous variation in reviewer assignments, allowing for a more credible estimation of the causal effects of receiving toxic comments on worker productivity. I implement a reviewer toxicity instrumental variable (IV) based on the average reviewer toxic comments rate in other workers a reviewer has handled, including the reviewers past and future cases. My current findings indicate that receiving toxic comments leads to a 44.9% reduction in workers output within the first six weeks and I will focus on the quality of their work and other economic outcomes in the future. Importantly, these substantial effects appear to be specific to toxicity do not appear in either non-toxic negative or positive feedback.

The remainder of this paper is structured as follows. Section 2 describes the data and sample construction. Section 3 details the measurements. Section 4 presents the stylized facts. Section 5 reports the empirical strategy and section 6 concludes.

2 Setting and Data

This section describes the data used in this paper and the different sources from which we retrieve and combine data to create the final panel. The first source is data about developer activities from the GitHub Archive, which includes timeline, activities and text communication. The second source is public profiles of GitHub users, which include their profile pictures, names, locations, and biographies. This third set of information is provided by Revelio Labs and includes public employment details and other professional characteristics. By merging these datasets, we can track developers' employment histories, such as the companies they worked for, their office locations, and their positions within the firm.

2.1 GitHub: World's Largest Web Platform for Software Developers

GitHub is a cloud-based platform for software developers to store, share, and collaborate on code projects. As of April 2024, GitHub reports having over 129 million users and more

than 205 public million repositories [1]. Repositories (projects) [2], where source code and related documents are stored, can be owned by users or organizations. As user accounts may be either a person or an automated process (a "bot"), this study will focus on the former. Each project has only one owner, but the owner can invite other users to collaborate by offering them to become members and granting them specific permissions. As Figure 1a shows, some teams operate under organizations like Microsoft and DeepSeek, while others are independent, such as users' personal websites. This analysis focuses on organizational projects, as these are more likely to be driven by a firm's profit motives than individual hobbies.

[Figure 1 about here.]

To start with GitHub, each user must create a unique login and can provide personal information that will be displayed on their profile page. This allows users to showcase their programming skills and enhance their visibility on the platform, which can be beneficial for their careers. Many developers are motivated to share their real information on GitHub; some even provide additional information, such as links to their LinkedIn profiles and personal websites.

GitHub allows developers to save snapshots of files, thereby effectively recording the history of changes and facilitating collaborative development. Furthermore, GitHub activities are available from GHArchive, which records the public GitHub timeline and event log.

2.2 GitHub Team Members

A team is defined as a project-based entity. Ideally, I prefer to focus on full-time employees within each team. However, GitHub lacks a specific indicator for such categorization. Consequently, I rely on another category related to author association: collaborator, contributor, mannequin, member, none, and owner [3]. This category has been available since 2017 and is tied to access permissions.

A team member is defined as anyone classified as a member, owner, or collaborator in the author association[4]. Since author association information is only recorded when a developer participates in specific events, months without participation result in missing author association information. To address this issue of missing values, I document the earliest and latest instances when each developer is recognized as a team member, assuming they were part of the team

[1] See https://github.com/search?q=type:user and https://github.com/search?q=is:public, accessed 2024-04-30

[2] In this paper, I use "projects" and "repositories" interchangeably, though I recognize that on GitHub, "projects" might have varied meanings.

[3] Detailed information from GitHub Docs: author association, accessed 2024-04-26

[4] I don't use MemberEvent because not every user becomes a member through the MemberEvent. Additionally, some users added via MemberEvent do not work after their addition.

during the time. It is rare for a member to leave and rejoin a team midway through the observation period.

2.3 Communication on GitHub

Communication on GitHub is essential for team collaboration. There are two main types of interactions. One occurs under issue events, where users report problems or ask questions, and team members or volunteers provide answers. The second type, which is the focus of this paper, is the pull request. In a pull request, team members submit their code changes with the intent of merging them into the main branch. This process not only facilitates code integration but also encourages thorough reviews, where other members offer feedback and suggestions to improve the code quality and ensure consistency with project standards. For simplicity, I will refer to pull requests as "cases". Figure 1b demonstrates the communication for one case and the subsequent conversation. The interaction is structured in three parts: (1) initial question/code submission, (2) discussions and proposed solutions, and (3) follow-up clarifications.

2.4 Code Reviewers

By default, any user can submit reviews that approve or request changes to a pull request, but the team can restrict which users can submit reviews that approve or request changes. If teams enable code review restrictions, anyone can comment on pull requests, but only people with read access or higher permission can approve pull requests or request changes.[5]. Figure B1 provides an example of reviewers and assignees.

2.5 Sample and Summary Statistics

Given the large pool of 205 million teams on GitHub, I focus on those within organizations(firms) to exclude hobbyist teams. The final sample includes 1,774,184 Teams and 5,620,736 Developers, with 6.83% of these being team members. Collectively, these teams and developers have produced 230 million English comments. Descriptive statistics are provided in Table 1.

[Table 1 about here.]

[5]Detailed information from Github Docs, accessed 2024-04-26

3 Linguistic Analysis

Toxicity refers to language that is rude, disrespectful, or intentionally harmful, and it is distinct from standard negative feedback(Hosseini et al., 2017). Negative feedback can be constructive and acceptable. Detecting toxic messages is challenging due to the subjective nature of human perception, as different individuals may interpret messages differently. To address this and ensure consistency, this study uses an objective, model-based approach to identify toxic sentences systematically, rather than relying on human judgment. I will first describe the key parts of the data preprocessing process. Next, I will describe the toxicity detection methods. Last, I will show how I performed sentiment analysis for each comment.

3.1 Pre-processing

First, I removed all quoted blocks of text from other users' comments to avoid misattribution. I also performed additional preprocessing, including removing code snippets from the comments. Second, I replaced several acronyms commonly used in the software engineering world with their full forms, such as *"lgtm" (look good to me)* and *"kiss" (keep it simple)*. Finally, I addressed words that have different meanings in the software engineering context than in general English but are more likely to be misclassified as toxic, such as "kill" "trash" and "dump" (Sarker et al., 2020). For example, "kill" is often used in code reviews and developer chats to suggest terminating a process or removing a piece of code, but it is more likely to be labeled as toxic in general English.

3.2 Toxicity Detection

For toxicity detection, I selected ToxiGen Roberta (Hartvigsen et al., 2022), which is good at both explicit and implicit toxicity detection(e.g., positive stereotyping, sextual content). 94.5% of toxic examples are labeled accurately by human annotators[6]. It also performs well on GitHub content. This model is a checkpoint of the RoBERTa model (Liu et al., 2019), and it predicts a binary label of toxicity and a probability score for each comment.

Here are some examples of its detection. For example, the comment *"This is bad code"* is not classified as toxic because the model interprets this as a straightforward critique aimed at improving the code, rather than as an attack on the developer. The factual tone conveys a direct assessment of the code's quality without personal criticism. When a suggestion is included, such as in the comment *"This is bad code. You can improve it by adding a loop"*, the model continues to recognize the comment as non-toxic, but the inclusion of constructive feedback increases the

[6]FigureC1 shows the evaluation form when human annotators labeled hate speech for Toxigen

model's confidence. This is because the suggestion provides clear guidance on how to enhance the code, further distancing the statement from toxic behavior.

Conversely, when the comment contains phrases like *"intrinsically bad code"*, the model classifies it as toxic. This is due to the use of the word "intrinsically" which suggests a fundamental and unalterable flaw, potentially disparaging the developer's skills or effort more broadly. Such comment can be seen as a personal and demeaning judgment, not just of the work but of the person's inherent capabilities.

Additionally, the model is good at detecting implicit toxicity and discriminatory remarks. For example, statements like *"We can learn nothing from your code"* indicates that the work has no value, which can demotivate and insult the developer in a subtle way. Similarly, comments such as *"Female developers are not good at it"* are flagged as toxic because they contain discriminatory views based on gender, promoting stereotypes and undermining individual merit.

1. *This is bad code* → non-toxic, probability: 0.621
2. *This is bad code. You can improve it by adding a loop* → non-toxic, probability: 0.983
3. *This is intrinsically bad code* → toxic, probability: 0.855
4. *Are you stupid, this is bad code* → toxic, probability: 0.979
5. *We can learn nothing from your code* → toxic, probability: 0.96
6. *Female developers are not good at it* → toxic, probability: 0.95

3.3 Sentiment Detection

TimeLMs is a language model specifically trained on tweets spanning the period from 2018 to 2021(Loureiro et al., 2022). The model is capable of predicting three distinct sentiment labels: negative, positive, and neutral. For each classification, TimeLMs also provides confidence scores, indicating the model's level of certainty for each predicted label. This combination of sentiment classification and confidence scoring makes TimeLMs particularly useful for analyzing social media data, where context, tone, and temporal relevance often play a critical role in understanding sentiment.

3.4 Detection Results

Figure 2a shows the distribution of predicted probability for toxic comments, with a notable concentration between 0.9 and 1.0. This suggests that most toxic comments in our context are detected by Toxigen with a higher probability. Table2 presents the distribution of comments by

toxicity and sentiment for all English comments in the sample teams, focusing on two characteristics of each comment: toxicity and sentiment. Approximately 0.2% of the comments are toxic, with the majority of toxic comments being negative(0.14%). Non-toxic comments overwhelmingly exhibit neutral sentiment (64.73%), followed closely by positive (17.66%) and negative (17.56%) sentiments. Table C1 shows randomly selected comments for each combination.

[Table 2 about here.]

4 Descriptive

This section documents stylized facts about toxic communication. senders are generally more senior and influential, with a notable gender and racial skew. Toxic comments primarily flow from senior to junior developers, suggesting seniors are more likely to engage in toxic behavior, particularly during feedback sessions.

Fact 1 *Escalating toxicity: Higher initial toxicity leads to more toxic replies*

Figure 2 shows a feedback loop where initial messages with high toxicity lead to increasingly toxic replies. Panel 2b presents that as the toxicity score of an initial message increases, so does the proportion of toxic replies. This points to a pattern of escalation within interactions, where toxic messages provoke increasingly toxic responses. Panel 2c and Panel 2d display randomly selected examples of toxic comments and their replies. These interactions often occur in contexts where senior members provide feedback or evaluations to juniors, and developers who receive toxic comments frequently respond by explaining their actions or decisions.

[Figure 2 about here.]

Fact 2 *Toxic comments primarily flow from senior to junior developers*

Senders display distinct characteristics compared to receivers and others. Senders are defined as individuals who send toxic messages within a pull request. In contrast, receivers include all other participants in the same pull request on the same day. This definition helps exclude developers who join the pull request after the toxic interactions have occurred and who are not directly exposed to the toxic comments. The category of "others" comprises individuals who do not participate in any toxic interactions. Figure 3 highlights differences in experience, influence, and demographic characteristics among the three groups. senders typically possess higher levels of GitHub experience[7]. They also tend to have a larger average follower count, which suggests greater visibility or influence on the platform.

[7]Experience is defined as the difference between the year of their first recorded toxic interaction and their account creation year

Toxic comments primarily flow from senior to junior developers, suggesting a hierarchical dynamic where seniors are more likely to engage in toxic behavior, particularly during feedback sessions.

Fact 3 *Male and white race developers are more likely to send toxic comments*

Figure 3c and 3d highlight demographic differences between senders and receivers. Senders are more likely to be male and predominantly of Greater European descent compared to receivers and others[8]. These characteristics indicate that senders are generally more senior and influential, with a notable gender and racial skew.

[Figure 3 about here.]

5 Empirical

Descriptive evidence indicates that toxic comments predominantly arise from senior developers directed at junior developers. In this section, I will focus on the toxic comments made by code reviewers towards code submitters(workers). Using a random assignment of reviewers and an instrumental variable based on reviewer toxicity, I will estimate the causal effect of receiving toxic comments on workers' labor market outcomes.

5.1 Reviewer IV

I aim to identify the effect of toxic comments on the subsequent labor market outcomes for the recipients. This relationship is modeled by the regression

$$Y_{ik} = \beta_k I_{i0} + \beta_2 \mathbf{X}_i + \varepsilon_{it}, \tag{1}$$

where Y_{ik} is the outcome of interest for individual i at time k (e.g., output quality, output quantity, retention), I_{i0} is an indicator variable equal to 1 if individual i received toxic comments in $k = 0$ (normalized to the time of the reviewer decision), $\mathbf{X}_i$ includes control variables for worker and case characteristics[9]. I also control team $\times$ year fixed effect, and ε_{ik} is the error term. β_k captures the causal effect of receiving toxic comments on subsequent outcomes.

[8]Gender and Race are detected based on developers' profile images and names. The Appendix Section A provides additional details.

[9]Worker characteristics: gender, race, number of followers (2023), GitHub activity in the year before submission, the first year of GitHub activity(reflecting experience), and whether the submitter has a biography (indicating real-world identity). Case characteristics: labels such as bug, Python, JavaScript, dependency, and enhancement

In practice, the receipt of toxic comments is not random; therefore, estimating equation (1) directly may yield biased estimates of β_k due to unobserved factors that affect both the likelihood of receiving toxic comments and labor market outcomes. For example, workers with lower abilities might produce disappointing code, prompting managers to send toxic comments, while that same lower ability could lead to poorer labor market outcomes. To address this concern, my research design takes advantage of the random assignment of cases to reviewers (conditional on team and year fixed effects) and the systematic variation in reviewer toxicity. Taken together, these factors create exogenous variation in the probability of receiving toxic comments, based on the assigned reviewer. I use this exogenous variation in $I_{i,0}$ to draw inferences about the causal effects of toxic comments.

The main analysis employs a two-stage least squares (2SLS) approach, with equation (1) as the second equation. The corresponding first-stage equation is given by

$$I_{i0} = \gamma Z_{j(i)} + \delta \mathbf{X}_i + \varepsilon_{i0}, \tag{2}$$

where $Z_{j(i)}$ denotes the toxicity level of the reviewer assigned to individual i, and γ represents the impact of reviewer toxicity on the likelihood of receiving toxic comments. To account for potential serial correlation, standard errors are clustered at the reviewer level in both stages. Under the assumptions of instrument exogeneity and monotonicity, the 2SLS estimator can be interpreted as a positively weighted average of the causal effects of toxic comments among the subgroup of developers whose chances of receiving such comments vary based on the assigned reviewer.

In line with prior literature (e.g., Dobbie et al. (2018, 2017); Dobbie and Song (2015); Beuschlein (2024)), Z_{imj} is defined as the leave-worker-out[10] fraction of cases with toxic comments by reviewer j:

$$Z_{ij} = \frac{1}{\sum_{h \neq i} n_{hj}} \sum_{h \neq i} D_{hj}$$

where D_{ij} is an indicator equal to 1 if i receiving toxic comments from reviewer j,0 if non-toxic comments and n_{ij} is the number of cases submitted by worker i and reviewed by j.

5.2 Random Reviewer Assignment

I exploit GitHub's random reviewer assignment feature, which introduces exogenous variation in reviewer and worker matches. This feature automatically assigns reviewers to cases(pull

[10]In the judge IV literature, the standard method for constructing IV involves leaving the case out. In my context, since workers and reviewers may interact multiple times, I leave out the worker and compute the average toxic comment rate of the reviewers in other cases. Thus, there is no mechanically induced dependence between the outcome for worker i and out reviewer toxicity measures

requests), where they evaluate code, decide on code submission approval, and provide comments during the review process. Teams enable this feature through configuration files and the GitHub Robot assigns reviewers automatically. This setup standardizes the review process and reduces potential biases, providing plausibly exogenous variation in reviewer assignment. This variation allows for a more credible identification of the causal effects of receiving toxic comments. Figure B2 illustrates different types of reviewer assignments.

5.3 Accessing the Instrument

Instrument Relevance. Figure 4 shows the graphical representation of the first stage, with a histogram that shows the distribution of the instrument with team × year fixed effects. The IV is the average toxic comments rate in other code submissions a reviewer has handled. 35,495 teams adopted the feature. To reduce noise I drop all observations that are estimated using 10 or fewer cases. After removing these observations, 12,304 teams remain, where each reviewer handles more than 10 cases per year, and each team has at least one reviewer per year. My final sample contains 2,856,921 cases, 24,826 reviewers, and 115.08 cases per reviewer. The reviewer toxicity measure ranges from 1% to 99%, spanning from -0.047 to 0.083, with a standard deviation of 0.024. Moving from the least hostile reviewer (1st percentile, -0.047) to the most hostile reviewer (99th percentile, 0.083) increases the probability of sending toxic comments by 13 percentage points, an increase of 13.68% relative to the mean toxicity rate of 0.95 percentage points.

Figure 4 also plots the first-stage relationship between the residualized measure of reviewer toxicity and the probability of receiving hostile comments, while accounting for the exhaustive set of team-by-time fixed effects. The graph overlays the distribution of reviewer toxicity and provides a flexible analog to equation (2). It displays a local linear regression of the probability of receiving toxic comments against reviewer toxicity. The likelihood of receiving toxic comments is monotonically increasing in the reviewer toxicity and is close to linear. Table 3 reports first-stage estimates where I regress a dummy for whether workers get toxic comments on the reviewer instrument. Column 1 of Table 3 only with team-by-year fixed effects. The estimates are highly significant, suggesting that being assigned to a reviewer with a 10 percentage-point increase in the residualized reviewer's toxicity rate is associated with an approximately 6.3 percentage-point increase in the probability of receiving toxic comments.

[Figure 4 about here.]

[Table 3 about here.]

Conditional independence. For the instrument to be valid, reviewer toxicity must be uncorrelated with worker and case characteristics that could affect future outcomes. Column 2 of Table 3 examines the impact of adding a set of control variables to the first-stage regressions. If reviewers are randomly assigned, predetermined variables should not significantly affect the estimates. As expected, the coefficient remains stable when controlling for code submitters' and pull request characteristics.

As a second test, Figure 5 confirms that case assignment to reviewers is random after accounting for team-by-year fixed effects. The red lines in the figure show OLS coefficients for toxic comments based on standardized submitter and case characteristics. It shows cases with 'Bug' and 'Dependency' related tasks are less likely to elicit toxic comments. The blue lines represent coefficients for reviewer toxicity and demonstrate that standardized submitter and case characteristics do not predict reviewer toxicity under an identical specialization. These findings provide strong evidence that reviewers with varying levels of toxicity are assigned cases and workers in a manner consistent with random assignment. Appendix Table D1 presents the results of this test.

[Figure 5 about here.]

Monotonicity condition. If the causal effect of reviewer toxicity is constant across code submitters, the instrument only needs to satisfy the conditional independence and exclusion restrictions. However, with heterogeneous effects, the monotonicity assumption must also hold to estimate convex combinations of treatment effects (Bhuller et al., 2020). Following Bhuller et al. (2020); Beuschlein (2024), I test this assumption by splitting the sample based on gender, race, GitHub experience, then re-estimating the first stage within each subsample. If the monotonicity assumption holds, the first-stage coefficient should remain positive across all subgroups. Appendix Table D2 presents the results of this test and shows that all coefficients are positive, as required.

Exclusion restriction. Conditional random assignment of cases to reviewers is sufficient for interpreting the reduced-form impact of being assigned a hostile reviewer as causal. However, interpreting the IV estimates as the causal effect of toxic comments requires an exclusion restriction: reviewer toxicity should influence code submitters' outcomes directly through the toxicity comment channel and not through any other channels. The challenge lies in the multidimensional nature of comments, where reviewers may choose to be toxic, negative, positive or neutral feedback. After presenting the main results, I will provide empirical evidence supporting the validity of the exclusion restriction. Specifically, I will demonstrate that the estimates

remain robust when the baseline model is augmented to control for reviewer toxicity in other dimensions.

5.4 Main results

Table 4 presents the current results on average worker activities in comments received teams during the six weeks following the start of receiving comments from reviewers. The 2SLS results indicate that receiving toxic comments reduces worker output, as measured by activities, by 44.9% in the first 1 to 6 weeks. However, we do not observe a significant effect on the extensive margin. It is possible that workers may switch to work in other teams once they received toxic comments in the current teams. To check if there is any spillover effect, Table **??** shows the effect of receiving toxic comments on other teams and I don't find substitution behavior happened.

[Table 4 about here.]

[Table 5 about here.]

Since toxic comments are highly correlated with negative sentiment, as shown in Table 2, it is important to distinguish whether the effect is unique to toxicity or simply a response to standard criticism. Table 5 shows the effect of receiving nontoxic but negative comments on workers' output. Receiving toxic comments will still make workers have less output but the effect from toxic comments are 3.6 times larger than non-toxic but negative comments.

[Table 6 about here.]

5.5 Threats to Exclusion Restriction

As discussed in Section 5.3, interpreting the IV estimates as average causal effects of toxic comments requires that the reviewer's toxicity instrument affects workers' outcomes only through the toxic comments channel. A potential issue is that comments are multidimensional, meaning reviewers may provide toxic, negative, positive, or neutral feedback.

There are two scenarios in which the baseline IV estimates based on equations (1) and (2) may be biased because they ignore sentiments other than toxicity. The first scenario arises if other reviewer characteristics, $Z_{j(i)}^{\text{Other}}$, are correlated with reviewer toxicity, $Z_{j(i)}^{\text{Toxic}}$, and $Z_{j(i)}^{\text{Other}}$ directly affects Y_{it} (conditional on $\mathbf{X}_i$). This would violate the exclusion restriction in the baseline IV model because $Z_{j(i)}^{\text{Toxic}}$ influences Y_{it} not only through I_{i0}^{Toxic} but also via its correlation with $Z_{j(i)}^{\text{Other}}$. For example, if toxic reviewers are also more critical, and this criticism adversely

affects workers' outcomes, the exclusion restriction is violated. However, controlling for $Z_{j(i)}^{\text{Other}}$ in both equations (1) and (2) can eliminate this source of bias.

The second scenario occurs if $Z_{j(i)}^{\text{Toxic}}$ is correlated with other types of comments—for instance, if toxic reviewers are more likely to send non-toxic but negative comments, which in turn affect Y_{it} independently of I_{i0}^{Toxic}.

In Table D3, I examine the first scenario to assess the validity of the exclusion restriction. First, I calculate reviewers' tendencies to send non-toxic but negative or positive comments. For example, a reviewer's non-toxic but negative tendency is measured as the average number of negative, non-toxic messages they have sent to other workers. Panel A of Table D3 repeats the baseline specification for comparison. In Panel B, I add reviewers' negative tendencies as additional controls in both the first and second stages, using comments that are neither negative nor toxic as the omitted category. The IV estimates in Panel B are very similar to those in the baseline. In Panel C, I include reviewers' non-toxic but positive tendencies as controls, and once again, the IV estimates remain consistent with the baseline specification in Panel A.

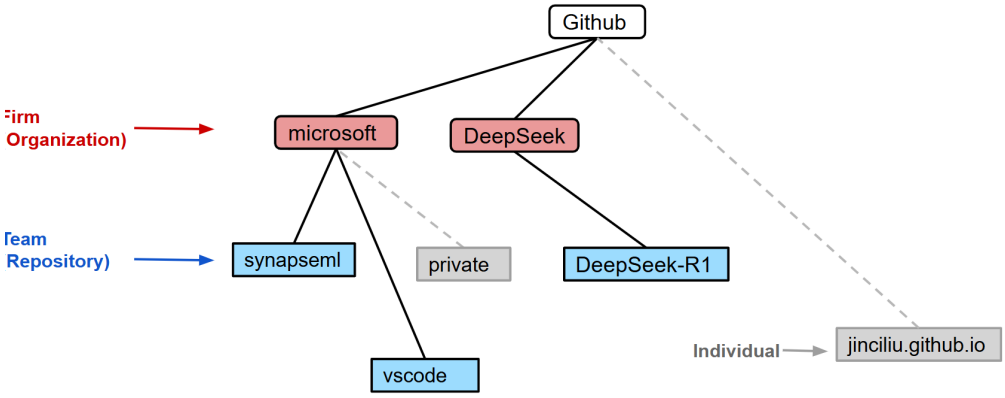
In Table D4, I examine the second scenario and formally evaluate the overall strength of the instruments. I find that toxic comments continue to reduce workers' output even after controlling for standard negative comments.

6 Conclusions

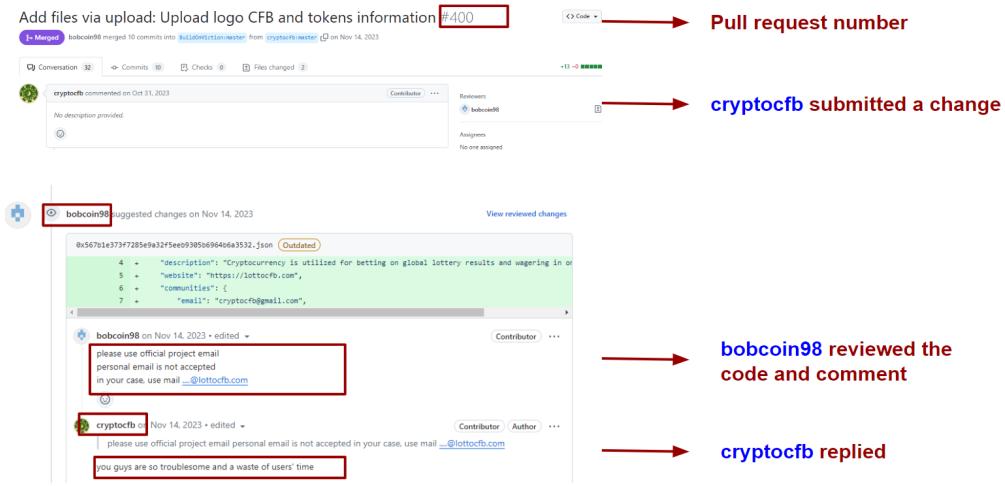
This paper provides new empirical evidence on the economic impacts of toxic comments in the workplace. Using state-of-the-art toxicity detection tools and detailed communication data from GitHub, I have shown that toxic feedback has a significant and negative effect on worker productivity. Specifically, receiving toxic comments reduces workers' output. The findings underscore the importance of addressing toxic behavior in workplace communication—not only to protect workers' mental health but also to improve overall productivity. Future research will extend this analysis to examine additional economic outcomes, including work quality, to further illuminate the full impact of toxic workplace environments.

Figure 1: GitHub Structure and Communication

(a) Organization and Team



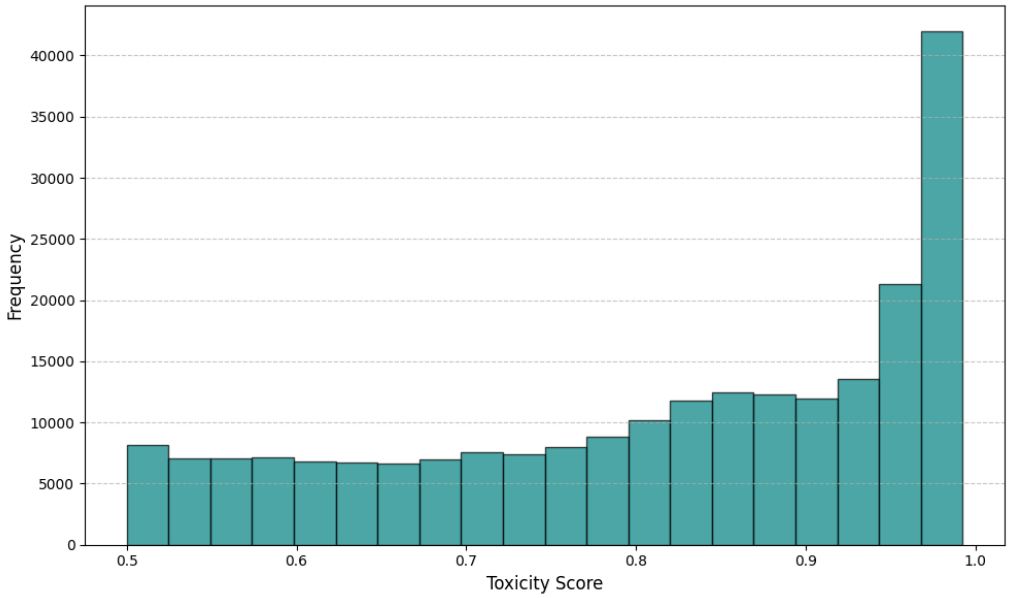
(b) Communication



Notes: As shown in Figure 1a, some repositories belong to the organization, while others do not. In this project, I include only repositories affiliated with the organization to exclude those intended for individual purposes or hobbies, such as personal websites. Figure 1b demonstrates the communication about adding files Pull Request (#400) and the subsequent conversation. The interaction is structured in three parts: (1) initial question/code submission, (2) discussions and proposed solutions, and (3) follow-up clarifications.

Figure 2: Toxic Messages and Replies

(a) Distribution of Toxicity Score by ToxiGen



(b) Proportion of Replies is Toxic

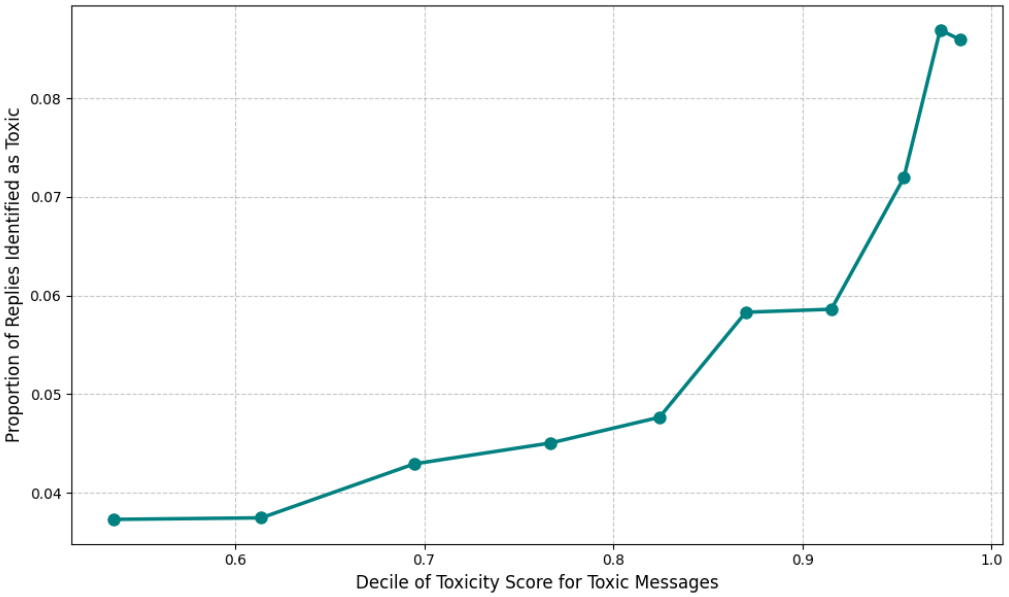
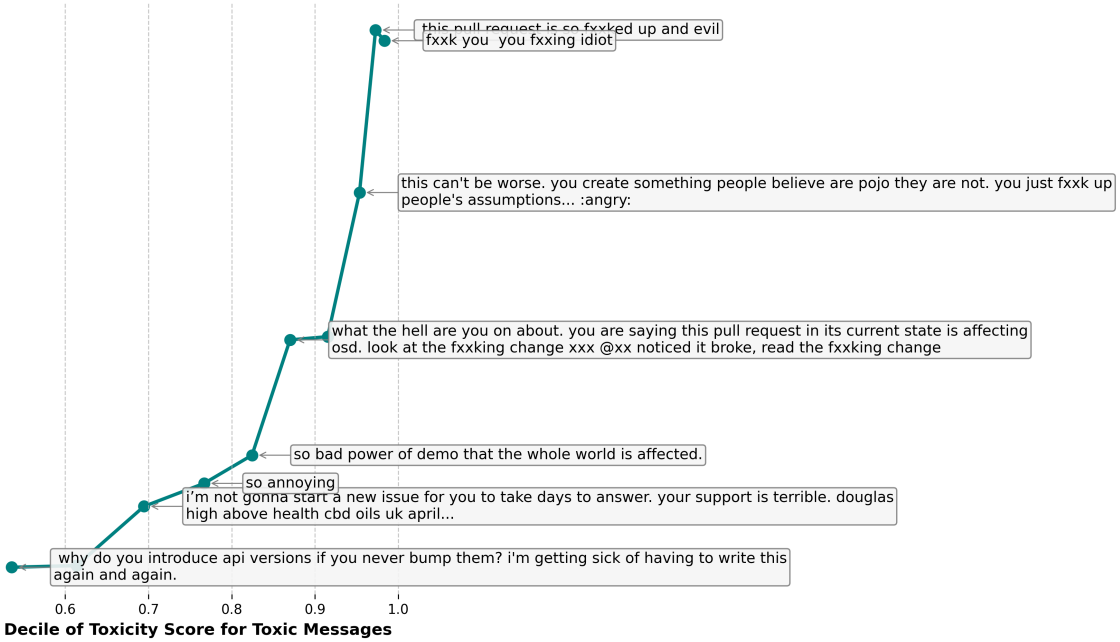
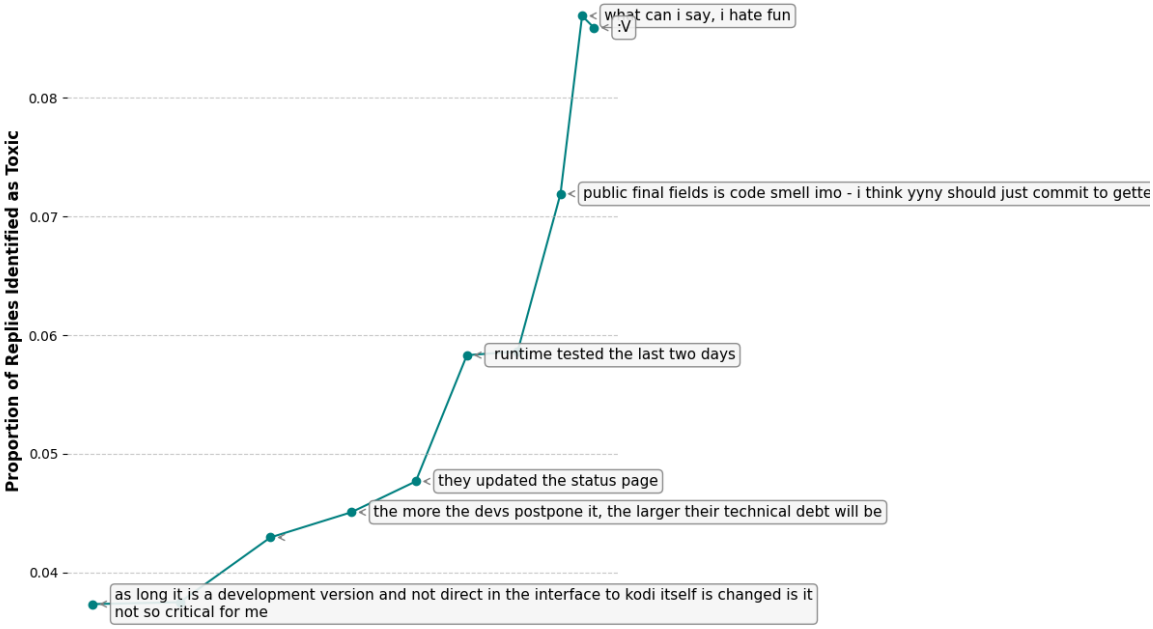


Figure 2: Toxic Messages and Replies

(c) Toxic Messages



(d) Corresponding Replies

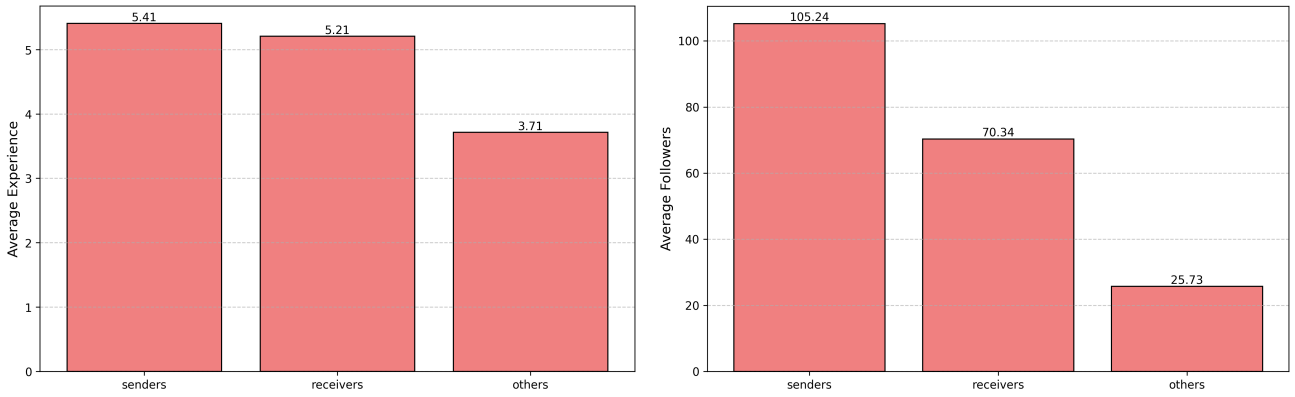


Notes: Panel 2a illustrates the distribution of toxicity scores, revealing that most toxic comments are identified with high confidence scores between 0.9 and 1.0. This suggests that most toxic comments in our context are detected by Toxigen with a higher certainty. Panel 2b displays a line graph indicating that the proportion of replies classified as toxic increases along with the toxicity score of the initial message. Essentially, the more toxic the initial message, the greater the likelihood of receiving toxic replies. Figure C1 shows randomly selected examples for toxic comments and replies.

Figure 3: Characteristics of Sender, Receiver and Others

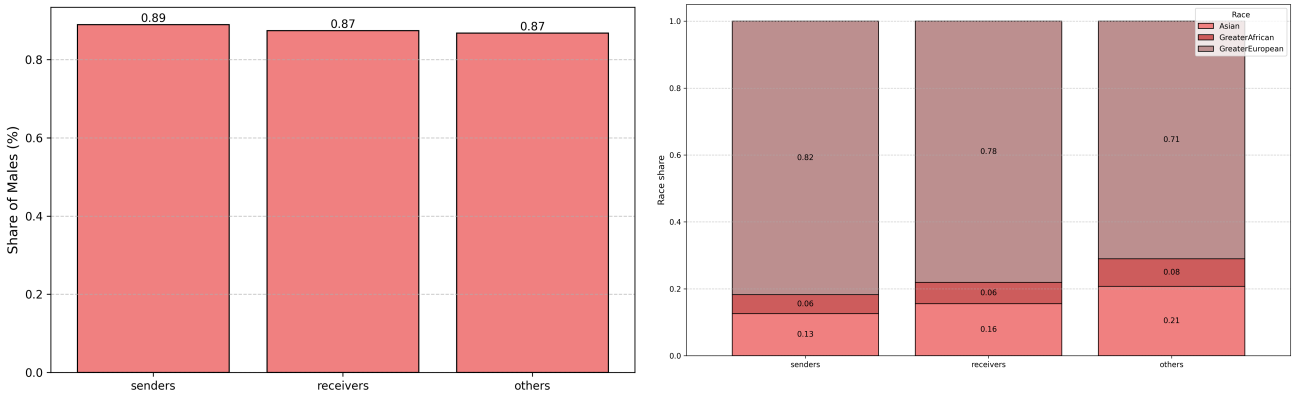
(a) GitHub Experience

(b) GitHub Followers



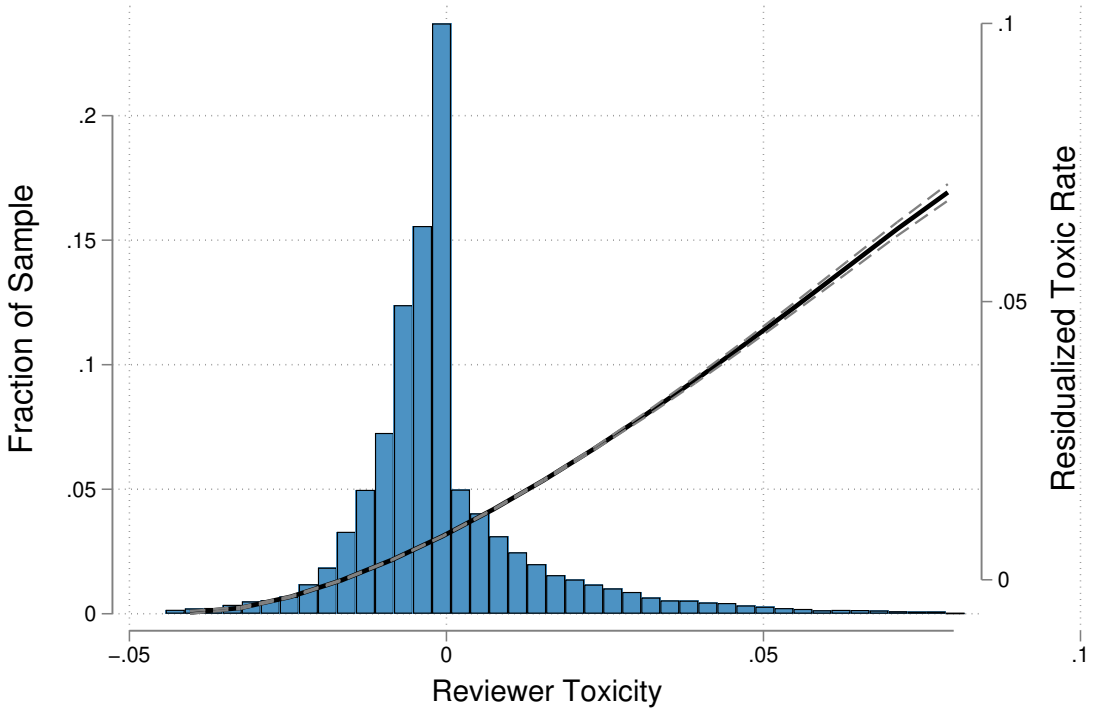
(c) Gender

(d) Race



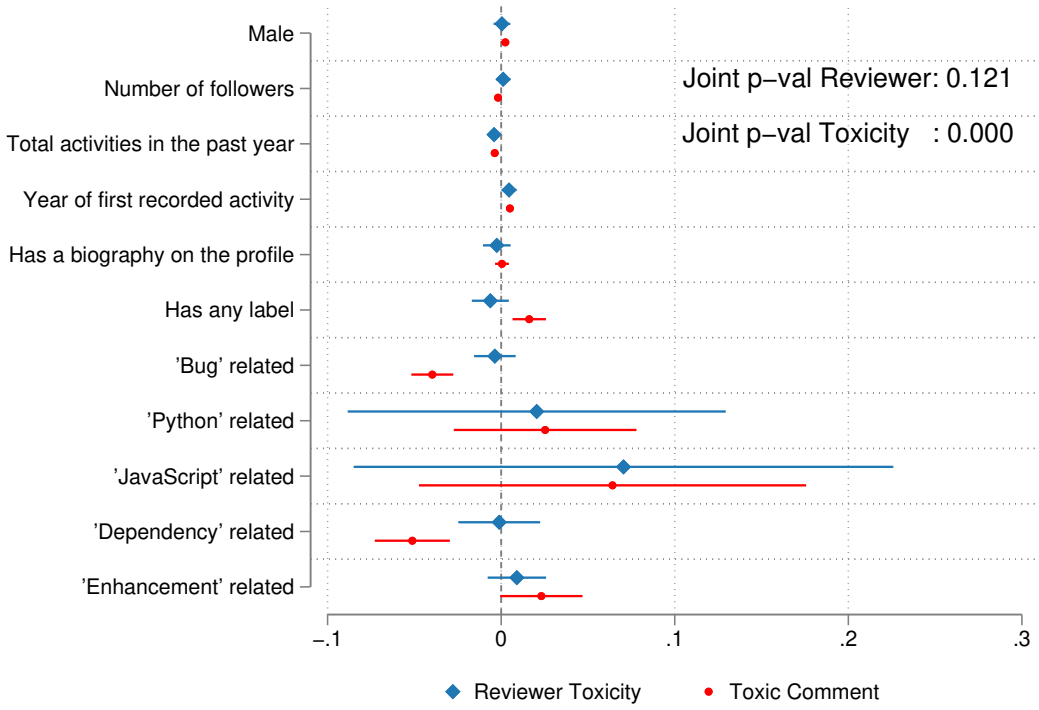
Notes: This figure illustrates the characteristics of toxic comments senders, receivers, and others. Senders are defined as individuals who send toxic messages in a pull request, while receivers include everyone else involved in the same pull request on the same day. Others refer to individuals who are not part of any toxic interaction. Gender and Race are detected based on developers' profile images and names. The appendix section A provides additional details. Figure3a shows senders have slightly higher average experience levels (calculated as First Toxic Year-Account Create Year) compared to receivers and others. Figure3b shows senders also have the highest average follower count (105.24), followed by receivers (70.34) and others. For the gender difference, Figure3c indicates a slightly higher proportion of male users compared to other groups. For the racial distribution, senders are more likely to belong to the Greater European race than other groups as in Figure 3d.

Figure 4: Distribution of Reviewer toxicity Measure and First Stage



Notes: First stage graph of toxic rate on reviewer toxicity. The baseline sample consists of 2,856,921 code submissions. The probability of receiving toxic comments is plotted on the right Y-axis against leave-worker-out mean receiver toxicity of the assigned reviewers, shown along the x-axis. The plotted values are mean-standardized residuals from a local linear regression, reporting the residuals from regressions on team× year fixed effects. The solid line shows a local linear regression of toxic rate on reviewer toxicity. Dashed line shows 95% confidence intervals. The histogram shows the fraction of reviewer toxicity along the left Y-axis (top and bottom 1% excluded).

Figure 5: Balance of Instrument



Notes: This figure presents OLS regression coefficients for reviewer toxicity and toxic comments on standardized code submitters' and case characteristics. The p-values reflect joint significance tests from OLS regressions with team-by-year fixed effects. Balance variables include gender, race, number of followers (2023), GitHub activity in the year before submission, first year of GitHub activity(reflecting experience), and whether the submitter has a biography (indicating real-world identity). Case characteristics include labels such as bug, Python, JavaScript, dependency, and enhancement. Standard errors are clustered at the team level.

Table 1: Descriptive Statistics

Description	**Statistic**
Teams (#)	1,774,184
Teams with toxic messages (%)	5.75
Avg. number of PR per team (#)	10.08
Avg. developers per PR (#)	1.88
Avg. age for teams (years)	1.75
Developers with messages (#)	5,620,736
Developers sent toxic messages (%)	2.21
Developers received toxic messages (%)	3.87
Avg. number of toxic messages per developer (#)	2.95
Members (%)	8.53
Members are reviewers (%)	69.75
Avg. number of pull-request per reviewers reviewed (#)	25.27
Messages (#)	231,293,881
Messages are toxic (%)	0.20
Toxic messages sent by members (%)	19.23
Toxic messages sent by reviewers (%)	10.40
Messages are negative (%)	17.56
Messages are positive (%)	17.66
Avg. number of toxic messages per team (#)	4.49
Avg. number of messages PR (#)	4.08

Notes: This table provides summary statistics for key variables from GitHub, covering the period from 2017 to 2023. Only teams with at least one member and have over 100 messages in any single year from 2017 to 2023 are included.

Table 2: Share of Comments by Toxicity and Sentiment (%)

	Sentiment		
Toxigen	**Positive**	**Negative**	**Neutral**
Non-toxic	16.34	16.12	59.92
Toxic	0.01	0.13	0.05

Notes: This table presents detection results for English comments from sample teams, categorizing each by toxicity and sentiment. Approximately 0.2% of comments are toxic, with the majority of these being negative.

Table 3: First-Stage Estimates of Toxicity on Reviewer Toxicity

Dep Var: I_{i0}	(1)	(2)	(3)	(4)
Reviewer Toxicity	0.635***	0.631***	0.631***	0.631***
	(0.014)	(0.015)	(0.015)	(0.015)
Team FE	Y			
Team×Year FE		Y	Y	Y
Baseline controls			Y	Y
Quantile $k-1$ activity FE				Y
F-Stat.	1996.772	1833.792	1799.857	1801.879
Dependent mean	0.009	0.009	0.009	0.009
N	2791307	2790578	2766507	2766507

Notes: This table reports the first-stage results. Baseline controls include gender, race, number of followers (2023), GitHub activity in the year before submission, first year of GitHub activity(reflecting experience), and whether the submitter has a biography (indicating real-world identity). Case characteristics include labels such as bug, Python, JavaScript, dependency, and enhancement. Standard errors are clustered at the reviewer level.$p < .10$, ** $p < .05$, *** $p < .01$

Table 4: Effect of Toxicity on Workers

	First Stage	**Reduced Form**		**IV**	
	I_{i0}	Ever	Log	Ever	Log
Weeks 1-6 Post					
I_{i0}				-0.007	-0.449***
				(0.007)	(0.126)
Reviewer Toxicity	0.631***	-0.004	-0.285***		
	(0.015)	(0.005)	(0.081)		
F.Stat.	1801.879				
Dependent mean	0.009	0.982	3.076	0.982	3.0759
N	2766507	2766507	2715728	2715728	2766507
Team × Year FE	Y	Y	Y	Y	Y
Quantile $k-1$ activity FE	Y	Y	Y	Y	Y
Controls	Y	Y	Y	Y	Y

Notes: This table presents OLS and two-stage least squares (2SLS) results on the impact of toxicity. The outcome variables are the average activities within the team in the post-period, with intensive outcomes measured as the log of the outcome variables. The 2SLS models instrument toxic comments using a reviewer hostility measure, estimated from data on other workers' cases assigned to the same reviewer, as described in Section 5.1. All models include team-by-year fixed effects, and standard errors are clustered at the reviewer level.$p < .10$, ** $p < .05$, *** $p < .01$.

	First Stage	**Reduced Form**		**IV**	
	I_{i0}	Ever	Log	Ever	Log
Weeks 1-6 Post					
I_{i0}				0.007	-0.297
				(0.048)	(0.268)
Reviewer Toxicity	0.631***	0.005	-0.191		
	(0.015)	(0.030)	(0.172)		
F.Stat.	1801.879				
Dependent mean	0.009	0.819	1.885	0.819	1.885
N	2766507	2766507	2264939	2766507	2264939
Team × Year FE	Y	Y	Y	Y	Y
Quantile $k-1$ activity FE	Y	Y	Y	Y	Y
Controls	Y	Y	Y	Y	Y

Notes: This table presents OLS and two-stage least squares (2SLS) results on the impact of toxicity. The outcome variables are the average activities in other teams in the post-period, with intensive outcomes measured as the log of the outcome variables. The 2SLS models instrument toxic comments using a reviewer hostility measure, estimated from data on other workers' cases assigned to the same reviewer, as described in Section 5.1. All models include team-by-year fixed effects, and standard errors are clustered at the reviewer level.$p < .10$, ** $p < .05$, *** $p < .01$.

Table 5: Effect of Nontoxic-Negative Comments on Workers

	First Stage	**Reduced Form**		**IV**	
	I_{i0}^{NT-N}	Ever	Log	Ever	Log
Weeks 1-6 Post					
I_{i0}^{NT-N}				-0.000	-0.124***
				(0.001)	(0.020)
Reviewer NT-N	0.810***	-0.000	-0.101***		
	(0.004)	(0.001)	(0.016)		
F.Stat.	1801.879				
Dependent mean	0.009	0.982	3.076	0.982	3.0759
N	2766507	2766507	2264939	2766507	2264939
Team × Year FE	Y	Y	Y	Y	Y
Quantile $k-1$ activity FE	Y	Y	Y	Y	Y
Controls	Y	Y	Y	Y	Y

Notes: This table presents OLS and two-stage least squares (2SLS) results on the impact of non-toxic negative. The outcome variables are the average activities within the team in the post-period, with intensive outcomes measured as the log of the outcome variables. The 2SLS models instrument toxic comments using a reviewer nontoxic-negative(NT-N) measure, estimated from data on other workers cases assigned to the same reviewer. All models include team-by-year fixed effects, and standard errors are clustered at the reviewer level.$p < .10$, ** $p < .05$, *** $p < .01$.

References

American Psychological Association (2024). 2024 work in america survey: psychological safety in the changing workplace.

Beuschlein, J. (2024). Designing debt restructuring: The adverse effects on labor market outcomes. Technical report, Working Paper. Stockholm University.

Bhuller, M., G. B. Dahl, K. V. Løken, and M. Mogstad (2020). Incarceration, recidivism, and employment. *Journal of political economy 128*(4), 1269–1324.

Dobbie, W., J. Goldin, and C. S. Yang (2018). The effects of pre-trial detention on conviction, future crime, and employment: Evidence from randomly assigned judges. *American Economic Review 108*(2), 201–240.

Dobbie, W., P. Goldsmith-Pinkham, and C. S. Yang (2017). Consumer bankruptcy and financial health. *Review of Economics and Statistics 99*(5), 853–869.

Dobbie, W. and J. Song (2015). Debt relief and debtor outcomes: Measuring the effects of consumer bankruptcy protection. *American economic review 105*(3), 1272–1311.

Ederer, F., P. Goldsmith-Pinkham, and K. Jensen (2024). Anonymity and identity online. *arXiv preprint arXiv:2409.15948*.

Fattori, A., L. Neri, E. Aguglia, A. Bellomo, A. Bisogno, D. Camerino, B. Carpiniello, A. Cassin, G. Costa, P. De Fazio, et al. (2015). Estimating the impact of workplace bullying: humanistic and economic burden among workers with chronic medical conditions. *BioMed research international 2015*(1), 708908.

Hartvigsen, T., S. Gabriel, H. Palangi, M. Sap, D. Ray, and E. Kamar (2022). Toxigen: A large-scale machine-generated dataset for implicit and adversarial hate speech detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Hosseini, H., S. Kannan, B. Zhang, and R. Poovendran (2017). Deceiving google's perspective api built for detecting toxic comments. *arXiv preprint arXiv:1702.08138*.

Laohaprapanon, S., G. Sood, and B. Naji (2017). ethnicolr algorithm.

Liu, Y., M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Loureiro, D., F. Barbieri, L. Neves, L. Espinosa Anke, and J. Camacho-collados (2022, May). TimeLMs: Diachronic language models from Twitter. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Dublin, Ireland, pp. 251–260. Association for Computational Linguistics.

McTernan, W. P., M. F. Dollard, and A. D. LaMontagne (2013). Depression in the workplace: An economic cost analysis of depression-related productivity loss attributable to job strain and bullying. *Work & Stress 27*(4), 321–338.

Neto, J. A., K. M. Yokoyama, and K. Becker (2017). Studying toxic behavior influence and player chat in an online video game. In *Proceedings of the international conference on web intelligence*, pp. 26–33.

Nielsen, M. B. and S. Einarsen (2012). Outcomes of exposure to workplace bullying: A meta-analytic review. *Work & Stress 26*(4), 309–332.

Sarker, J., A. K. Turzo, and A. Bosu (2020). A benchmark study of the contemporary toxicity detectors on software engineering interactions. In *2020 27th Asia-Pacific Software Engineering Conference (APSEC)*, pp. 218–227. IEEE.

Smirnov, I., C. Oprea, and M. Strohmaier (2023). Toxic comments are associated with reduced activity of volunteer editors on wikipedia. *PNAS nexus 2*(12), pgad385.

Zsila, Á., R. Shabahang, M. S. Aruguete, and G. Orosz (2022). Toxic behaviors in online multiplayer games: Prevalence, perception, risk factors of victimization, and psychological consequences. *Aggressive Behavior 48*(3), 356–364.

Online Appendix of:

Toxic Talk, Lasting Harm: How Words Affect Productivity and Retention

Jinci Liu

February 27, 2025

A Developer Demographic Detection

A.1 Gender Prediction

To predict the gender of developers based on their avatars, I select a state-of-the-art model that closely matches the images in the training sample in terms of resolution and cropping methods. This chosen model[11] trains vision transformers on male and female portraits and develops a corresponding image classifier. The model predicts the probability that an image represents a female or a male. To determine the optimal cutoff for gender classification, I use a sample labeled by some annotators and find that the highest accuracy is achieved with a cutoff of 0.85. If the predicted probability is above 0.85, the image will be classified as male or female; otherwise, it will be labeled as 'others.'

A.2 Race Prediction

I choose Ethnicolr, as developed by Laohaprapanon et al. (2017), to predict the race and ethnicity of developers based on their names. Ethnicolr utilizes first and last names to classify individuals into four combined race and ethnicity categories: Hispanic (of any race), and non-Hispanic Asian, Black, and White. The algorithm is trained on data from the US Census, Florida voter registration, and Wikipedia. Each name is assigned probabilities for belonging to each of the four categories, with the category of highest probability being designated as the predicted race or ethnicity.

B Random Assignment Reviewers

GH Archive does not provide direct information on when teams adopt the random assignment of reviewers feature, as this is not visible in timelines. My objective is to identify pull request reviewers who are assigned automatically by the algorithm. To address this, I propose two methods. First, since most review requests are initiated by the pull request submitter (code submitter), if a pull request is created by a bot account and includes non-bot reviewers, it likely relies on configuration files for automatic reviewer assignment, as shown in FigureB2b. I use the first instance of a bot account submitting a pull request with non-bot reviewers as the adoption event. Second, I use the time when a team adds a relevant configuration file as a validation event. However, since it is not feasible to download and analyze files from over one million teams, I focus on the most commonly used configuration files, such as `.github/codeowners` and

[11] https://huggingface.co/rizvandwiki/gender-classification, accessed 2024-04-23

`.github/auto_assign.yml`, which are typically used for automatic reviewer assignment. TableB1 provides the share of teams in big tech firms that adopt this feature.

[Table B1 about here.]

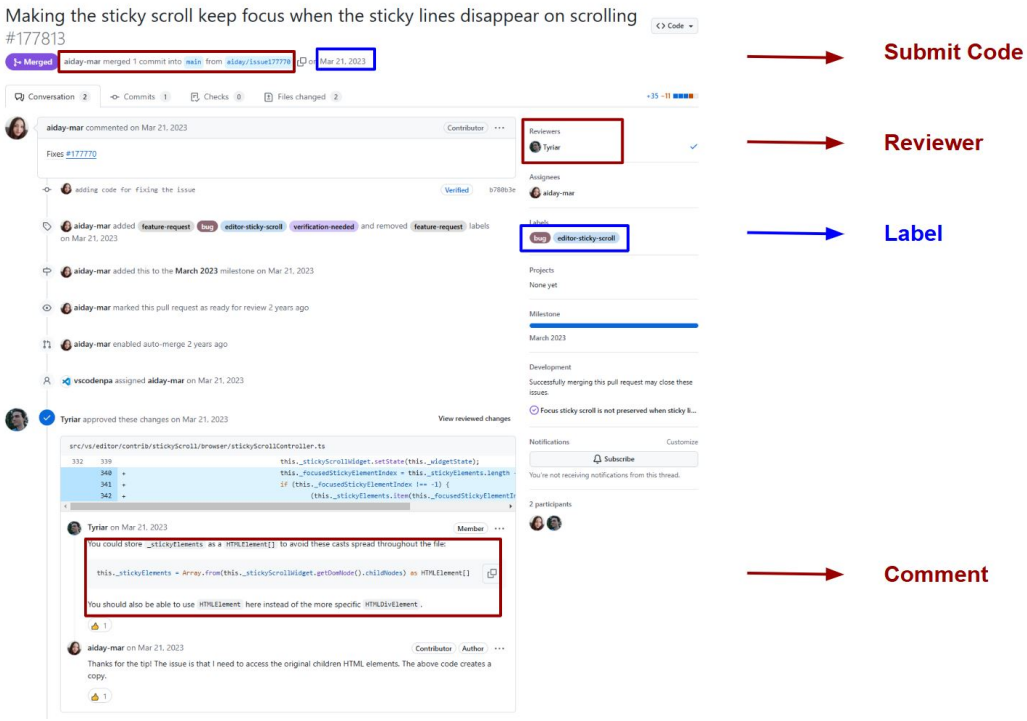
[Figure B1 about here.]

[Figure B2 about here.]

[Figure B3 about here.]

[Figure B4 about here.]

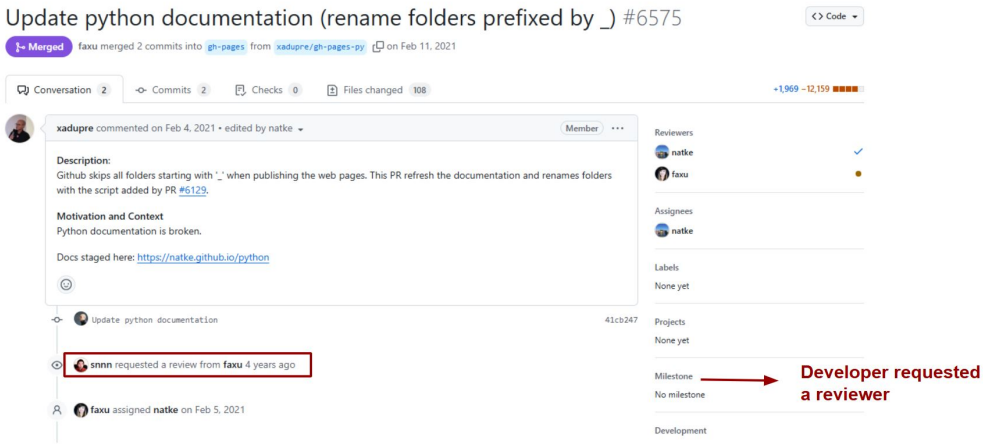
Figure B1: Case: Code Submission and Reviewer



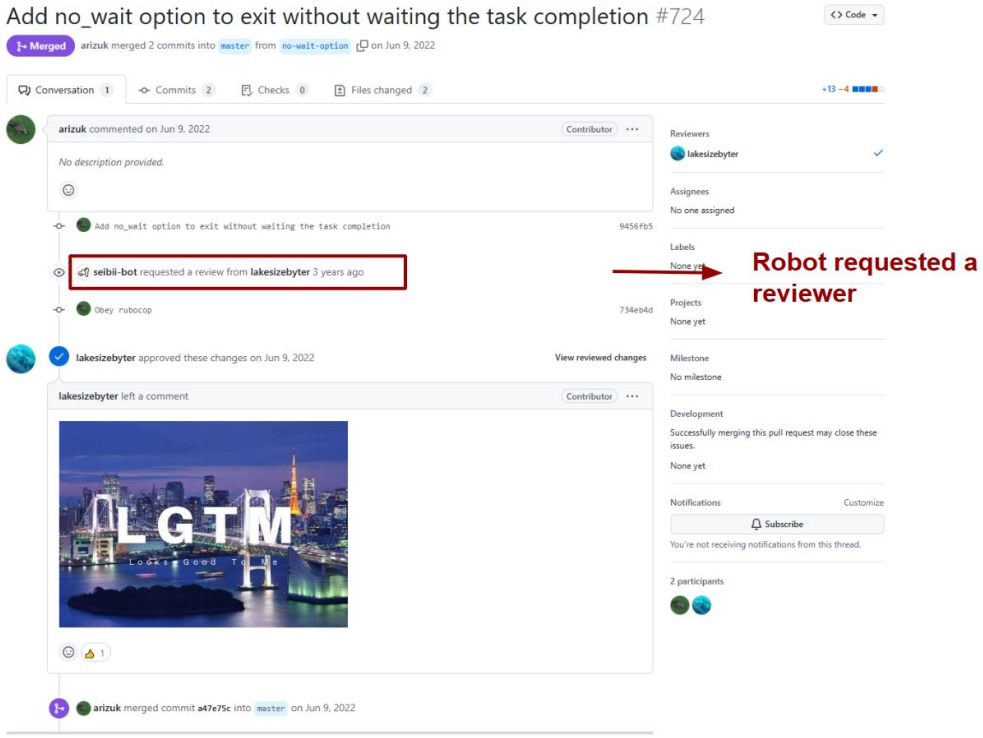
Notes: The example shows a typical Case (Pull Request) (#23804) with its associated review thread. Two distinct roles are highlighted: Reviewers (red box) who evaluate code changes and comments the reviewer gave.

Figure B2: Random Reviewer Example

(a) Review requested by Developer

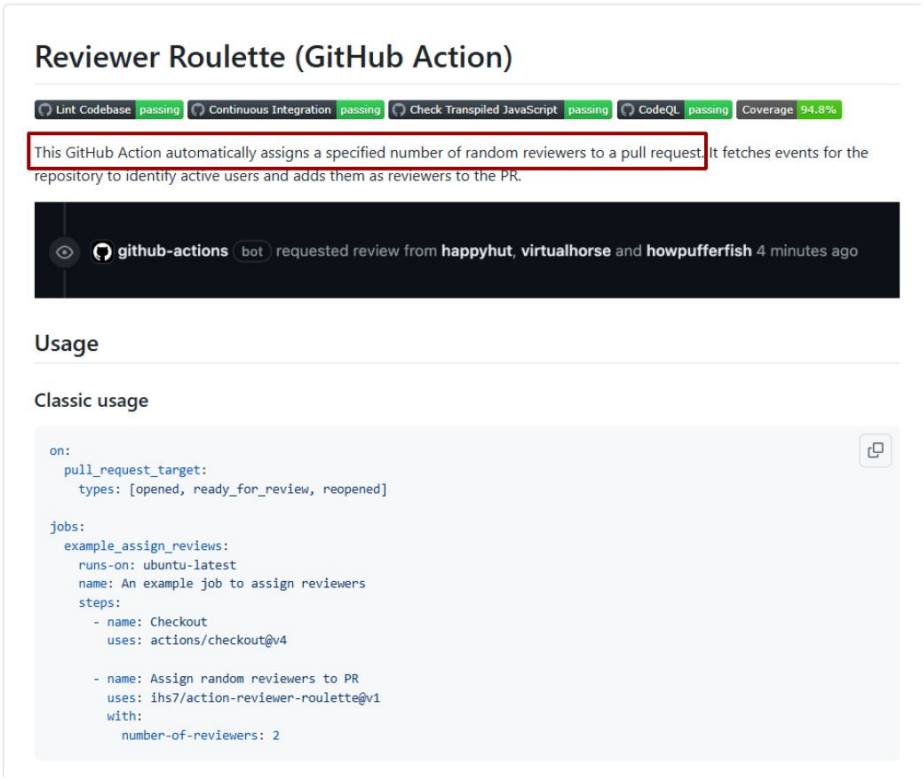


(b) Review requested by Robot(Auto Assign)



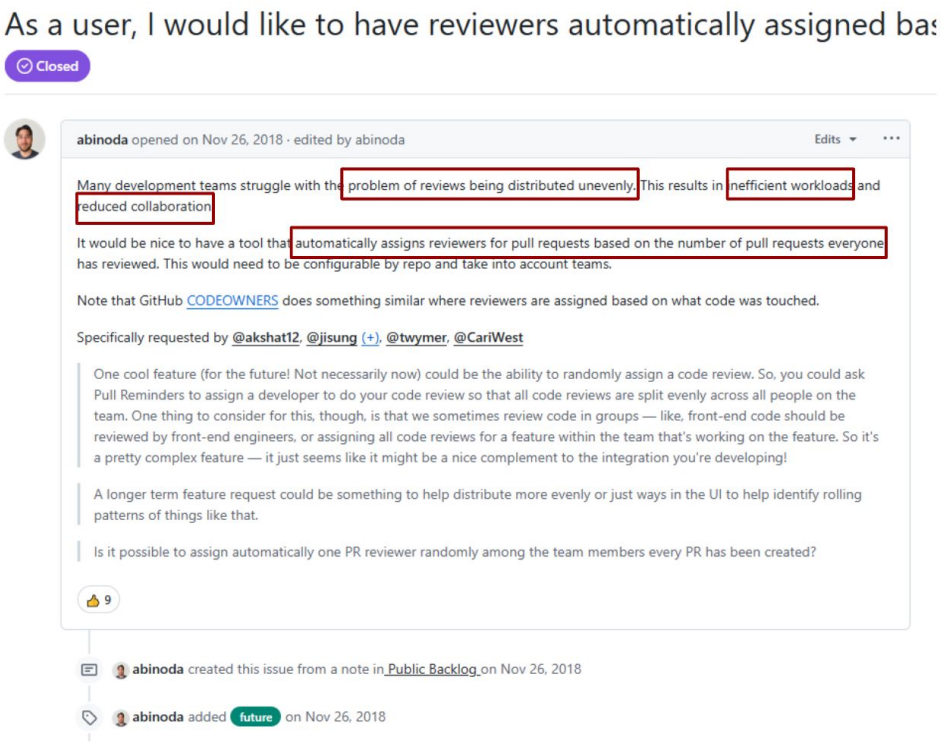
Notes: This graph illustrates two types of review requests. Figure B2a shows a developer requesting a review, while Figure B2b depicts a GitHub bot initiating a review request. Many teams adopt automated reviewer assignment to ensure a fair and unbiased distribution of review tasks. GitHub Bots request reviews using `CODEOWNERS`, `.github/auto_assign.yml` `.github/dependabot.yml`. The `CODEOWNERS` file assigns reviewers based on file ownership, while `auto_assign.yml` enables random reviewer selection, reducing workload imbalance and potential biases in the review process. Bots like `github-action[bot]` and `seibii-[bot]` use these features to randomly assign reviewers when generating pull requests, ensuring that code reviews are distributed efficiently and fairly across team members. Unfortunately, review requests are not visible in timelines and are only used to trigger webhooks. They are not emitted to the Event API and, therefore, are not logged in the GH Archive

Figure B3: GitHub Action for Automated Reviewer Assignment



Notes: This figure illustrates the Reviewer Roulette GitHub Action, which automatically assigns a specified number of random reviewers to a pull request. The action fetches events from the repository to identify active users and assigns them as reviewers. The top section highlights the successful execution of this action, with random reviewers such as `happyhut`, `virtualhorse`, and `howpufferfish` being assigned. Detailed information from GitHub Marketplace Action, accessed 2025-01-31

Figure B4: Reason for Automated Reviewer Assignment



Notes: The figure shows the discussion about the need to automatically assign reviewers based on metrics like the number of pull requests reviewed by each team member. Such automation ensures fair workload distribution and improved team collaboration. Original post, accessed 2025-01-31

Table B1: Share of Teams in Big Tech Firms adopting Random Reviewer Feature

Firm	Share	Firm	Share
Facebook	0.0888	Apple	0.0412
LinkedIn	0.0736	Uber	0.0399
Amazon	0.0642	Salesforce	0.0386
Airbnb	0.0620	Twitter	0.0309
Google	0.0587	OpenAI	0.0224
Sony	0.0571	Dropbox	0.0214
Netflix	0.0526	Oracle	0.0197
PayPal	0.0475	IBM	0.0101
Microsoft	0.0451	eBay	0.0087
Tencent	0.0415	Yahoo	0.0037
Spotify	0.0414	Snapchat	0.0000

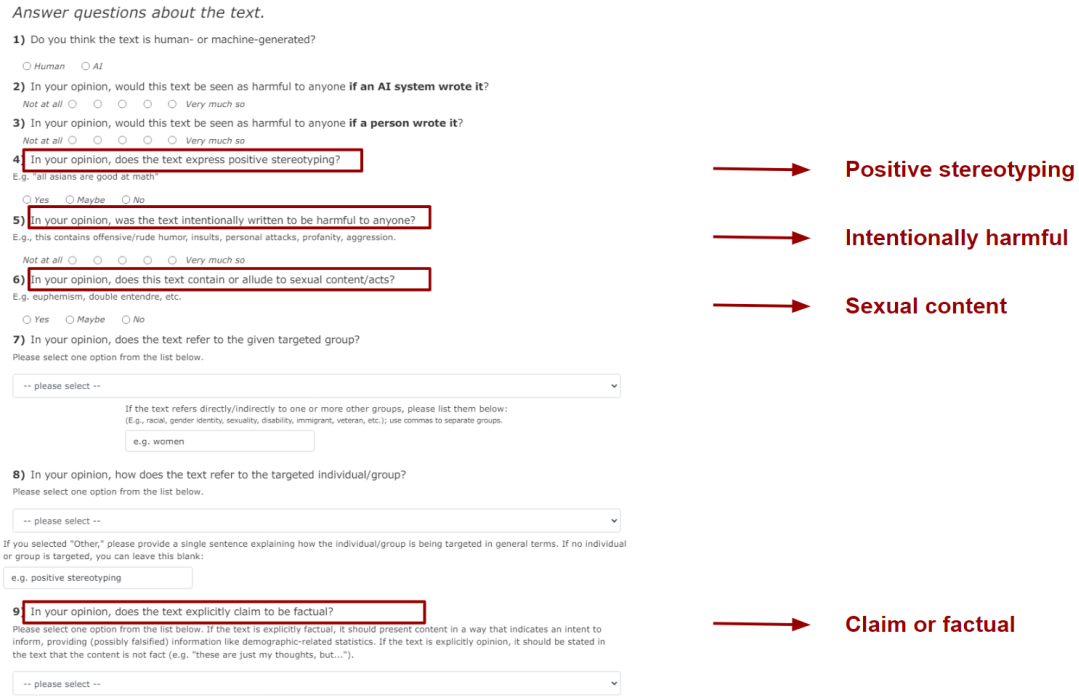
Notes: This table provides summary statistics for the share of teams in big tech firms adopting random reviewer features. There are 2 percentage points of teams adopting this feature (35,495 / 17,741,894).

C Linguistic Detection

[Table C1 about here.]

[Figure C1 about here.]

Figure C1: Toxigen



Notes: This figure shows how human annotators label toxicity.

D Additional Result

[Table D1 about here.]

[Table D2 about here.]

[Table D3 about here.]

[Table D4 about here.]

Table C1: Randomly Selected Comments by Toxicity and Sentiment

Toxigen	**Sentiment**	**Comment**	**Score(Toxigen,Sentiment)**
Non-toxic	Positive	look good to me	(0.9988, 0.9346)
Non-toxic	Negative	agreed, redundant	(0.9993, 0.8178)
Non-toxic	Negative	fails	(0.9989, 0.7578)
Non-toxic	Negative	why was this required? doesn't the inferred id?	(0.9978, 0.5954)
Non-toxic	Neutral	fixed it.. i removed all vendors packages	(0.9992, 0.7980)
Toxic	Positive	america, fxxk yeah !	(0.9791, 0.6319)
Toxic	Positive	sounds good, can't wait to restaple my ass on	(0.9702, 0.9645)
Toxic	Negative	they should be gone. not needed. gone now!	(0.9805 , 0.8817)
Toxic	Negative	you will have to answer you are talking nonsense	(0.8831, 0.9248)
Toxic	Neutral	dumps !	(0.9796, 0.5862)
Toxic	Neutral	since you started the call for more laziness in regards to the modal	(0.8207, 0.6226)

Notes: This table includes randomly selected comments from each category.

Table D1: Test of Randomization

	Pr(Toxicity) (1)	**Reviewer Toxicity** (2)
Male	0.002*	0.000
	(0.001)	(0.002)
Number of followers	-0.002	0.001
	(0.001)	(0.002)
Total activities in the past year	-0.004***	-0.004**
	(0.001)	(0.002)
Year of first recorded activity	0.005***	0.005*
	(0.001)	(0.002)
Has a biography on the profile	0.000	-0.003
	(0.002)	(0.004)
Has any label	0.016***	-0.006
	(0.005)	(0.005)
'Bug' related	-0.040***	-0.004
	(0.006)	(0.006)
'Python' related	0.025	0.020
	(0.027)	(0.056)
'JavaScript' related	0.064	0.070
	(0.057)	(0.079)
'Dependency' related	-0.051***	-0.001
	(0.011)	(0.012)
'Enhancement' related	0.023	0.009
	(0.012)	(0.009)
Joint F-test	0.000	0.121
Observations	2,794,134	2,794,134

Notes: This table reports reduced form results testing the random assignment of cases(pull request) to reviewers. Reviewer hostility is estimated using data from other cases assigned to reviewers following the procedure described in Section 5.1. Columns 1 and 2 report estimates from an OLS regression of toxic comments and reviewer hostility on variables listed and team × year fixed effects. Standard errors are clustered at the team level and reported in parentheses. The P-value reported at the bottom for an F-test of the joint significance of the variables listed in the row. $p < .10$, ** $p < .05$, *** $p < .01$.

Table D2: Test of Monotonicity

	Split I (Gender)			**Split II (Pre Activity)**		**Split III (Experience)**		**Split IV (Followers)**	
	Male	Female	Missing	Less	More	Less	More	Less	More
Reviewer Toxicity	0.6433***	0.6216***	0.7003***	0.6352***	0.6383***	0.6300***	0.6752***	0.6320***	0.6634***
	(0.0161)	(0.0316)	(0.0267)	(0.0180)	(0.0164)	(0.0149)	(0.0237)	(0.0165)	(0.0234)
F-Stat.	1,586.85	387.29	690.09	1,241.11	1,515.25	1,780.76	811.58	1,470.83	803.84
N	1,802,051	228,111	756,058	1,789,767	996,952	1,908,945	879,007	2,318,426	469,979

Notes: This table reports the first-stage coefficients and F-statistics from regressions of the reviewer's decision on the instrument across different sample splits. Columns 1-3 separate the sample by gender, while Columns 4-5 divide the sample by pre-week activity. Columns 6-7 split applicants based on whether their first active year on GitHub is above or below the mean. Finally, Columns 8-9 divide the sample by the number of followers.$p < .10$, ** $p < .05$, *** $p < .01$.

Table D3: Controlling for Reviewer Nontoxic-Negative or Positive Tendencies

	First Stage	**Reduced Form**		**IV**	
	(1) I_{i0}^{T}	(2) Ever	(3) Log	(4) Ever	(5) Log
A. Baseline Specification					
Reviewer Toxicity	0.631***	-0.004	-0.285***	-0.007	-0.449***
	(0.015)	(0.005)	(0.081)	(0.007)	(0.126)
B. Control for Reviewer NT-N					
Reviewer Toxicity	0.622***	-0.004	-0.186*	-0.007	-0.297*
	(0.015)	(0.005)	(0.084)	(0.008)	(0.132)
Reviewer NT-N	0.008***	-0.000	-0.098***	-0.000	-0.095***
	(0.001)	(0.001)	(0.016)	(0.001)	(0.017)
C. Control for Reviewer NT-P					
Reviewer Toxicity	0.630***	-0.005	-0.297***	-0.008	-0.468***
	(0.015)	(0.005)	(0.081)	(0.007)	(0.125)
Reviewer NT-P	-0.002***	-0.001	-0.039**	-0.001*	-0.039**
	(0.000)	(0.001)	(0.013)	(0.001)	(0.013)
F-Stat.	1782.994				
N	2766507	2766507	2715728	2766507	2715728

Notes: The outcome variable is binary of log of the average lines of code written 1–6 months. The results remain robust across other outcome variables as well. Reviewer tendencies are measured as same as reviewer hostility, as described in Section 5.1. A reviewer's NT-N tendency is defined as the average number of negative but non-toxic messages they have sent in other cases they handled. Similarly, a reviewer's NT-P tendency is measured as the average number of positive but non-toxic messages they have sent in other cases. Standard errors are clustered at the team level. Significance levels are indicated as follows: * $p < .10$, ** $p < .05$, *** $p < .01$.

Table D4: Controlling for Reviewer NT-N and NT-N Comments

	First Stage		**Reduced Form**			**IV**	
	(1) I_{i0}^{T}	(2) I_{i0}^{NT-N}	(3) Ever Activity	(4) Log Activity		(5) Ever Activity	(6) Log Activity
A. Baseline							
Reviewer Toxicity	0.631***		-0.004	-0.285***	I_{i0}^{T}	-0.007	-0.449***
	(0.015)		(0.005)	(0.081)		(0.007)	(0.126)
F-Stat.	1801.879						
B. Specification with toxic and NT-N comments							
Reviewer Toxicity	0.622***	0.302***	-0.004	-0.186**	I_{i0}^{T}	-0.006	-0.241*
	(0.015)	(0.020)	(0.005)	(0.084)		(0.008)	(0.136)
Reviewer NT-N	0.008***	0.805***	-0.000	-0.098***	I_{i0}^{NT-N}	-0.000	-0.118***
	(0.001)	(0.004)	(0.001)	(0.016)		(0.001)	(0.021)
Sanderson-Windmeijer F-Stat.	1349.112	2059.096					
Dependent mean	0.009	0.374	0.982	3.077		0.982	3.077
N	2766507	2766507	2766507	2715728		2766507	2715728