"Did You Miss My Comment or What?" Understanding Toxicity in Open Source Discussions

Courtney Miller,* Sophie Cohen,§ Daniel Klug,* Bogdan Vasilescu,* Christian Kästner*
*Carnegie Mellon University §Wesleyan University
{courtneymiller, dklug, vasilescu}@cmu.edu

ABSTRACT

Online toxicity is ubiquitous across the internet and its negative impact on the people and online communities it effects has been well documented. However, toxicity manifests differently on various platforms and toxicity in open source communities, while frequently discussed, is not well understood. We take a first stride at understanding the characteristics of open source toxicity to better inform future work designing effective intervention and detection methods. To this end, we curate a sample of 100 toxic GitHub issue discussions combining multiple search and sampling strategies. We then qualitatively analyze the sample to gain an understanding of the characteristics of open-source toxicity. We find that the prevalent forms of toxicity in open source differ from those observed on other platforms like Reddit or Wikipedia. We find some of the most prevalent forms of toxicity in open source are entitled, demanding, and arrogant comments from project users and insults arising from technical disagreements. In addition, not all toxicity was written by people external to the projects, project members were also common authors of toxicity. We also provide in-depth discussions about the implications of our findings including patterns that may be useful for detection work and subsequent questions for future work.

1 INTRODUCTION

There are many anecdotes like the tweet above, in blog posts, social media posts, podcasts, and at practitioner events—maintainers of open source software, often volunteering their time, talk openly about how sometimes interactions with others in open source can be toxic, rude, mean, or unkind, e.g., [3, 24, 44, 54, 108]. "Toxicity," defined here as "rude, disrespectful, or unreasonable language that is likely to make someone leave a discussion"[1] is a huge problem online [26]. Virtually all online platforms recognize the threat that toxicity, or the various types of behavior under its umbrella, pose on the health and safety of online communities. As a result, a number of prevention and mitigation policies and interventions have been proposed, including codes of conduct, moderation, counterspeech, shadow banning, or just-in-time guidance to comment authors.

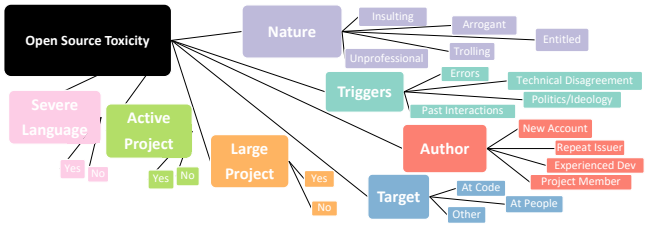

Figure 1: Identified characteristics of open source toxicity

Expectedly, open source communities are not immune to toxicity. While the term "toxicity" as defined above has only recently started being used in the open-source literature [17, 79, 86], the presence of behaviors "likely to make someone leave" have long been documented by researchers and practitioners in this space. For example, the Linux Kernel Mailing List is notorious for having discussions with a tone that "tends to discourage people from joining the community."[2] Generally, in open source the tone of project discussions is something newcomers pay attention to when deciding to join projects [76] and it can also act as a barrier to onboarding [93]. In an attempt to discourage abuse and harassment and to set norms of acceptable behavior many open source projects have adopted codes of conduct [55, 98].

Contributor disengagement, especially when precipitated, is of major concern in open source [46]. With many important open source projects being maintained by one or two volunteers [4], possible demotivation, burnout, and disengagement from the community can increase the risk of projects becoming abandoned or undermaintained. In turn, this can degrade software quality and security downstream, in the often lengthy software supply chains that those open source projects are part of. Toxicity is also a major threat to diversity and inclusion, as prior work has found that it can especially impact members of certain identity groups, particularly women [66], who are already severely underrepresented [2, 112].

While automated detectors of toxicity in open-source discussions, based on machine learning and various text-based features of the discussion comments, are starting to emerge [79], paving the way for more effective prevention, mitigation, and further study of toxicity in the future, the design of such detectors tends to lack a fundamental understanding of the nature of the phenomenon and its unique characteristics in open source. We present some of the key characteristics of open source toxicity found during our qualitative analysis in Figure 1. We argue that a more *fundamental understanding of when, how, and why toxicity occurs in open source* is needed. First, researchers have repeatedly shown that classifiers of

[1] Originally proposed by Google's project Jigsaw, this definition was used widely by Google in their toxicity detection research [25, 48] and by much subsequent research on machine-learning-based toxicity detection. The definition is also used by many online publishers and platforms, including The New York Times and Disqus.

[2] https://lwn.net/Articles/559061/

toxicity, sentiment, emotion, etc. rarely generalize well beyond the speci c contexts in which they have been developed [68, 86], even within the same general software engineering domain, which denotes an incomplete understanding of the underlying phenomena. Second, prior research on toxicity on other platforms (see Section 2), on the one hand, and the few emerging results and many anecdotes on toxicity in open source, on the other hand, suggest that toxicity can manifest quite di erently in open source discussions than it does elsewhere online. This makes it unclear which existing toxicity prevention and mitigation norms, practices, and interventions, if any, can be expected to succeed in open source.

In this paper, we address this gap by providing a thorough qualitative exploration and catalogue of toxicity in open source. First, we used a diverse sampling strategy followed by manual labeling to collect 100 toxic issues from open source projects hosted on GitHub . We study toxicity in issue threads since issues are one of the predominant places where developers can discuss the ideas, challenges, and goals of the development process. Issues are also a major channel for project maintainers to interact with their users, and the channel where many of the previous anecdotes of demanding, unreasonable, aggressive, or entitled interactions have originated. Second, we used qualitative analysis on these manually labeled toxic issues to discover common themes in the antecedents and characteristics of toxicity in open source.

Our analysis reveals several notable ways in which toxicity presents itself di erently in open source. Unlike some other platforms where the most frequent types of toxicity are hate speech or harassment, we nd entitlement, insults, and arrogance are among the most common types of toxicity in open source. We also learned that many of the ways projects address toxicity are closely connected to the GitHub interface itself and open source culture more broadly, such as locking issues as too heated or invoking a project's code of conduct. This work highlights the unique nature of toxicity in open source and provides a comparison to toxicity on other platforms. Our ndings can help support future work on the construction of e ective detection and intervention tools for open source toxicity.

2 STATE OF THE ART

Toxicity is often considered an umbrella term for various antisocial behaviors including trolling [42], aming [53], hate speech [21, 35, 87], harassment [26, 97, 102], and cyberbullying [51]. There is a long history of studying various forms of toxicity (e.g., bullying) and possible interventions (e.g., school councilors, moderators) in social sciences [70, 90]. More recently, toxicity has received a lot of attention in the context of online platforms [92], especially on social media [103]. Research has rmly established how toxic interactions can cause harms, both for people directly involved as well as for observers. For example, studies of the impact of cyberbullying on adolescents' mental health have found a signi cant relationship between cybervictimization and depression [50, 67, 104].

Within computer science, the focus has often been either (a) on automated detection, typically focused on speci c forms of toxicity such as hate speech [10, 21, 110], microaggressions [9], or cyberbullying [5, 84, 111], or (b) on HCI research for platforms like Wikipedia [1, 75, 80], Facebook [96], and Twitter [44, 57].

In this work, we focus on understanding toxicity in a speci c community: Open source developers and their public discussions on GitHub. As we will discuss, toxicity in open source may manifest di erently than in other online communities. We hope our ndings will inform subsequent work on open source toxicity detection and e ective intervention/community design.

Toxicity in Open Source. Many open-source maintainers have reported experiencing frequent toxic interactions with the very people who bene t from their often voluntary labor (e.g., being criticized, belittled, and insulted). For example, maintainers have shared anecdotes in blog posts [24, 45, 54], conference talks [83, 101], and podcasts [47], often explaining how they take steps to protect their communities or their own mental health, including quitting open source entirely. For example, in a widely shared 2017 blog post, maintainer Lawson describes how the complaints, questions, and requests for enhancement from users can feel like a constant stream of negativity which can be mentally and emotionally exhausting [54]. Another ex-open source community leader, when explaining why they quit, described how I had been told that I needed a `tough skin' to work in the community, and I needed to `not take it personally' when developers were abrasive during code review [39]. Toxicity in open source is often written o as a naturally occurring if not necessary facet of open source culture. The aforementioned community leader describes how When I complained about the toxic environment, I was told it was `tradition' to be blunt and rude in order to have a truly open dialogue.

The recent exodus from the Perl Foundation serves as a case study of the impacts of toxicity in open source [85]. Four high-ranking Perl community members stepped down due to community-related issues; one of them, when elected as community leader in April 2016, set the goal to make the mailing list a place on which we can have technical conversations without worrying about abusive language or behavior [107]; however in April 2021, he stepped down explaining how the chain of continuous bullying and hostility I've been receiving has caused him signi cant emotional distress [108]. The Linux community is another example that is notorious for experiencing similar challenges with toxicity [24, 28].

To curb toxicity, many open source projects have adopted code of conduct, a document describing expected and unacceptable behavior [55, 98]. Some projects have also experimented with bots that aim to automatically detect toxicity to respond to suspected toxic text with a precon gured message (e.g., Safe-Space[4] and Sentiment-Bot,[5] typically using a general-purpose toxicity detector, such as Perspective [48]), though adoption is rare.

Despite many anecdotes and plenty of discussions among practitioners, academic research on toxicity in open source is relatively sparse. One line of prior research touches on the negative e ects of toxicity and impoliteness on existing and potential contributors, and, by extension, project sustainability as a whole. Researchers have found evidence that impolite issue comments are associated with slower resolution times [23], that the unfriendliness of maintainers in issue and pull request comments is seen as a deterrent to newcomers [76, 94] and, more generally, that unhappiness and insu cient peer support increases the risk of disengagement [38, 62, 77]. Attracting and retaining contributors is

[4] https://github.com/charliegerard/safe-space
[5] https://github.com/behaviorbot/sentiment-bot

Table 1: Synopsis of toxicity of several popular online platforms

Platform	Common forms of toxicity	Harms of toxicity	Interventions against toxicity
Reddit	Inter-group con ict among subreddits, including coordinated attacks on other communities [20, 52]. A small portion of subreddits is responsible for the vast majority of inter-group con ict observed [52]. Di erent subcommunities face di erent amounts of trolling depending on their category [29]. Additionally the most common types of toxic elicitation comments on Reddit were (1) strong-toned, condescending; (2) sarcastic; (3) digressive; and against common sense/values [109].	The recipients of inter-group toxicity tend to participate less in said groups [52]. Additionally a survey of 15,000+ users found that 50% of respondents wouldn't recommend Reddit to others because of the hateful or o ensive content and community [82].	Subreddit moderators are considered a critical rst-line of defense against violations of community guidelines [49]. Reddit has also seen success with an anti-harassment policy and banning toxic subreddits [81]. According to Chandrasekharan et al. [11] more accounts than expected discontinued using the site; those that stayed drastically decreased their hate speech usage by at least 80%.
Wikipedia	Wikipedia trolls repeatedly and intentionally cause harm to both the community and encyclopedia [83]; they contribute unhelpful and nonconstructive remarks and attack users. Trolls usually work alone and anonymously. Wikipedia vandals cause time-intensive damage including deleting articles, using inappropriate usernames, and adding incorrect/irrelevant/policy-violating text to articles [75].	The vandalism of Wikipedia can be harmful to the broader community, causing misinformation and hundreds of millions of damaged views [75]. Trolling is a major source of friction. Experiencing harassment tends to decrease user participation in those communities [97].	Research has found communities participate in a collaborative vandal ghting process [38]. Furthermore, many researchers have developed techniques for automated detection of Wikipedia vandalism [74]. The Wikipedia Counter-Vandalism Unit describes counter-vandalism studies in depth [105].
Twitter	Online harassment is considered endemic to Twitter [39]; studies report hate speech, doxxing [58], cyberbullying, misogyny, racism, and Islamophobia as most prevalent [16]. In addition, restorms are a common occurrence on Twitter [53], where an account receives a sudden and intense in ux of negative attention [73]. Firestorms, mob mentality, trial by Twitter [59] are employed as mob vigilantism [16].	Sterner and Felmlee [95] found cyber aggression on Twitter is extensive and often extremely o ensive, with the potential for serious, deleterious consequences for its victims. A study of bullying detection on Twitter found that bullies post less, participate in fewer online communities, and are less popular than normal users [13].	Twitter deals with harassment through deletion, suspension, and warning of harassing accounts [58]. In addition they rely on authorized reporters or trusted aggers who have special privileges to identify and report inappropriate content on behalf of others [58].
Stack Over ow	A study of norm violations on Stack Over ow found that personal harassment, swearing, and other unwelcoming comments were the most frequently observed types of norm violations [14]. Community guidelines also request to avoid greetings and `thank you' in questions and articles are often downvoted or deleted for violating such community conventions [31]. Especially female community members fear harsh criticism [33].	Representation of minority groups is low [33] and rst-time users are turned away by harsh interactions [31]. A former Stack Over ow executive vice president admitted in a blog post that too many people experience Stack Over ow as a hostile or elitist place, especially newer coders, women, people of color, and others in marginalized groups [41]. (http://stackover ow.blog/2018/04/26/).	Stack Over ow uses two main tools to address violations of their code of conduct; (1) respected community members who act as moderators; (2) bots that detect comments with toxic language, such ciently severe comments are reported to the moderators [14]. Mentoring mechanisms for new users have been explored to avoid behaviors that might receive harsh responses [31].
GitHub	Prior work: Cheriyan et al. [15] found o ensive language to be less prevalent on GitHub compared to other platforms. Our results: The most common types of toxicity we nd are insulting, entitled, and arrogant comments. A common trope is entitled and arrogant tool users who are critical of the project and have unreasonable expectations. Project members also write toxic comments but typically in response to demands.	Prior work: Open-source contributors deciding whether to work on a project consider the friendliness of project correspondence [76]. Some newcomers disengage due to negative interactions [94]. Some maintainers report stress and burnout [79]. Our results: Maintainers often invest substantial time to respond to toxic comments.	Prior work: GitHub provides moderation tools, but no automation. Third-party bots for toxicity detection exist, but are rarely adopted. Our results: Responses are manual. The most frequently used tools used in response to toxicity in our sample were closing issues, locking issues as too heated, and deleting issues

a major challenge in open source [19], especially when it comes to underrepresented groups [7].

Another recent line of research focuses on detection of toxicity in open source and related issues. Most closely related, Raman et al. [79] proposed a classi er to automatically detect toxicity on GitHub , though that classi er produces a high rate of false positives (as we will nd in Sec. 3 and as Sarker et al. [86] noted as well). Recently, Cheriyan et al. [15] also proposed a learning-based detector of toxic language in online software communities, although they consider a narrower de nition (only swearing and profanity). Egelman et al. [27] built a detector for the related concept of pushback in code reviews in a corporate setting, with a similar motivation to curb pushback before it ultimately results in developers abandoning projects More broadly, researchers have also been exploring the role of emotions in software engineering [40, 63, 64, 71], including means to detect a range of emotions like anger or frustration [32, 34], though not for our notion of toxicity.

However, even though multiple studies have focused on automated detection, their success has been mixed. Already prior researchers have hinted that toxicity in open source manifests differently than elsewhere online and that more domain adaptation for detection and interventions is needed. We go one step further and argue that what is missing primarily is a deep understanding of the fundamental nature of toxicity in open source: what are its common forms, what scenarios it occurs in, who are its originators, and how these characteristics compare to other online platforms where toxicity occurs. Only given such understanding can we begin to design the most e ective detection, prevention, and mitigation strategies. Yet some qualitative work is still missing. The only exception is the contemporary study by Ferreira et al. [28] of emails from the Linux Kernel Mailing List that were associated with rejected code changes. The authors found that uncivil comments occur frequently among these, most commonly in the form of frustration, name calling, and impatience. Our work adds to this emerging literature by expanding the scope beyond the Linux Kernel Mailing List which is known as a contentious project and by looking more broadly at toxicity and issue discussions on GitHub.

Toxicity on Other Platforms. Most past research on toxicity in

Figure 2: Flowchart of methodology for data collection and analysis process

online communities has been community-speci c. While rarely made explicit, this seems to be either due to data availability or speci c characteristics of toxicity on certain platforms. For example, Reddit has a phenomenon called*brigading* in which users from one subcommunity on the site post antagonistically in another community in a coordinated fashion, overwhelming that community and its moderators [20, 52]. In addition, Reddit relies on community moderation by volunteers and o ers*shadow banning* as a key tool that allows moderators to hide a user's content from a subcommunity without making this transparent to that user. It is not obvious that studies on toxicity and interventions in Reddit generalize to other online communities, which do not seem to face brigading (widely) and do not have shadow banning as a tool, including GitHub .

To give an overview of the diversity of toxicity observed across platforms, resulting harms, and possible interventions attempted, we summarize representative studies of 4 popular online platforms in Table 1. It becomes clear that Reddit is not an outlier, but that all platforms di er in substantial ways. We therefore argue that all interventions should be tailored to the speci cs of a platform and to that end, it becomes necessary to rst understand characteristics of toxicity on a given platform which we do for GitHub in this paper.

3 RESEARCH DESIGN

To understand toxicity in open source, we curate a sample of existing toxic issues and qualitatively analyze them (cf. Fig. 2).

Considerations for Ethical Research Design. Toxicity in open source is a challenging topic to study. While some developers openly speak about their experience with toxicity (and associated stress and burnout), those discussions tend to be generic, abstracting from concrete instances. In previous research, we tried contacting developers who publicly spoke about toxicity, asking for examples, but they largely preferred not to discuss concrete instances or had already deleted them.

Toxic comments are usually unpleasant interactions, so we intentionally discarded research designs that relied on interviewing or surveying developers, especially those who had not already publicly talked about their experiences with toxicity, since it could bring back unpleasant memories and cause additional stress, in addition to the stress that the toxic interactions may have already caused in the rst place. In line with the Belmont Report's ethical principle of Bene cence for human subject research [30], we considered potential risks to be too high for an initial study on toxicity. Instead, we decided to work entirely with public artifacts (existing issue discussions and public information about participants and projects) without contacting participants.

Note that there are inherent tradeo s in our decision (as in all research design): Working with public artifacts allows us to sample more deliberately and analyze toxic issues as project outsiders perceive them, but our results are necessarily based on external observations and interpretations. For example, while we can usually gather enough context to understand the trigger of toxicity and understand the roles of participants, we cannot judge the perceived harm caused by toxicity in most cases.

Curating a Sample of Toxic Issues. We curated a dataset of toxic issues from which we sample for our subsequent qualitative analysis. Since toxicity is relatively rare (Raman et al. estimate 6 in 1000 issues to be toxic [79], likely an overestimate, see below), analyzing a random sample of GitHub issues to nd su cient toxic ones is not feasible. However, since we do not yet have a highly accurate and unbiased detector for all forms of toxicity, any search strategy we could use to identify toxic issues may bias the kind of issues we detect. This is a known limitation not only of our research, but of most similar research on detecting hate speech and related phenomena [78].

To ensure diversity (though not necessarily representativeness), we combine four di erent search strategies in issue comments posted before June 2020. We then stratify a nal sample for qualitative analysis from results of these strategies. While each strategy is biased and noisy, they generally pick up on di erent signals, and thus provide a more diverse picture. The four strategies are:

- **Language-based toxicity detection.** We use our our toxicity detector from prior work to classify (1) all 13.4 million issues and (2) all 14.5 million issue comments posted on GitHub from March to May 2020 [79], mined from GHTorrent [37]. Our classi er is based on two o -the-shelf machine learning models for toxicity and politeness developed in other domains, noticeably biased toward text with strong language. We consider only issues in English as detected by the same classi er.
- **Code of conduct.** In the same issue comments, we perform a textual search for `code of conduct` which is often invoked when reacting to toxic comments.
- **Locked issues.** As in our previous work [79], we identify GitHub issues that have been locked as `too heated' before June 2020. We use the GitHub API to identify such locked issues among the 200 most recent issues in over 600,000 GitHub projects (all projects with 5 or more stars).
- **Deleted issues.** In addition, we recovered many deleted issues those issues recorded in GHTorrent but no longer available through the GitHub API for the same 600,000 projects.

Since all strategies turned up many issues that were not toxic, we then performed a manual labeling step on a random subset of all candidate toxic issues. To facilitate this labeling, we built a simple internal tool that showed us one issue at a time and provided buttons to label comments. To assure reliability of our labels, we asked multiple authors to independently label issues; inter-rater reliability as measured by Cohen's unweighted kappa coe cient was 0.82 for 149 comments with two independent labels and Fleiss' kappa

Table 2: Issues detected and labeled with di erent strategies to build our sample, with the number of issues labeled as toxic and the false positive rate (FPR) of each strategy.

Strategy	#detect.	#labeled	#toxic	FPR
Toxicity model (issue)	168,293	494	82	.83
Toxicity model (comment)	42,911	225	69	.69
Code of conduct	668	179	33	.82
Locked issue (too heated)	2,530	165	40	.76
Deleted issue	60,959	1,644	29	.98

coe cient was 0.72 for 43 comments with three independent labels typically interpreted as substantial to near-perfect agreement [61]. Similar to prior observations on labeling pushback [27], this supports that external observers are very likely to agree on whether a given issue comment is toxic or not, justifying our manual labeling strategy by researchers not involved in the original discussion.

In Table 2, we report how many issues we labeled per strategy (some issues were detected by multiple strategies), and how many of those we labeled as toxic.[6] Note, that we distinguish between issues where the language-based classi er detects toxicity in the initial issue posting and those where it detects toxicity in subsequent comments. It shows that all strategies have high rates of false positives; we argue that much better detection strategies are needed before they can be deployed as mitigation strategies, and hope that this research can help to build better detectors.

We assemble our nal dataset of 100 toxic issues as a strati ed sample from issues manually labeled as toxic. We pick 20 random issues from each group in Table 2, discarding duplicates (when an issue is identi ed by multiple strategies) and limiting the sample to at most two toxic issues per repository to maintain project diversity.

Qualitative Analysis of Sampled Issues. To understand the important characteristics of open source toxicity, we qualitatively analyzed the 100 issues in our sample and their context, using thematic analysis [18], following the trustworthiness criteria by Lincoln and Guba [56] as demonstrated by Nowell et al. [69]. As we describe next, this was an iterative and re ective process, constantly moving back and forth between stages of engagement with the data, coding, memoing, searching for themes, and re ning, as recommended during qualitative analysis [18].

Overall, our analysis consisted of several phases. We started by immersing ourselves in the data, carefully reading issue threads to understand the problems discussed, the project context, and the relationship of the authors of toxic comments with those projects as well as their past public activities (including past issues) on GitHub. This typically took 15 to 30 minutes per issue, was conducted in groups of two or three researchers, and involved exploring, besides the issue threads themselves, also the project homepages and user pro le pages of the discussants. As we were engaging with the data, we kept posing sensitizing questions regarding what we were observing (what, who, how, where, why, what for, etc.) and, guided by these, generated an initial set of codes to describe the toxic interactions through an otherwise inductive, data-driven process [43]. We also wrote down brief analytic memos [61] summarizing emergent patterns and the possible connections among the codes.

After analyzing 35 issues this way, we paused to sort and collate the codes we had assigned so far into a coding manual with detailed de nitions and examples. In the manual we organized the codes into higher-level categories following the sensitising questions above, to cover the key characteristics identi ed up to this point nature of toxicity, nature of the comment, language severity, triggers, authors, position in discussion, project size and domain, resolution, and identi able harms. We then revisited all 35 issues using focused coding [22] to make sure our codes were applied consistently. A single researcher then analyzed and coded the remaining 65 issues in the same way, involving other researchers for di cult and ambiguous cases, and extending the coding frame iteratively for new observations, when needed.

For the interpretive categories,[7] in particular those related to the nature of the toxicity, we then searched for themes using card sorting [88]. This involved printing each issue discussion and sorting and resorting them into di erent themes, discussing group boundaries and subgroups, again in groups of two or three researchers.

Finally, we systematically searched for relationships between the emerging themes across combinations of all nine categories of our coding frame, further exploring observations we had written down in our analytic memos, e.g., whether toxicity by experienced developers tends to use less severe language than toxicity by new accounts. We used exploratory data analysis and visualization to help with this process, like those in Figure 3.

Reporting and Data Sharing. Throughout the following result sections, we refer to the issues using the identi ers I1 I100 and illustrate our ndings with slightly sanitized quotes. Note that while we did not observe any examples of overt hate speech, examples contain insults, entitlement, and personal criticisms, sometimes with explicit wording. Even though all issue discussions were public, we intentionally do not include direct links to them (1) to protect the parties involved and (2) because some of them have since been deleted. We will share our replication package with links and raw summaries only upon request for research purposes (available to reviewers on HotCRP).

Threats to Validity and Credibility. Our research su ers from the usual threats to be expected for this kind of research design. Qualitative analysis allows us to engage in depth with individual issue discussions in ways that is di cult to automate, but we cannot generalize beyond the sampled issues without further validation. As discussed above, this was unavoidable it is not practically feasible to draw a random sample of toxic issues. However, our strati ed sample is still large and diverse enough to uncover what we expect are common forms of toxicity across GitHub in general. Importantly, each strategies we use to identify potentially toxic issues has unavoidable biases that overlap with criteria we subsequently analyze, however di erent strategies have di erent biases. For example, we detect deleted and locked issues only in the 600,000 most-starred projects, hence not nding issues in the smallest projects, whereas detection based on linguistics and code of conduct analyze comments within a 3-month window. In contrast, the former are not biased toward speci c words as the latter are. All this supports diversity in our sample, but prevents statistical generalization.

[6] Based on these numbers, we roughly extrapolate that there are at least 400 toxic issues each day on GitHub; or at least 1.3 toxic comments per 1000 comments.

[7] Some categories are not interpretive, e.g., position in discussion and author a liation.

Figure 3: Comparing the nature of toxicity to the severity of the language (left) and the target of the toxicity (right), and analyzing the overlap between insulting, entitled and arrogant comments (bottom).

Regarding construct validity, we used the same de nition of toxicity that was used by recent related work including research papers on toxicity and toxicity detecting APIs, and we assured inter-rater reliability during labeling. However, toxicity is a complex topic and di erent people may perceive di erent text as toxic, possibly based on context that is not shared with everybody. As explained, we intentionally did not ask the original authors or recipients for con rmation. As all qualitative research, our analysis relies on some judgement.

4 TOXICITY OBSERVED ON GITHUB

During our analysis we observed a range of forms of toxicity on GitHub . We start with a discussion of the nature, severity, and target of toxicity, as these are some distinctive characteristics compared to other platforms (cf. Sec. 2). We refer to di erent kinds of toxicity (e.g., trolling vs doxxing) as the *nature* of toxicity. We observed many insults, some trolling, entitlement, and arrogance, and several comments that were not overtly toxic but were certainly unprofessional. Issue discussions do not always t neatly into a single category, for 38 issues we assign multiple. For example, a comment may be entitled and insulting at the same time. Overlap is particular common particularly among insulting, entitled, and arrogant comments, as shown in Figure 3 (bottom). While toxic comments are sometimes followed by other toxic comments (see Sec. 8), here we consider only the rst toxic comment in an issue discussion. We consider language to be *severe* if it contains expletives (e.g., cursing, swearing), which was the case for about half of our sample. Finally, we consider whether toxic comments were addressed at people or the project in general as the *target* of toxicity, with half of our sample containing toxicity targeted at speci c people.

Insulting. Over half of our sample contained insults (55 cases), i.e., disrespectful or scornful expressions, often using curse words or intentionally o ensive language. Toxic insulting comments tend to be targeted at people rather than at the code itself (Fig. 3; 37 of 55 insults in our sample). A frequent example of this category is a user lashing out at the project members in frustration at a problem with the code itself or friction during the debugging process. For example a user of a GUI crypto wallet with a built-in crypto miner noticed the presence of the miner and interpreted it as malware (a misunderstanding, the presence, deactivated by default, was mentioned as intentional feature in the readme). The user threw explicit curse words at the maintainers of the project and accused them of being criminal crooks for trying to infect other computers with malware (I12).

The insults are often written in an aggressive or emotionally charged manner or are accompanied by criticisms of the project itself. For example, a project member was unhappy with the colors of a project, reporting colors are horrible for [...], just look at this s**t (I23). Even after a contributor provided a link to the documentation, the user remained unsatis ed and unapologetic.

While there are strong insults with explicit curse words, fewer use severe language than do not (Fig. 3; 24 of 55 insults).

Entitled. Entitled comments are a frequent source of toxicity in our sample (25 issues in our sample), a form of toxicity not previously observed as a common problem on other platforms. Entitled comments make demands of people or projects as if the author had an expectation due to a contractual relationship or payment. Examples include complaining that software does not meet the author's quality standards, features are not added quickly enough, insisting their requests are given priority, or demanding bugs be resolved quickly. It often contains severe language (Fig. 3), this form of entitlement is often perceived (and called out) as toxic by maintainers.

In multiple cases, the user was dissatis ed with some aspect of the initial response to their issue, and continued the discussion or raised demands when the maintainers considered it settled. For example, a user complained about the language used in documentation, several contributors had a discussion, made changes, and closed the issue, upon which the original user, still unhappy, opened a new issue saying le an issue, maintainers close, reopen, again close - whilst ignoring the essence of the issue (I43). Similarly, a user, upon being told that their suggestion was based on a misunderstanding of the project, began aggressively criticizing the contributor for how they addressed the issue, saying like just add the avor text or show me how to or something. Don't just fu**ing close people's tickets they would like some hel (I37).

Entitlement is commonly targeted at people (Fig. 3), making demands of what individuals should do or insulting them for not doing what they wanted them to do or do it fast enough.

Arrogant. Arrogance was another common form of toxicity (21 issues in our sample) that caused con icts in some discussions. We consider comments as *arrogant* when the author imposes their view on others from a position of perceived authority or superiority (earned or not) and demands that others act as recommended (in contrast to entitled comments making demands based on some expectation of product or service quality).

The vast majority of arrogant comments in our sample were directed at speci c people. They tend to be triggered by technical disagreements, where the author speaks down to others about their opinion regarding the technical debate at hand. For example, during a discussion on an application repository about the app being locked in portrait mode, there was a debate about whether it was legally required for accessibility reasons. One of the users in the discussion was unfamiliar with some of the legislation being discussed and asked for more information, a second user responded saying ever hear about [standard]? A baseline for developers. Use G gle. (I48).

Trolling. Trolling is a form of toxicity also often seen on other

Figure 4: Comparing the trigger of toxicity with the nature and author.

platforms, where users simply engage in destructive discussions. We observed 17 trolling comments in our sample. While trolling comments tend to have more severe language and tend to target the project itself, they seem to have a shorter life span. For example, a user was generally unhappy with a project and wrote *Worst. App. Ever. Please make it not the worst app ever. Thanks* (I2), followed by a pull request that deleted all the code in the repo; after the maintainer closed the issue, the user responded *Merge my PR damnit* and nothing else happened.

Unprofessional. Finally, there are many comments (23 in our sample) that we categorize as *unprofessional*, which are not overtly toxic and probably not intended to be toxic, but that nonetheless can create an unwelcoming, impolite, or unprofessional atmosphere that may be perceived by others as toxic. Examples include self-directed pejorative terms (e.g. *It seems like I have been acting like a re**rd. Sorry. [...]* (I19)), self-deprecating humor, and jokes and puns with explicit vocabulary or terms broadly perceived as politically incorrect or unacceptable in a professional setting. We suspect that the unprofessional comments may make some participants or observers uncomfortable, to the point where they may discourage others from joining the discussion or the community [23, 76].

5 TRIGGERS OF TOXICITY

With *triggers* we refer to reasons why somebody may have posted a toxic comment, e.g., in response to a technical problem or a disagreement with another participant, as best we were able to infer from the content and context of the toxic comment, which often refers to a speci c complaint or con ict. In this context, it often differs whether a discussion is opened directly with a toxic comment or whether a toxic comment is posted in response to an evolving discussion, which we discuss as position .

Failed Use of Tool/Code or Error Message. Many toxic comments (34 in our sample) report trouble with using the software, often including a more or less vague description of the problem or an error message. The author is usually visibly upset about not being able to use the tool, often complaining about wasted time.

Some comments actually report the problem in some detail to help the project or receive help with their immediate problem, but still include toxicity, typically expressing frustration. For example, as one such user of a popular library puts it, *just tried reinstalling your buggy, sh**ty software for the third time. Maybe you guys can get one that works right and stick to it without changing it all the time* (I78). Yet, in other cases, users simply vent about problems without seeking help or any attempt to provide constructive feedback to the project.

Issues raised in response to problems are not necessarily toxic from the start. In several cases toxicity emerged later from discussions, when the maintainers did not immediately respond or solve the problem with the information provided (12 out of 29 cases in our sample). In some cases, the users respond with toxic messages when asked for more information or asked to follow the issue template, for example *Yeah, not really sorry i'm lazy, and it's more to help you then me. It's simple to understand: [...]. don't need a ret**ded format to understand that! thanks* (I93).

As shown in Figure 4, toxicity triggered by failed tool use is often entitled, insulting, unprofessional, or just trolling (destructive complaint without any intention to help, e.g. *It doesn't work. F*** this* (I1)).

Technical Disagreement. We frequently observed toxicity arising (Fig. 4, 22 issues in our sample, particularly arrogant comments) when users had di erent views on some technical component of the project, bug x, feature, or other technical decision. For example, a user of a tool was arguing with the project owner that Windows XP should be supported. After technical back and forth the user eventually insults the maintainer personally, saying *Your [project] is good, seems you're a good programmer, but as a sysadmin/engineer you're... er... well, `not so good`* (I49).

Politics/Ideology. We fairly frequently (31 issues in our sample) observed toxicity arising over politics or ideology di erences e.g., referring to speci c beliefs about open source culture, process generally, or the involvement of speci c companies (especially Microsoft was a frequent target in our sample). These are often associated with insulting, entitled, and trolling comments (Fig. 4). For example, a user wrote a hostile issue in a Microsoft project titled *WHY!??!?!!?!?!?* which simply said *Revenue. F**k you guys* (I28). These kind of comments could be either opening comments (18 of 31 in our sample) or be posted in response to a discussion (13 of 31 in our sample).

Past Interactions. Finally, we observed several cases (6 issues in our sample) where toxic comments were posted that referred to past interactions of the author with the project, without continuing to discuss the previous technical issue, but shifting to personal attacks or complains or meta discussions about process. For example, a user was unsatis ed with the response time on an existing issue so they created a new one asking *did you miss my comment or what?* (I29). These comments were often posted in a new issue after the old one was not answered or closed, and they occur often in the opening comment of the new issue (4 of 6 in our sample).

6 AUTHORS OF TOXICITY

Issue discussions are a central place where external users of a project communicate with project members, where external users may help other users, and where project members may have public

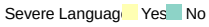

Figure 5: Comparing author and severity of the toxicity

technical discussions among themselves[8]. We observed that the authors of toxic comments have di erent roles.

Through our analysis, we identi ed four main types of authors. First, we distinguish between project members and authors external to the project. We consider as project members all contributors to the project, which includes all GitHub accounts that previously contributed code to the repository (e.g., including casual contributors who only submitted a handful of pull requests). This view of project members is consistent with perceptions of who constitutes project members in an open source community [99]. Second, among nonmembers, we distinguish new accounts (often anonymous), accounts that have repeatedly opened issues but have little or no own public development activity on GitHub (termed repeat issue reporters; users of open source, who do not contribute themselves), and accounts of experienced open-source contributors with many public code contributions to other projects on GitHub. In addition, 12 toxic comments were posted by deleted accounts hence we do not have enough context to classify them.

New Account. We observed 13 toxic comments in our sample from accounts that have no or at most one prior activities on GitHub. These accounts are seemingly created just for the purpose of posting an issue; they typically have no identi able information in their pro le and act anonymously. New accounts engage in anonymous trolling but also all other kinds of toxicity (insults, entitled, unprofessional). The majority of toxic messages of new accounts seem triggered by the failed use of a tool, often seemingly from the perspective of end users with limited technical knowledge. Toxicity from new accounts is often observed right in the opening message of the issue and tends to use severe language (Fig. 5). For example, a new user was trying to download an application but was having issues and wrote an issue titled Cant even install the fu**ing app where they complained that they could not nd the download, upon which another user pointed them to the project's release page (I81).

Repeat Issue Reporter. We observed 39 toxic comments in our sample that were authored by accounts that had previously opened other issues, but that had no own public code contributions. These accounts typically request help through multiple issues, often across multiple projects, but rarely contribute beyond writing issues.

Toxic comments from repeat issue reporters tend to appear in large and active projects, and tend to be triggered by the failed use of a tool or political/ideological disagreements (Fig. 4). Their authors also frequently write comments that are insulting, arrogant, trolling, or entitled, often with severe language (Fig. 5). For example, a 6-year old account with clear name, pro le picture, and contact email has created hundreds of issues over the years, before posting an issue sh**ty package (with no context or further content) to a mid-sized repository of a web UI component (I26).

Experienced Contributors. We observed 16 toxic comments authored by experienced open-source contributors in projects to which they have not contributed code previously. One might expect experienced open-source contributors to have more empathy for other open-source maintainers. Experienced developers tended to use less severe language (Fig. 5), but they participate in all forms of toxicity, including trolling, within our sample. For example, when an experienced contributor was upset that a new package manager update did not include Python 2, a project member responded with a workaround, to which the author then responded and I recommend you quit! There are many more where python2 is used [...] and you deleted it from the repository. Do you think at all with your head or do you have a hamburger head pla (I82).

Project Member. Finally, many toxic comments were authored by members of the project (19 in our sample), usually when replying to issue discussions. Toxicity from project members disproportionately occurs in smaller projects in reaction to a demand, complaint, or perceived a ront from another user (which we did not consider toxic according to our de nition). Project member toxicity also tends to be less severe (Fig. 5), mostly unprofessional comments or insultstargeted at code, including occurrences of self-directed pejorative terms. A common scenario is that external users have high demands and expectations on the project members' time, and in response the project members can be dismissive in ways that can be perceived as toxic. However, project members tend to not engage in `unprovoked' attacks. For example, during a ght about a naming con ict between two libraries a project member defended the project by arguing you can be mad all you want, but let's be realistic here... this project you're ghting for so passionately, doesn't have as many stars as I have thumbs down for telling you that you're being ridiculous (I84). Out of the 19 toxic comments written by project members, 4 occurred in projects that had a code of conduct either linked in their README or on their project website.

7 PROJECT CHARACTERISTICS

We observed project characteristics that may associate with toxicity, especially popularity and domain, which could point to di erent processes and cultural norms (as observed previously, e.g., for game vs application engineers [5] and between software ecosystems [6]).

Project Size. We found that the vast majority of toxic comments were written in popular repositories[9] with high levels of activity. However, 17 comments were from small and inactive projects, including what appeared to be homework assignments.

In less popular projects, the nature of toxicity is often insulting, trolling, or unprofessional, mostly directly in the opening comment, but we found no entitled comments, possibly because users have lower expectations in the rst place. Several toxic comments in small projects appeared to be trolling or jokes among friends, e.g, Dear Mr. [project owner name], Could you perhaps please get your s**t together and reincorporate the brilliant switch statement once

[8] https://guides.github.com/features/issues/

[9] We used a threshold of 20 stars for our labels, but classi cation usually agrees on many measures of activity and attention, like total comments, recent commits, number of issues, or downloads, where available.

again, bi**h. XoXo, [author username]" (I10).

Project Domain. Toxicity occurred often in projects we consider as libraries or end-user focused applications, which likely also are the most common projects on GitHub. In contrast, discussions in gaming projects were often reported by the used language-based toxicity detector (cf. Sec. 3), but actual toxicity in gaming project made up only a small fraction of toxic comments (13 in our dataset). However, toxicity in projects related to gaming and to mobile apps tends to use more severe language, e.g., more cursing, in our dataset. Projects targeted at end users had a higher proportion of toxic comments directly in the opening issue, rather than evolving from the discussion.

8 AFTER TOXICITY: HARMS & REACTIONS

We observed a range of reactions from project maintainers and others who publicly react to toxic comments in open source, which may help understand possible mitigation strategies and opportunities for automated support. Their reactions also give us some, albeit very limited, insight into potential harms caused by the toxic comments.

Tools to curb discussions. GitHub currently o ers several mechanisms to curb discussions and sanitize toxic comments after the fact, many of which were often used in our sample, often in combination:

- Closing issues: Maintainers can mark an issue as closed, which removes them from the list of open issues shown in the repository. Users can still comment on closed issues. This was used for 45 issues in our sample.
- Locking issues: Maintainers can lock an issue (optionally selecting a reason) to limit further commenting to maintainers only. This was used for 26 issues in our sample, though we sampled based on the lock reason `too heated and hence cannot generalize frequency in practice.
- Deleting issues: The original author of the issue and maintainers of the repository can delete the entire issue. Our sampling strategy covered 20 deleted issues (by construction).
- Deleting and editing comments: A comment's author and the repository's maintainer can edit and delete the comment. Editing can be used to remove toxic wording from a comment, though the history of the text is still accessible. This was used rarely in our sample.
- Hiding comments: Maintainers can hide comments, selecting from a list of reasons (`abusive' relates to toxic comments). Those comments are folded in the user interface and only visible if a user explicitly looks at them. This is a recent GitHub feature and was almost never used in our sample.
- Blocking users: Maintainers can block accounts from interacting with the repository. This option is sometimes invoked as a threat (2 cases), but we found only one discussion where maintainers actually blocked a user.
- Invoking the code of conduct: While not a built-in GitHub feature, a common response to toxicity is to post a link to the project's code of conduct (or quote sections) to call out toxic behavior. This was used in 21 issues in our sample.

Note that locked issues, deleted issues, and comments invoking the code of conduct may be overrepresented in our sample, because of how our sample was strati ed (cf. Sec. 3).

Reactions. When faced with a toxic issue or comment, maintainers have a choice of how to react, if at all. They can engage or use the tools above to lock the issue without further discussion. In many cases others engage in the discussion before maintainers do.

While every case is di erent, we found a number of common patterns. In a majority of cases (58 in our sample), maintainers and others tried to engage in subsequent discussions about the issue raised, which turned constructive in 32 cases and escalated into more toxicity in 19 cases. Beyond closing, locking, and discussions, only in 7 cases was there no visible reaction.

When maintainers invoked the code of conduct, the author of the toxic comment usually did not engage any further. However, there were also a few cases where the author pushed back on being policed in their speech, e.g., "Again. No discussion allowed. No critique allowed. Just pushing ngers into the ears and singing. To avoid hearing about the impending doom, to avoid hearing the truth about the quality of this project" (I24), which were usually shut down by locking the discussion. In one case, a user called out for violating the code of conduct responded insisting "I will neither change my language, nor my tone or style. Both, language and tone, are perfectly valid, given the circumstances. I will remain myself, and will repel this attack to my individuality" (I43), referring to invoking the code of conduct as "CoC-Fascism" upon which projects members banned the user.

Locking the discussion was usually e ective, we saw only few cases where the author opened another issue. If they did, it was likely opening with a toxic comment.

Harms. Our method does not allow to reliably measure harms that toxic comments cause, especially indirect harms on bystanders and potential future contributors, who decide not to engage with the repository or open source in general (cf. Sec. 2). Nonetheless, we can infer some harms from reactions.

In almost all cases a maintainer reacts to the toxic issue or comment, even if just to close or lock the issue. That is, maintainers need to use some of their time for extra work.

Furthermore, in many cases, maintainers and others respond patiently after insults, entitled demands, and other toxic comments, trying to help users even if their behavior is questionable. Maintainers often engage to explore whether there is truly an issue behind strongly worded complains. Even when maintainers invoke the code of conduct, they usually do so in a custom comment tailored to the speci c case. All this requires substantial e ort which can be emotionally taxing to developers over time and cause fatigue, as many practitioners attest to (cf. Sec. 2).

9 DISCUSSION AND IMPLICATIONS

#1. Toxicity presents di erently on GitHub: The nature of toxicity observed in our GitHub sample has unique characteristics compared to other platforms, both generic ones like Reddit and Wikipedia as well as software-speci c ones like Stack Over ow (cf. Sec. 2). While insults, arrogance, and trolling were observed in our sample we did not observe more severe uses of language that are common on other platforms like hate speech and o ensive cursing leading us to hypothesize that toxicity on GitHub tends to use milder language compared to other popular platforms like Twitter or Reddit. In contrast to arrogant, insulting, and trolling comments, entitled comments seem to be a phenomenon more speci c to open source and the dynamics of free-to-use software, with seemingly

free support despite no contractual obligations.

#2. Open-source experience does not prevent toxicity: Despite high-pro le known cases in the Linux community [24, 28], we were surprised that most toxicity on GitHub does not stem from trolls and anonymous users, and that experienced open-source developers and even project maintainers authored many toxic comments as well albeit usually not the most severe ones, language-wise.[10] On the one hand, this is concerning because all toxicity can foster an unwelcoming climate, threatening diversity and sustainability. On the other hand, it shows that also maintainers are humans that face emotions as part of their work and may often be exposed to stress [54, 79], e.g., from constant demands for user support. Since maintainer toxicity in our sample was frequently in response to another comment, often in frustration, future work could consider more closely studying the speci c kinds of interactions that frustrate members, where toxic messages might be used as indicator of frustration.

Given that intergroup con ict on many platforms can be traced to few individuals (e.g., 74% of Reddit con icts started by 1% of communities [52]), future research should also investigate whether a few accounts (maintainers, experienced developers, or repeat issue reporters) repeatedly engage in toxicity and cause an outsized number of toxic interactions. This could inform future mitigation strategies, such as blocking or shadow-banning repeat o enders. More research is needed to understand whether toxicity from maintainers is frequent (in our sample) because maintainers actually engage in toxic behavior frequently or because maintainer messages are just more common among all GitHub comments.

#3. Research into toxicity harms needed: Our research can only provide limited insights into harms of toxicity. Even though toxic messages appear to be fairly rare, some harms clearly exist given from blog posts and talks by open-source practitioners (cf. Sec. 2), but further research is needed to understand the severity and pervasiveness of harms.

While some forms of toxicity seem more obviously harmful (wasting maintainers' time, causing negative emotions and stress) and are often called out clearly, others, especially unprofessional comments, are less obvious in how they a ect participants directly or the project indirectly. Past research has shown that prospective contributors are attentive to the tone of discussions in a new project [76]. Also, while toxic comments directed at code are quite common in our sample (in line with results from prior work on anger detection in collaborative software development [34]), research is needed to determine whether people perceive or react to toxicity di erently when it's directed at their code rather than at themselves directly. Future research into perceived harms and the association of unprofessional toxic language to open source sustainability could provide more evidence on how to mitigate di erent forms of toxicity.

Our dataset (and the method used to build it) can provide a starting point for engaging with community members about perceived harms. A more reliable detector could provide an important path to enable more quantitative research designs at scale.

[10] This does not seem unique to GitHub ; studies on Wikipedia similarly have found toxicity from established members [106].

#4. Opportunities to build open-source-speci c detectors: O -the-shelf language-based detectors of toxicity trained on data from other platforms can now be con dently expected to not generalize well to GitHub . The milder language in insults makes them more di cult to detect by looking for `bad words,' which tend to reveal more comments of an unprofessional or trolling nature. Entitlement and arrogance are often toxic not through the use of strong words, but through the message they are conveying and may require tools trained speci cally on a corpus of toxic open source discussions. Possible features to look for might include references to timing, urgency, and priority.

In addition, several other discussed characteristics may be useful as contextual features in a classi er, such as the position of the comment, the role and history and anonymity of the author, the size and domain of the project, and even the content of the discussion. Given the common triggers of errors and technical disagreements tailoring detectors to understand technical discourse may be helpful to provide context. Using information about the past toxic interactions of a user for detecting future toxicity may also be e ective, given how some repeat issue reporters and maintainers may have distinct communication patterns.

#5. Tailored interventions are promising: Our results suggest that some common manual interventions are e ective at curbing toxic discussions. Right now, maintainers use multiple tools and often write custom messages to deal with toxicity. We suspect there is potential for streamlining manual and automated interventions.

Toxicity manifests in many di erent ways and not all may be reliably detected by a tool. Classic detection-based interventions like automatically asking This message looks similar to others previously reported as toxic. Would you like to edit it? (as deployed on Instagram and NextDoor) may be feasible for strong language (including unprofessional jokes), emotional language, and arrogance. For entitled messages from new users and repeat issue reporters, a message about open-source cultural norms may be more tting.

Beyond detection-based approaches, there are many other possible technology-supported interventions. For example, a platform-wide system for reporting of toxic behavior to ban users after a certain number of warnings (strikes) and tools that automate the repetitive task of responding to toxic users (close, lock, post prepared message) could reduce time and emotional involvement.

Similarly, a substantial number of toxic comments in our sample relate to users' frustration with attempting to use the project or maintainers' frustration with such requests. It is worth exploring whether projects that (a) manage other support channels (e.g., forums, Slack [12] or Gitter [72, 91], Stack Exchange [3, 100]), (b) convey clear expectation for communication standards in support requests, or (c) raise the cost for support requests (e.g., enforcing reporting templates) attract fewer toxic interactions or just move them elsewhere.

More exchange about possible solutions and best practice is needed to share and adopt e ective practices, ideally based on empirical evidence of their e ectiveness and costs.

10 CONCLUSIONS

In this paper we explore online toxicity in open-source communities on GitHub. We nd that the frequently present types of toxicity in

open-source communities are di erent than those reported on other online platforms like Reddit or Wikipedia. Within open source, entitled and demeaning complaints, arrogance, and insults are common forms of toxicity. Since the prevalent forms of toxicity in open source di er compared to other platforms, tools and interventions built to address toxicity are unlikely to be e ective out of the box in open-source communities. For example, tools meant to identify and remove hate speech on a platform like Facebook would likely not have the same impact on toxicity in open source since this form of toxicity seems rare on GitHub.

Future work should investigate the harms of various common forms of toxicity in open source (e.g., entitlement and arrogance) on contributors and evaluate the role of project members and experience. Additionally future work should design, build, and evaluate open source speci c tools and interventions for detecting and addressing toxicity.

11 ACKNOWLEDGEMENTS

First and foremost, special thanks is given to Chanel for being the an exemplary canine researcher and for providing emotional support, motivation, and inspiration on a daily basis. We greatly appreciate earlier e orts by Minxuan Cao and Naveen Raman in identifying toxic comments and building an initial classi er, work that in many ways encouraged this more qualitative exploration. We would also like to thank Huilian Sophie Qiu for helping with the exploratory data analysis.

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant Numbers DGE1745016 and DGE2140739. Any opinions, ndings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily re ect the views of the National Science Foundation. Kaestner's work was supported in part by the National Science Foundation and the Sloan Foundation. Vasilescu has been supported in part by the NSF (awards 1901311 and 2107298) and the Sloan Foundation. This work was also supported in part by a Google Faculty Research Award.

REFERENCES

[1] B Thomas Adler and Luca De Alfaro. 2007. A content-driven reputation system for the Wikipedia. In International Conference on World Wide Web. 261 270.
[2] Khaled Albusays, Pernille Bjorn, Laura Dabbish, Denae Ford, Emerson Murphy-Hill, Alexander Serebrenik, and Margaret-Anne Storey. 2021. The diversity crisis in software development. IEEE Software 38, 2 (2021), 19 25.
[3] Anonymous. 2014. Leaving Toxic Open Source Communities. https://modelviewculture.com/pieces/leaving-toxic-open-source-communities.
[4] Guilherme Avelino, Leonardo Passos, Andre Hora, and Marco Tulio Valente. 2016. A novel approach for estimating truck factors. In 2016 IEEE 24th International Conference on Program Comprehension (ICPC). IEEE, 1 10.
[5] Jennifer Bayzick, April Kontostathis, and Lynne Edwards. 2011. Detecting the presence of cyberbullying using computer software. (2011).
[6] Christopher Bogart, Christian Kästner, James Herbsleb, and Ferdian Thung. 2016. How to break an API: cost negotiation and community values in three software ecosystems. In Proc. Found. Software Engineering (ESEC/FSE). 109 120.
[7] Amiangshu Bosu and Kazi Zakia Sultana. 2019. Diversity and inclusion in open source software (OSS) projects: Where do we stand?. In International Symposium on Empirical Software Engineering and Measurement (ESEM). IEEE, 1 11.
[8] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. Qualitative Research in Psychology 3, 2 (2006), 77 101.
[9] Luke Breitfeller, Emily Ahn, David Jurgens, and Yulia Tsvetkov. 2019. Finding microaggressions in the wild: A case for locating elusive phenomena in social media posts. In Proc. Int'l Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 1664 1674.
[10] Peter Burnap and Matthew Leighton Williams. [n.d.]. Hate speech, machine classi cation and statistical modelling of information ows on Twitter: Interpretation and communication for policy decision making. In Internet, Policy & Politics.
[11] Eshwar Chandrasekharan, Umashanthi Pavalanathan, Anirudh Srinivasan, Adam Glynn, Jacob Eisenstein, and Eric Gilbert. 2017. You can't stay here: The e cacy of Reddit's 2015 ban examined through hate speech. Proceedings of the ACM on Human-Computer Interaction, CSCW (2017), 1 22.
[12] Preetha Chatterjee, Kostadin Damevski, Lori Pollock, Vinay Augustine, and Nicholas A Kraft. 2019. Exploratory study of Slack Q&A chats as a mining source for software engineering tools. In International Conference on Mining Software Repositories (MSR). IEEE, 490 501.
[13] Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Emiliano De Cristofaro, Gianluca Stringhini, and Athena Vakali. 2017. Mean birds: Detecting aggression and bullying on Twitter. In Web Science Conference. 13 22.
[14] Jithin Cheriyan, Bastin Tony Roy Savarimuthu, and Stephen Crane eld. 2020. Norm violation in online communities A study of Stack Over ow comments. arXiv preprint arXiv:2004.05526 (2020).
[15] Jithin Cheriyan, Bastin Tony Roy Savarimuthu, and Stephen Crane eld. 2021. Towards o ensive language detection and reduction in four Software Engineering communities. In Evaluation and Assessment in Softw. Eng. 254 259.
[16] Usman W Chohan and Aron D'Souza. 2020. A critical appraisal of the Twitterverse. Social Media Critical Research Series (2020).
[17] Moataz Chouchen, Ali Ouni, Raula Gaikovina Kula, Dong Wang, Patanamon Thongtanunam, Mohamed Wiem Mkaouer, and Kenichi Matsumoto. 2021. Anti-patterns in modern code review: Symptoms and prevalence. In Int Conf. Software Analysis, Evolution and Reengineering (SANER). IEEE, 531 535.
[18] John W Creswell and Cheryl N Poth. 2016. Qualitative inquiry and research design: Choosing among ve approaches. Sage publications.
[19] Kevin Crowston, Kangning Wei, James Howison, and Andrea Wiggins. 2008. Free/Libre open-source software development: What we know and what we do not know. ACM Computing Surveys (CSUR) 44, 2 (2008), 1 35.
[20] Srayan Datta and Eytan Adar. 2019. Extracting inter-community con icts in Reddit. In Int'l AAAI Conference on Web and Social Media, Vol. 13. 146 157.
[21] Fabio Del Vigna12, Andrea Cimino23, Felice Dell'Orletta, Marinella Petrocchi, and Maurizio Tesconi. 2017. Hate me, hate me not: Hate speech detection on Facebook. In First Italian Conference on Cybersecurity (ITASEC17). 86 95.
[22] Norman K Denzin and Yvonna S Lincoln. 2011. The Sage handbook of qualitative research. Sage.
[23] 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.
[24] Alan Diggs. 2021. `Windows is sh*t:' Linux Users and The Technical Superiority Problem. https://medium.com/linuxforeveryone/windows-is-sh-t-linux-users-and-the-technical-superiority-problem-196a597aa860. Accessed: 2021-08-19.
[25] Lucas Dixon, John Li, Je rey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classi cation. In AAAI/ACM Conference on AI, Ethics, and Society. 67 73.
[26] Maeve Duggan. 2014. Online Harassment. https://www.pewresearch.org/internet/2014/10/22/online-harassment/.
[27] 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). IEEE, 174 185.
[28] Isabella Ferreira, Jinghui Cheng, and Bram Adams. 2021. The Shut the f**k up Phenomenon: Characterizing Incivility in Open Source Code Review Discussions. arXiv preprint arXiv:2108.09907 (2021).
[29] Pnina Fichman and Samantha Sharp. 2020. Successful trolling on Reddit: A comparison across subreddits in entertainment, health, politics, and religion. Proc. Assoc. Information Science and Technology 57, 1 (2020), e333.
[30] United States. National Commission for the Protection of Human Subjects of Biomedical and Behavioral Research. 1978. The Belmont report: ethical principles and guidelines for the protection of human subjects of research. Vol. 2. The Commission.
[31] Denae Ford, Kristina Lustig, Jeremy Banks, and Chris Parnin. 2018. We Don't Do That Here How Collaborative Editing with Mentors Improves Engagement in Social Q&A Communities. In Proc. Conf. Human Factors in Computing Systems.
[32] Denae Ford and Chris Parnin. 2015. Exploring causes of frustration for software developers. In International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE). IEEE, 115 116.
[33] Denae Ford, Justin Smith, Philip J Guo, and Chris Parnin. 2016. Paradise unplugged: Identifying barriers for female participation on Stack Over ow. In Joint Meeting on the Foundations of Software Engineering (ESEC/FSE). 846 857.
[34] Daviti Gachechiladze, Filippo Lanubile, Nicole Novielli, and Alexander Serebrenik. 2017. Anger and its direction in collaborative software development. In Int'l Conf. Software Eng.: New Ideas and Emerging Results (ICSE-NIER). 11 14.
[35] Iginio Gagliardone, Danit Gal, Thiago Alves, and Gabriela Martinez. 2015. Countering online hate speech. Unesco Publishing.
[36] R Stuart Geiger and David Ribes. 2010. The work of sustaining order in Wikipedia: The banning of a vandal. In ACM Conference on Computer Supported

Cooperative Work (CSCW). 117–126.
[37] Georgios Gousios and Diomidis Spinellis. 2012. GHTorrent: GitHub's data from a firehose. In *Int'l Conf. Mining Software Repositories (MSR)*. IEEE, 12–21.
[38] 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.
[39] Joshua Guberman, Carol Schmitz, and Libby Hemphill. 2016. Quantifying toxicity and verbal violence on Twitter. In *ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW) Companion*. 277–280.
[40] Emitza Guzman and Bernd Bruegge. 2013. Towards emotional awareness in software development teams. In *Proceedings of the 2013 9th joint meeting on foundations of software engineering*. 671–674.
[41] Jay Hanlon. 2018. Stack Overflow Isn't Very Welcoming. It's Time for That to Change. https://stackoverflow.blog/2018/04/26/stack-overflow-isnt-very-welcoming-its-time-for-that-to-change/.
[42] Claire Hardaker. 2010. Trolling in asynchronous computer-mediated communication: From user discussions to academic definitions. *Journal of Politeness Research* (2010).
[43] Rick H Hoyle, Monica J Harris, and Charles M Judd. 2002. *Research methods in social relations*. Wadsworth Fort Worth, TX.
[44] Amanda Lee Hughes and Leysia Palen. 2009. Twitter adoption and use in mass convergence and emergency events. *International journal of emergency management* 6, 3-4 (2009), 248–260.
[45] Bryan Hughes. 2017. Why I'm leaving the Node.js project. https://medium.com/@nebrius/why-im-leaving-the-node-js-project-bff946845a77. Accessed: 2021-08-23.
[46] Daniel Izquierdo-Cortazar, Gregorio Robles, Felipe Ortega, and Jesus M Gonzalez-Barahona. 2009. Using software archaeology to measure knowledge loss in software projects due to developer turnover. In *2009 42nd Hawaii International Conference on System Sciences*. IEEE, 1–10.
[47] Schykle Jason Evangelho, Alan Diggs. 2020. Linux + Coffee #4: Tribalism and Toxicity. Podcast. https://player.fm/series/series-2567058/linux-coffee-4-tribalism-and-toxicity
[48] Jigsaw. 2017. Perspective API: Using machine learning to reduce toxicity online. https://www.perspectiveapi.com/. accessed Aug 21, 2021.
[49] Prerna Juneja, Deepika Rama Subramanian, and Tanushree Mitra. 2020. Through the Looking Glass: Study of Transparency in Reddit's Moderation Practices. *Proceedings of the ACM on Human-Computer Interaction* 4, GROUP (2020), 1–35.
[50] Anat Brunstein Klomek, Frank Marrocco, Marjorie Kleinman, Irvin Sam Schonfeld, and Madelyn S Gould. 2008. Peer victimization, depression, and suicidiality in adolescents. *Suicide and Life-Threatening Behavior* 38, 2 (2008), 166–180.
[51] Robin M Kowalski, Susan P Limber, and Patricia W Agatston. 2012. *Cyberbullying: Bullying in the digital age*. John Wiley & Sons.
[52] Srijan Kumar, William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2018. Community interaction and conflict on the web. In *Proc. WWW Conf.* 933–943.
[53] Hemank Lamba, Momin M Malik, and Jurgen Pfeffer. 2015. A tempest in a teacup? analyzing firestorms on Twitter. In *International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 17–24.
[54] Nolan Lawson. 2017. What it feels like to be an open-source maintainer. https://nolanlawson.com/2017/03/05/. Accessed: 2021-08-17.
[55] Renee Li, Pavitthra Pandurangan, Hana Frluckaj, and Laura Dabbish. 2021. Code of Conduct Conversations in Open Source Software Projects on Github. *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW1 (2021), 1–31.
[56] Yvonna S Lincoln and Egon G Guba. 1985. *Naturalistic inquiry*. sage.
[57] Yabing Liu, Chloe Kliman-Silver, and Alan Mislove. 2014. The tweets they are a-changin': Evolution of Twitter users and behavior. In *Eighth International AAAI Conference on Weblogs and Social Media*.
[58] J Matias, Amy Johnson, Whitney Erin Boesel, Brian Keegan, Jaclyn Friedman, and Charlie DeTar. 2015. Reporting, reviewing, and responding to harassment on Twitter. *Available at SSRN 2602018* (2015).
[59] Aline Maya. 2019. Hated in the Nation and #DeathTo: What are the Consequences of Trial by Twitter? *Black Mirror and Philosophy: Dark Refl.* (2019).
[60] Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica* 22, 3 (2012), 276–282.
[61] Matthew B Miles, A Michael Huberman, and Johnny Saldana. 2014. *Fundamentals of qualitative data analysis*. Sage Los Angeles, CA.
[62] Courtney Miller, David Gray Widder, Christian Kästner, and Bogdan Vasilescu. 2019. Why do people give up flossing? a study of contributor disengagement in open source. In *IFIP Int'l Conf. Open Source Systems*. Springer, 116–129.
[63] Sebastian C Müller and Thomas Fritz. 2015. Stuck and frustrated or in flow and happy: Sensing developers' emotions and progress. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, Vol. 1. IEEE, 688–699.
[64] Alessandro Murgia, Parastou Tourani, Bram Adams, and Marco Ortu. 2014. Do developers feel emotions? an exploratory analysis of emotions in software artifacts. In *Int'l Conf. Mining Software Repositories (MSR)*. 262–271.
[65] Emerson Murphy-Hill, Thomas Zimmermann, and Nachiappan Nagappan. 2014. Cowboys, ankle sprains, and keepers of quality: How is video game development different from software development?. In *Proc. ICSE*. 1–11.
[66] Dawn Nafus. 2012. 'Patches don't have gender': What is not open in open source software. *New Media & Society* 14, 4 (2012), 669–683.
[67] Charisse L Nixon. 2014. Current perspectives: the impact of cyberbullying on adolescent health. *Adolescent health, medicine and therapeutics* 5 (2014), 143.
[68] Nicole Novielli, Fabio Calefato, Filippo Lanubile, and Alexander Serebrenik. 2021. Assessment of off-the-shelf SE-specific sentiment analysis tools: An extended replication study. *Empirical Software Engineering* 26, 4 (2021), 1–29.
[69] Lorelli S Nowell, Jill M Norris, Deborah E White, and Nancy J Moules. 2017. Thematic analysis: Striving to meet the trustworthiness criteria. *International Journal of Qualitative Methods* 16, 1 (2017), 1609406917733847.
[70] Dan Olweus. 1994. Bullying at school. In *Aggressive behavior*. Springer, 97–130.
[71] Marco Ortu, Alessandro Murgia, Giuseppe Destefanis, Parastou Tourani, Roberto Tonelli, Michele Marchesi, and Bram Adams. 2016. The emotional side of software developers in JIRA. In *Proc. MSR*. IEEE, 480–483.
[72] Esteban Parra, Ashley Ellis, and Sonia Haiduc. 2020. GitterCom: A Dataset of Open Source Developer Communications in Gitter. In *International Conference on Mining Software Repositories (MSR)*. 563–567.
[73] Jürgen Pfeffer, Thomas Zorbach, and Kathleen M Carley. 2014. Understanding online firestorms: Negative word-of-mouth dynamics in social media networks. *Journal of Marketing Communications* 20, 1-2 (2014), 117–128.
[74] Martin Potthast, Benno Stein, and Robert Gerling. 2008. Automatic vandalism detection in Wikipedia. In *European Conf. Inf. Retrieval*. Springer, 663–668.
[75] Reid Priedhorsky, Jilin Chen, Shyong (Tony) K Lam, Katherine Panciera, Loren Terveen, and John Riedl. 2007. Creating, destroying, and restoring value in Wikipedia. In *Proc. Int'l Conf. Supporting Group Work*. 259–268.
[76] Huilian Sophie Qiu, Yucen Lily Li, Susmita Padala, Anita Sarma, and Bogdan Vasilescu. 2019. The signals that potential contributors look for when choosing open-source projects. *Proc. Human-Computer Interaction* 3, CSCW (2019), 1–29.
[77] Huilian Sophie Qiu, Alexander Nolte, Anita Brown, Alexander Serebrenik, and Bogdan Vasilescu. 2019. Going farther together: The impact of social capital on sustained participation in open source. In *Proc. ICSE*. 688–699.
[78] Md Mustafizur Rahman, Dinesh Balakrishnan, Dhiraj Murthy, Mucahid Kutlu, and Matthew Lease. 2021. An Information Retrieval Approach to Building Datasets for Hate Speech Detection. *arXiv preprint arXiv:2106.09775* (2021).
[79] 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 *International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*. 57–60.
[80] Sam Ransbotham and Gerald C Kane. 2011. Membership turnover and collaboration success in online communities: Explaining rises and falls from grace in Wikipedia. *Mis Quarterly* (2011), 613–627.
[81] Reddit. 2015. Promote ideas, protect people. https://redditblog.com/2015/05/14/promote-ideas-protect-people/.
[82] Reddit. 2015. Reddit March 2015 Survey. https://docs.google.com/document/u/1/d/1QJBPZt0oa3UCkL6QGBHp6vITXs3f1bYcCyA5xIQcFZw/pub.
[83] Lena Reinhard. 2014. *This is bigger than us: Building a future for Open Source*. JSConf EU. https://www.youtube.com/watch?v=-thLNvxFUu4
[84] Kelly Reynolds, April Kontostathis, and Lynne Edwards. 2011. Using machine learning to detect cyberbullying. In *2011 10th International Conference on Machine learning and applications and workshops*, Vol. 2. IEEE, 241–244.
[85] Jim Salter. 2021. The Perl Foundation is fragmenting over Code of Conduct enforcement. https://arstechnica.com/gadgets/2021/08/the-perl-foundation-is-fragmenting-over-code-of-conduct-enforcement/.
[86] Jaydeb Sarker, Asif Kamal Turzo, and Amiangshu Bosu. 2020. A Benchmark Study of the Contemporary Toxicity Detectors on Software Engineering Interactions. In *Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 218–227.
[87] Anna Schmidt and Michael Wiegand. 2017. A survey on hate speech detection using natural language processing. In *Fifth International Workshop on Natural Language Processing for Social Media*. 1–10.
[88] Margrit Schreier. 2012. *Qualitative content analysis in practice*. Sage publications.
[89] Pnina Shachaf and Noriko Hara. 2010. Beyond vandalism: Wikipedia trolls. *Journal of Information Science* 36, 3 (2010), 357–370.
[90] Sonia Sharp and Peter Smith. 2002. *School bullying: Insights and perspectives*. Routledge.
[91] Lin Shi, Xiao Chen, Ye Yang, Hanzhi Jiang, Ziyou Jiang, Nan Niu, and Qing Wang. 2021. A First Look at Developers' Live Chat on Gitter. *arXiv preprint arXiv:2107.05823* (2021).
[92] Peter K Smith, Jess Mahdavi, Manuel Carvalho, Sonja Fisher, Shanette Russell, and Neil Tippett. 2008. Cyberbullying: Its nature and impact in secondary school pupils. *Journal of child psychology and psychiatry* 49, 4 (2008), 376–385.
[93] Igor Steinmacher, Gustavo Pinto, Igor Scaliante Wiese, and Marco Aurélio Gerosa. 2018. Almost there: A study on quasi-contributors in open-source software projects. In *Proc. ICSE*. 256–266.
[94] Igor Steinmacher, Igor Wiese, Ana Paula Chaves, and Marco Aurélio Gerosa. 2013. Why do newcomers abandon open source software projects?. In *Int'l Workshop Cooperative and Human Aspects of Softw. Eng. (CHASE)*. 25–32.
[95] Glenn Sterner and Diane Felmlee. 2017. The social networks of cyberbullying on Twitter. *International Journal of Technoethics (IJT)* 8, 2 (2017), 1–15.