

ASTRO: Defining and Identifying Hostile Behavior on GitHub

Ben Treves¹, Maya Treves¹, Michalis Faloutsos¹, Jeremy Blackburn², Emiliano De Cristofaro¹

¹University of California, Riverside

²Binghamton University

btrev003@ucr.edu, mtrev008@ucr.edu, michalis@cs.ucr.edu, jblackbu@binghamton.edu, emilianodc@cs.ucr.edu

Abstract

Open Source Software (OSS) platforms like GitHub thrive on large-scale collaboration, yet hostile interactions can erode participation and undermine community health. This motivates the need to identify and profile hostile user-generated content on GitHub. In this paper, we employ a multi-faceted hostility definition that encompasses user behavior that deters collaboration or participation by other users, including toxic, insulting, and threatening behaviors. We then present *ASTRO*, a systematic framework for identifying and profiling user-generated hostile content on GitHub at scale. Given a user or repository and its associated comments, we generate a multi-dimensional hostility vector that captures activity across each hostility attribute. Algorithmically, we introduce and combine two layers: 1) the PerspectiveAPI functionality, and 2) an appropriately-prompted LLM to find a good balance between accuracy and computational cost.

We collect and then deploy *ASTRO* on three datasets: (a) 2,344 hostile users, (b) 1,253 hostile repositories, and (c) 100 benign random users. First, we show that our two-layer method achieves good performance and cost: it achieves an F1 score of 0.85 on our ground truth, while reducing the cost of using only the LLM layer by a factor of roughly 29. Second, we study the relationship between different dimensions of hostility and the distribution of hostile behavior across users and repositories, e.g., hostile users post much more than non-hostile ones. Finally, we shed light on communities that appear more conducive to hostile behavior, e.g., projects related to video game development. Overall, our work demonstrates that a nuanced understanding of hostility and the ability to detect it effectively can help model it and assist the development of approaches to mitigate its adverse effects on GitHub and other OSS platforms.

1 Introduction

Open Source Software (OSS) platforms like GitHub harness the collective expertise of large communities of contributors. As user collaboration is critical for maintaining and fostering their quality, e.g., by facilitating bug detection (Zhao and Deek 2004), the presence of hostile interactions can dramatically discourage participation, posing a significant threat to the ecosystem (Sayago-Heredia et al. 2025). Hostility can trigger cascading effects: it might deter

potential contributors or frustrate existing ones, ultimately undermining the integrity and the progress of the projects as a whole (Graziotin, Wang, and Abrahamsson 2014; Ortu et al. 2015; Sayago-Heredia et al. 2025).

By most measures, GitHub is the most popular website for hosting and collaborating on OSS projects, with over one billion repositories (Tyson 2025) and 150 million users.¹ While we are interested in the overall OSS ecosystem, this work focuses on GitHub due to its popularity.

Problem Statement. In this paper, we study hostile behavior on GitHub – more precisely, investigating how to identify and profile hostile user-generated content on GitHub and to elicit interesting patterns. To do so, we first need to define hostility carefully, since a definition that is too wide or too narrow can lead to misleading conclusions. For example, profanity per se might not necessarily be situationally inappropriate (Jay and Janschewitz 2008) and might not ultimately deter collaboration on OSS platforms; depending on the context, the resulting harm might be limited, as, e.g., swearing can also serve a functional/cathartic function (Pamungkas, Basile, and Patti 2023). At the same time, hostility, e.g., Linus Torvald’s rather infamous behavior, extends past profanity (Torvalds 2013).

Next, we need a scalable and effective methodology to classify GitHub interactions as hostile. Prior work mostly relies on qualitative and/or smaller-scale studies (Miller et al. 2022; Ferreira, Adams, and Cheng 2022; Ehsani et al. 2024), or focuses on different domains such as code reviews (Devlin et al. 2019), pull requests (Sultana 2022), or gender-based discrimination (Sultana and Begum Kali 2024). Whereas larger-scale studies primarily stop at preliminary analyses (Raman et al. 2020; Mishra and Chatterjee 2024), often based on the outdated (and, to the best of our knowledge, no longer available) GHTorrent dataset (Gousios and Spinellis 2012). Finally, we aim to use this methodology to assess whether specific communities are more prone to hostility and to study possible trends in the behavior of user groups and repositories.

Overall, in this paper, we identify and address the following research questions:

- **RQ1.** How can we accurately classify nuanced forms of hostile user-generated content on GitHub at scale?

- **RQ2.** What patterns can we identify in the behavior of users and repositories that tend to be hostile?

Our study builds on GitHub issue comments from public repositories (from September 2006 to June 2025), including metadata like text, author, creation date, and parent repository/issue thread. We collect 1,253 GitHub repositories and all their comments, as well as 2,344 users and all their comments. These two datasets were collected via targeted sampling to increase the likelihood of capturing hostile behavior, as described in Section 2.

We then introduce *ASTRO*, a reusable and tunable framework for identifying and profiling hostile behavior on GitHub. *ASTRO* encompasses a scalable algorithmic approach that can process large numbers of posts, scalably and accurately, by combining two layers: 1) a computationally efficient layer to filter out ostensibly benign comments, followed by 2) a Large Language Model (LLM)-based classification for improved accuracy for the remaining comments.

More precisely, we critically revisit emotional concepts (“attributes”) that can negatively impact conversations as defined by PerspectiveAPI (Lees et al. 2022). We pick five of the six production attributes of the API: insult, threat, identity attack, toxicity, and severe toxicity, while leaving out profanity, which is more likely, in this context, to lead to an overly-sensitive definition of hostility (see Section 3.1). If PerspectiveAPI returns scores below 0.3 for all five attributes, we filter out that comment.

For the second layer, we use an open-source LLM – namely, llama3-70b-8192, selected after careful evaluation – to build a five-dimensional hostility vector by prompting it on a comment along with the definitions of the five attributes (insult, threat, identity attack, toxicity, severe toxicity). While PerspectiveAPI has been used extensively in prior work as a sole measure of hostility on online communities, it has also been shown not to be particularly effective on GitHub (Sarker, Turzo, and Bosu 2020), as we also find (see Section 4). In the process, we also experiment with different LLMs, comparing open-source models to proprietary ones, and find that the former outperform the latter.

Finally, we conduct an exploratory study of several interesting hostile users and repositories.

Contributions. Our work makes several main contributions:

1. **Scalability and accuracy:** Our approach strikes a good balance between accuracy and scalability. Our lower-accuracy but less expensive first layer eliminates 96.6% of the comments. As a result, the cost of the more expensive LLM layer is reduced by over an order of magnitude. As a measure of this reduction, note that we spent a total of \$35.56 on LLM API calls on the filtered comments, saving us a potential \$1,043.84 had we not incorporated this first layer – a reduction by a factor of 29 in monetary cost, which also translates to saving on energy and environmental costs (Moore et al. 2025).
2. **Hostility definition matters.** We capture hostility along five dimensions, providing a more accurate picture of hostility, while allowing flexibility of use to fit the desired context. For example, if we did not consider insults to be hostile, the number of hostile comments would

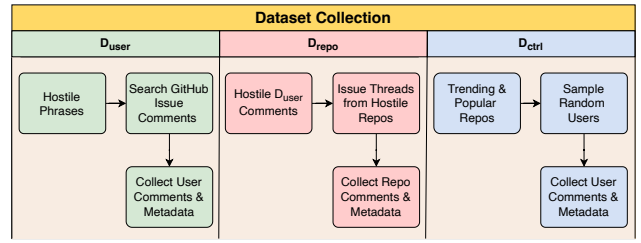


Figure 1: Overview of our data collection process.

drop by 6.6%, or much more so if one only wanted to consider threats that are not toxic, which would eliminate most (98.3%) of the comments from our dataset. In other words, the five-dimensional approach, along with the overall modular design of *ASTRO*, affords future users of our framework great deal of versatility.

3. **Communities exhibit different hostility levels.** Some communities are significantly more prone to hostility. For example, repositories centered around video games or operating systems development are among the most hostile in our data sample, and they draw hostile users with similar types of hostility.
4. **Pride and hostility: when do users become hostile?** Most hostile users in our dataset seem to: (a) often respond with hostile behavior when faced with criticism about their work, and (b) are quite active developers, having a number of public repositories.

Broader Implications. Although our analysis is performed on a sample of publicly available repositories – thus, our observations may not apply to private repositories or other platforms – we uncover a number of worrisome and problematic behaviors in what should, at least in theory, be a community-driven platform combining both automated techniques to enforce codes of conduct and moderation tools.

Understanding OSS platform-specific hostile behavior constitutes an essential milestone toward inclusive, respectful, and overall efficient collaboration spaces for OSS developers. Identifying hostility is the first step in developing mitigation techniques. On GitHub and other OSS collaboration platforms, hostility can foster antagonism and discourage developers from contributing to projects (Raman et al. 2020), thereby stifling innovation and limiting the usefulness of open-source communities. By detecting and taking punitive action against users who instigate hostility, platforms can increase healthy interactions in OSS projects, with users at large benefitting from more collaborative and higher-quality software.

2 Dataset

Using a custom crawler, we collect comments posted by GitHub users in the issue threads of public repositories made between Sep 11, 2006 and Jun 3, 2025. For each comment, we gather its textual data, author, creation date, parent repository, and parent issue thread. We provide a visualization of our data collection process in Figure 1. More precisely, we

Dataset	#Users	#Repos	#Issues	#Comments
D_{user}	2,344	24,456	90,492	235,177
D_{repo}	91,644	1,253	141,345	498,341
D_{ctrl}	100	361	595	1,083
Total	102,137	24,656	232,432	734,601

Table 1: Overview of our Datasets. #Issues represents # of unique issue threads. For D_{user} , #Repos reports the number of unique repositories where users in this dataset posted a comment, and, for D_{repo} , #Users represents the number of unique users who posted a comment in these repositories.

collect and study three datasets, as also summarized in Table 1:

- D_{user} , a sample of hostile users and all their issue thread comments,
- D_{repo} , a subset of hostile repositories from D_{user} and all their comments, and
- D_{ctrl} , a control dataset of randomly selected users.

User Dataset (D_{user}). Due to the relative scarcity of hostility on GitHub (e.g., Raman et al. (2020) report that less than 1% of the issue threads are toxic, with toxicity decreasing over time), we specifically look for hostile users by searching for hostile activity. To do so, we start by compiling a list of 100 hostile phrases via the Google Gemini 2.5 Flash LLM, which are included in our anonymous repository.² We search for each phrase through the GitHub search interface and apply filters to only include results from issue thread comments, then we compile a list of users posting this hostile content from the usernames of every author that appears in the first five result pages of each phrase.

Next, we build a preliminary dataset of comments by collecting the issue comments posted by each user. For each username, we crawl the issue thread search page of GitHub for all comments owned by that user, i.e., we crawl pages using the GitHub search bar query `is:issue owner:[username]`. From each resulting page, up to a maximum of ten pages, we save the repository names and issue numbers. The result is a list of the repositories and specific issue threads that the user has commented on. From there, we go back and collect each issue thread, saving any comments made by the user. This yields D_{user} , a dataset of 2.3K users and 235K comments.

Repository Dataset (D_{repo}). Next, we build a repository-centric corpus. We select repositories by gathering the hostile comments from the D_{user} dataset and compiling a list of the repositories they were posted in. We do so by using *ASTRO*, presented in Section 3, which labels comments using an LLM-based hostility classification pipeline (after filtering out comments with low PerspectiveAPI scores).

From the list, we collect the most recent issue threads from each repository, up to the 200 most recent threads. We then collect every comment within those threads, along with relevant information, such as the issue and repository it was posted in, the author of the comment, the date, and whether

²<https://anonymous.4open.science/r/ASTRO-GitHub>.

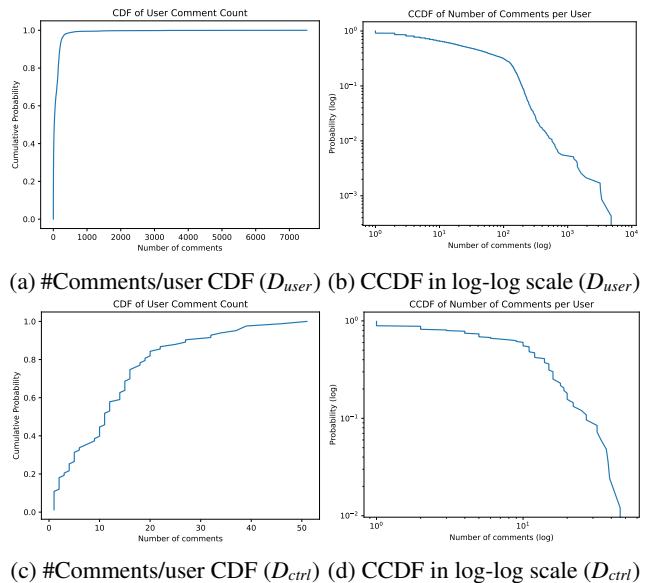


Figure 2: CDFs and CCDFs of the number of comments per user in D_{user} top row, and D_{ctrl} bottom row.

the issue was locked at the time of collection. The end result is D_{repo} , a dataset of 1,253 repositories and 498K comments.

Control Dataset (D_{ctrl}). Finally, we create a baseline dataset. We start by identifying 50 trending and all-time popular repositories. We gather the username of each user that has posted a comment in at least one of the most recent 200 issue threads from each of these repositories, and randomly sample 100 users from this list. We then create D_{ctrl} by collecting each of these 100 users’ issue comments, regardless of repository; this results in a dataset of 1,083 comments.

Dataset Statistics. To get a sense of what our dataset looks like, we look at the total number of comments posted by each user or in each repository. Overall, a relatively small percentage of the users post the majority of the comments. Recall that we collect D_{user} with a limit of ten result pages per user, and D_{repo} with a cap of 200 issue threads per repo; therefore, while the higher ends of the scale are not indicative of the entire user population, the overall trends remain.

Our dataset indicates that hostile users are much more active commenters than benign users. In Figure 2, we visualize the differences in commenting behavior between the hostile users of D_{user} and the benign users of D_{ctrl} . For D_{user} , the median number of comments per user is 28. One in three users (33.6%) post at most 10 comments, two-thirds (68.4%) at most 100, while only 0.5% have more than 1,000. This is starkly different from the D_{ctrl} baseline, where the median number of comments per user is 10, with about one in six users (17.0%) posting no comment, half posting at most 10 comments, and none posting more than 100 comments.

We also look at the number of issue threads each user posts in: once again, a relatively small number of users comment on the overwhelming majority of issue threads. More precisely, the median number of issues per user is 15, with slightly less than half (43.0%) in at most 10 threads, slightly

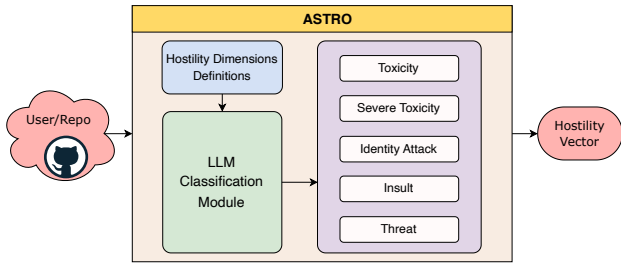


Figure 3: Overview of the *ASTRO* framework.

more than half (58.0%) in at most 25 threads, and only a few (7.4%) in more than 100. By comparison, for D_{ctrl} , the median is 6.5 with almost two-thirds of users (61.0%) posting in at most 10 issue threads, only one user in over 25 threads, and none posting more than 100.

We also compare repositories with and without hostile comments. For D_{repo} , the median number of comments per repository is 319, with the distribution being a bit less skewed. More precisely, about one in eight repositories (12.3%) have at most 10 comments, about one third (34.5%) at most 100, and only 5.3% have more than 100 comments. There are 20 (1.6%) repositories with only two comments, the lowest comment count in D_{repo} . We also look at the number of issue threads each repository has. The median is 126, with almost one quarter of repositories (23.8%) having at most 10 issue threads, about half (47.5%) having fewer than 100, and about half (45.1%) having 200 or more. There are 96 repositories (7.7%) with only one issue thread, and no repositories with zero.

3 The *ASTRO* Framework

In this section, we present *ASTRO*, our framework for classifying and profiling hostile GitHub content. We present a high-level visualization in Figure 3.

3.1 Defining Hostility

Defining hostility is inherently complex, as is the case with many human-centric concepts. To begin addressing this, we provide a comprehensive and flexible definition that lends itself to a relatively unambiguous implementation.

Aiming to develop a definition of hostility appropriate to the GitHub context, we set out to capture the complexity of interactions encompassing inappropriate, harmful, and/or disruptive behavior without casting too wide a net. To this end, we use the definitions of five PerspectiveAPI (Lees et al. 2022) production attributes, emotional concepts that can negatively impact conversations: *toxicity*, *severe toxicity*, *identity attack*, *insult*, and *threat*.³ The intuition is that if text exhibits any of these traits in a substantial way, we consider it hostile; we discuss this in detail in Section 3.2.

PerspectiveAPI production attributes also include *profanity* (defined as the use of “swear words, curse words, or other obscene or profane language”). We intentionally exclude it as, arguably, the presence of profanity on GitHub might not

encompass hostility in all contexts. As also mentioned in the introduction, the presence of profane content does not in itself signify hate speech (Jay 2009; Malmasi and Zampieri 2017). In fact, prior work has argued that models relying too heavily on detecting foul language in in-group conversations, which may often be adjacent to OSS interactions, tend to mistakenly conflate toxicity with profanity (Rosenblatt, Piedras, and Wilkins 2022), even possibly reflecting hidden racial and ethnic biases (Sap et al. 2019).

For instance, manual analysis of GitHub content yielding high profanity scores shows that swear/curse words are often used in a non-hostile manner (e.g., paraphrased for anonymity, “*Why the hell is this appearing in my terminal?*” or “*I noticed this sh** too*”).

A tunable definition. Our definition can easily accommodate customization, e.g., to accommodate the different contexts of research studies. Given the modular design of *ASTRO*, the selection (or weighting) of attributes used for hostility classification can be adjusted to better support different analysis objectives. For example, a study that focuses exclusively on toxicity could weigh the dimensions of toxicity and/or severe toxicity higher.

3.2 Hostility Classification

We introduce a two-layer approach to quantify hostility by using the PerspectiveAPI functionality and a Large Language Model (LLM)-based solution to balance scalability and accuracy.

First Layer: Filtering out ostensibly benign comments. Prior work has shown that PerspectiveAPI is ineffective at identifying offensive activity on GitHub (Sarker, Turzo, and Bosu 2020). However, we find it effective at identifying *benign* comments with high precision. By contrast, it often misclassifies benign comments as hostile (Sarker, Turzo, and Bosu 2020). Therefore, we use the API as a filtering mechanism to eliminate comments that are almost certainly benign.

Second Layer: In-depth classification. Having significantly reduced the number of comments, we can afford to use a computationally expensive LLM-based method for the final classification. We ask the LLM to provide a binary value for each attribute as, at least anecdotally, we find the LLMs to be not very consistent with score ranges. In future work, we will explore techniques from the NLP community that use weighted averages of token probability to get a numerical score at of LLMs (Betley et al. 2026). From a deployment perspective, we can select among multiple LLMs for this layer. We conduct a relatively extensive evaluation of several candidate LLMs as discussed in Section 4. Ultimately, we evaluate *ASTRO* on our datasets using llama3-70b-8192, an open-source LLM made available by Meta (70b denotes the number of model parameters, and 8192 indicates that the model supports a context length of 8,192).

Our hostility classification approach is tailored to users and repositories, as discussed next. In the rest of this section, we focus on the second layer of the classification.

User Hostility. To classify whether a user is hostile, we feed the LLM with their list of issue thread comments. However, first, we create a list of “potentially hostile” comments. More

³<https://developers.perspectiveapi.com/s/about-the-api-attributes-and-languages>.

precisely, to improve efficiency when working with large datasets, we discard comments that are confidently non-hostile, i.e., for which PerspectiveAPI returns scores below 0.3. Recall that PerspectiveAPI scores a comment by its perceived impact on a conversation based on a given attribute; by setting a threshold of 0.3, we exclude comments where an average reader would have less than a 30% probability to interpret the comment as encompassing any of the attributes (toxicity, severe toxicity, identity attack, insult, threat). Although we have validated this cutoff by manually inspecting comments with scores less than 0.3 to ensure a negligible number of false negatives, this is another tunable parameter that can be adjusted according to specific needs.

Next, we feed the list of “potentially hostile” comments to llama3-70b-8192 along with the attribute definitions. This yields, for each comment from each user, a *hostility vector* of five dimensions, with each dimension corresponding to a binary value for each attribute. As a default state to minimize false negatives, we flag a user as hostile if at least one of their comments has at least one hostile attribute. Again, this threshold (at least *one*) can be tuned as needed.

Repository Hostility. To identify hostile *repositories*, we feed issue thread comments to the LLM and ask it to determine the five attributes (toxicity, severe toxicity, identity attack, insult, threat). Once again, we first narrow down the content to feed the LLM by excluding comments that are confidently not hostile, i.e., for which PerspectiveAPI returns scores below 0.3 for all five attributes. For each remaining comment, we get a five-dimensional hostility vector, with each dimension corresponding to a binary value for each attribute. Ultimately, we consider, by default, a repository to be hostile if any comment is detected to contain at least one hostile attribute. Naturally, practitioners and researchers could decide to set minimum thresholds of hostile comments and/or issue threads, etc.

4 Evaluating LLMs for Hostility Classification

In this section, we present our evaluation of several LLMs for *ASTRO*’s hostility classification module. We create a ground-truth dataset, evaluate several open-source and proprietary models on it, and perform a small agreement study.

Ground-Truth Dataset. We build and annotate a dataset to include a significant presence of both hostile and benign comments. We start with a sample of all comments in D_{repo} . Due to the scarcity of hostile comments, we use a stratified sampling technique by processing the comments through PerspectiveAPI and filtering out comments that do not meet at least a 0.3 threshold on any attribute. We also exclude comments that contain text in any language other than English. We then randomly sample 100 comments from the resulting strata of “potentially hostile” comments and have a domain expert manually label each comment as either hostile or not, stopping once the dataset has an equal split of 50 hostile and 50 benign comments.

LLM Evaluation. We evaluate both state-of-the-art open-source LLMs (specifically, those hosted on groq.com as of August 2025) and proprietary ones. The former include

Model	Precision	Recall	F1 Score
llama-3.3-70b-versatile	0.657	0.920	0.767
gemma2-9b-it	0.685	1.000	0.813
deepseek-r1-distill-llama-70b	0.719	0.920	0.807
llama3-70b-8192	0.815	0.880	0.846
mistral-saba-24b	0.793	0.920	0.852
gemin-2.5-pro-preview-03-25	0.833	0.600	0.698
gpt-4.1	0.782	0.860	0.819
PerspectiveAPI	0.762	0.320	0.451

Table 2: Summary of our LLM hostility classification evaluation.

llama-3.3-70b-versatile, gemma2-9b-it, deepseek-r1-distill-llama-70b, llama3-70b-8192, and mistral-saba-24b, whereas for the latter we consider gemini-2.5-pro-preview-03-25 and gpt-4.1. We also include, as a baseline, the classifications made by PerspectiveAPI, selecting at least one attribute with a score above 0.7, a threshold shown to adequately balance precision and recall (Nakka 2025; Gervais, Dye, and Chin 2025), to classify comments as hostile.

In Table 2, we report the corresponding precision, recall, and F1 scores. Overall, all LLMs outperform the PerspectiveAPI baseline across the board in terms of F1 score. In particular, while all models as well as the baseline obtain comparable precision, PerspectiveAPI yields poor recall. This might be due to PerspectiveAPI being trained on comments from online forums (e.g., The New York Times and Wikipedia), thus being overly conservative on GitHub content. In contrast, LLMs may be more flexible in understanding context and adapting to different environments. Alternatively, LLMs are known to be extremely effective one shot classifiers in general, and they just might simply outperform the production Perspective API, which existed long before LLMs.

Interestingly, open-source LLMs outperform proprietary models, at least on our sample, e.g., mistral-saba-24b and llama-3.3-70b-8192 are the most accurate. While the former yields a slightly higher F1 score, we ultimately instantiate *ASTRO* with the latter as it achieves higher precision, making it correct more often when classifying content as hostile.

Agreement Study. To assess whether different LLMs yield significantly different determinations, we conduct an agreement study to quantify how similar each model is to the others in comment hostility classification. Table 3 reports the agreement matrix where each cell represents the percentage of ground-truth comments for which the respective models yield the same classification. The highest agreement is between deepseek-r1-distill-llama-70b and llama-3.3-70b-versatile, which is expected, as the former was distilled from the latter. By contrast, the smallest agreements are primarily with PerspectiveAPI, which, as discussed above, suffers from low recall. Overall, all LLM models have reasonable agreement, except for a couple of cases involving Gemini; in particular, our chosen model, llama3-70b-8192, has acceptably high agreement with all other models, confirming its suitability to classify hostile content.

Model	llama-3.3-70b	gemma2-9b	deepseek-r1	llama3-70b	mistral-saba	gemma-2.5-pro	gpt-4.1	PerspectiveAPI
llama-3.3-70b	1.00	0.85	0.86	0.82	0.76	0.66	0.81	0.51
gemma2-9b-it		1.00	0.79	0.77	0.75	0.61	0.76	0.46
deepseek-r1-distill			1.00	0.86	0.76	0.72	0.85	0.57
llama3-70b-8192				1.00	0.78	0.78	0.83	0.61
mistral-saba-24b					1.00	0.72	0.77	0.57
gemma-2.5-pro						1.00	0.73	0.75
gpt-4.1							1.00	0.60
PerspectiveAPI								1.00

Table 3: Agreement matrix with the rate of common predictions between any two models on our ground-truth dataset. Model labels in the first row are abbreviated to ease presentation.

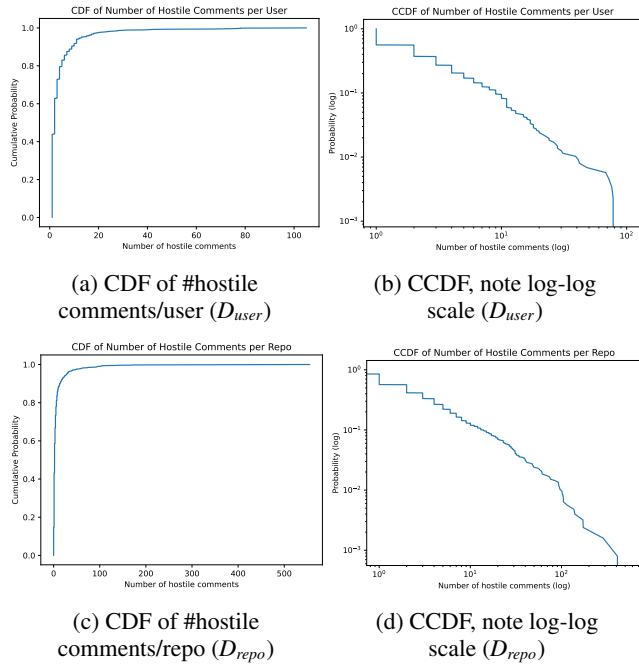


Figure 4: CDFs and CCDFs of the number of hostile comments per user in D_{user} and per repository in D_{repo} .

5 Experimental Evaluation

In this section, we report the results of our experimental evaluation deploying *ASTRO* on our dataset of GitHub users and repositories. First, we provide a general characterization of hostility in our datasets, then we shed light on hostile users from D_{user} and the hostile repositories from D_{repo} .

5.1 Quantifying Hostility

First Layer: Filtering out benign comments. The first layer (PerspectiveAPI) classifies 96.6% of our 735K posts as benign. As mentioned earlier, the precision of this characterization is very high: these posts are almost certainly benign; thus, we do not consider them further. This reduces the workload for the next layer *by a factor of 29*. Overall, we spent a total of \$35.56 on LLM API calls on the filtered comments, which would have been \$1,043.84 had we not incorporated this first layer.

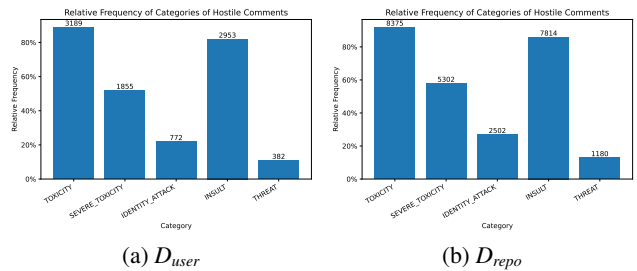


Figure 5: Relative number of comments flagged with each hostility category on D_{user} and D_{repo} . Note that comments may fall into multiple categories.

Second Layer: In-depth classification. We begin by discussing hostility statistics. For D_{user} , we find 874 users (30.1%) and 3,584 comments (1.5%) classified by *ASTRO* as hostile. For D_{repo} , we find 1,070 repositories (85.4%) and 9,135 comments (1.8%) classified as hostile. This is in stark contrast to D_{ctrl} , where no users or comments are hostile.

In Figure 4, we report the Cumulative Distribution Functions (CDFs) and Complementary Cumulative Distribution Functions (CCDFs) of the number hostile comments per user and per repositories in our datasets. In D_{user} , 44.1% of users have only one hostile comment and 83.1% have five or less, with an average of 4.1 hostile comments per user. In D_{repo} , 43.3% of repos have one hostile comment and 78.0% have five or less, with an average of 7.3 per repo. We observe that the number of hostile comments per user in D_{user} shares a distinctly similar trend to the number of comments per repo in D_{repo} . For instance, we find that while most users and repos post relatively few hostile comments, there are a few outliers that post extreme quantities of them. We delve deeper into some of these outliers as case studies below.

In Figure 5, we also report the proportion of hostile comments that were classified with each hostile category. Again, D_{user} and D_{repo} share similar aggregate hostility characteristics at different scales, even though they pertain to different entities. For instance, both datasets have almost the exact same ratios of hostility categories from their hostile comments, potentially indicating that hostile users are prone to participating in environments with the same hostile nature as them.

Attribute	Toxicity		Severe Toxicity		Identity Attack		Insult		Threat	
	<i>corr</i>	<i>agr</i>	<i>corr</i>	<i>agr</i>	<i>corr</i>	<i>agr</i>	<i>corr</i>	<i>agr</i>	<i>corr</i>	<i>agr</i>
Toxicity	1.00	1.00	0.30	0.61	0.10	0.27	0.01*	0.77	-0.07	0.17
Sev. Tox.			1.00	1.00	0.43	0.66	0.42	0.67	0.19	0.54
Id. Attack					1.00	1.00	0.22	0.38	0.22	0.78
Insult							1.00	1.00	0.01*	0.24
Threat									1.00	1.00

Table 4: Pearson correlations (*corr*) and agreement (*agr*) between the hostility categories of hostile comments in D_{user} . NB: All correlations have p-values below 0.001, except the ones marked with *, which are above 0.05.

5.2 Profiling Hostile Users

Types of Hostility. Recall that *ASTRO* labels a user as hostile if the LLM determines that at least one of their comments matches the definition of either toxicity, severe toxicity, insult, threat, or identity attack. We now set to study the interplay between different hostility dimensions. To do so, we compute the Pearson correlation over the 3,546 hostility five-dimensional vectors of the hostile comments in D_{user} , as reported in Table 4. Overall, we do not find any strong correlations. Three categories have moderate positive correlations, specifically: 1) severe toxicity and identity attack, 2) severe toxicity and insult, and 3) severe toxicity and toxicity. This suggests that severe toxicity is a predictor of other forms of hostility and could possibly be used alone to monitor problematic behavior by GitHub users.

We also compute the agreement between types of hostility in Table 4. The strongest agreements are between identity attack and threat (0.778) and toxicity and insult (0.774). In other words, comments in D_{user} are frequently classified with the same value for both pairs of categories. Given that these two pairs of categories have a weak or insignificant correlation between them, we can derive the potential for the existence of a non-linear relationship between them within our datasets.

Who Are the Most Hostile Users? Next, as a case study, we manually inspect a few of the most hostile users in our dataset. The top three most hostile users in our dataset in terms of *number* of hostile comments are: a bot that catalogs user messages from a video game message board onto GitHub issue threads, a user that develops modifications to existing video games, and a user that develops utility libraries for programming languages. We ignore the bot as it simply copies messages from other users in a third-party platform and catalogs them as comments in a repository, meaning that the hostility displayed does not necessarily originate from GitHub.

The video game developer user has 78/1501 (5%) comments classified as hostile. Their average hostility vector, calculated from *ASTRO*’s vector representation of their hostile comments, shows that they mostly post text categorized under toxicity (91%) and insult (71%). The user often posts crude and exaggerated comments such as (paraphrased for anonymity): “*how many f***ing comments did you guys*

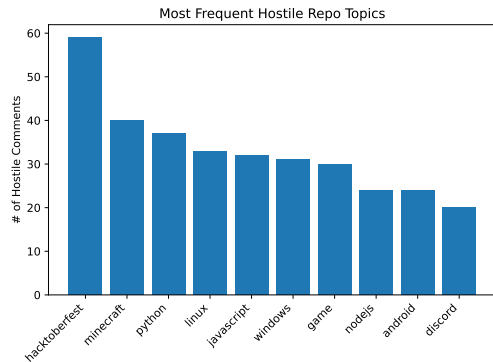


Figure 6: Top 10 most frequent topics of hostile repositories from D_{repo} .

write while I was away?” or “*I killed the last guy who said that.*” They also appear to be a productive developer on GitHub, having created numerous public repositories and collaborating with others.

The utility library developer has 78/3398 (2%) comments categorized as hostile, with most offending categories being toxicity (100%), insult (97%), and severe toxicity (73%). They seem to be quite authoritative in their own repositories, not taking kindly to criticism that they do not agree with – posting, for instance (paraphrased for anonymity): “*I have no problem about being seen as a dictator in my own repositories.*” They also discuss censorship in their issue comments, talking about “*unseen agents in power who enforce political correctness*” that interfere with online discourse, and respond to questions of privacy in their work with responses that include calling the critic a hypocrite for using a pseudonym (referring to their GitHub username) to voice such a concern.

5.3 Profiling Hostile Repositories

Topics. We examine the most frequent topics as tagged in the repositories in D_{repo} , which consists entirely of hostile repositories. This can provide insight into which topics might be more often associated with hostile behavior. In Figure 6, we report the top 10 most frequent hostile topics in D_{repo} for the absolute number of hostile comments. We also report the percentage of hostile comments for these topics in the right-most column of Table 5. Popular topics with frequent hostility include operating systems, e.g., “linux” and “windows,” as well as programming languages (e.g., “python” and “javascript”). The most frequent topic overall is “hacktoberfest,” which refers to an event promoting developers to develop new repositories over October, frequently appearing tagged with other topics. We also observe a relatively high number of hostility on game-based repositories, with the 2nd most frequent topic being “minecraft” (Duncan 2011) and the 7th one being “game.”

We also look at similarities in hostility between topics using *k*-means clustering (Macqueen 1967). To do so, we aggregate all hostile comments from repositories tagged with each topic, represented as hostility vectors from the output

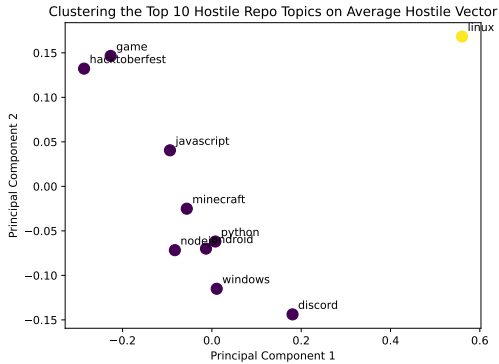


Figure 7: 2d PCA visualization of the top 10 topics wrt the average hostility vectors of their comments.

of *ASTRO*, and calculate the average hostility vector. The result is a hostility profile of each topic, which we report in Table 5. We opt to use the silhouette score method (Shahapure and Nicholas 2020) to find the optimal number of clusters k . We test values of k between 2 and 10 and find that $k = 2$ yields the best score. In Figure 7, we present a visualization of the clusters, projected onto 2 dimensions via Principal Component Analysis (PCA) (Hotelling 1933), showing that “linux” is an outlier topic, demanding closer manual investigation.

Case Study. As visible from Table 5, “Linux” also stands out for having the highest average hostility in every dimension – in particular, for Identity Attack and Threat, the average for “Linux” is often 2-5x the values for the other nine topics. In our dataset, there are 33 hostile repositories tagged with “Linux”, with 299 total hostile comments. Interestingly, 140 of these comments originate from a single repository, namely, “ValveSoftware/Dota-2,” which allows players of a popular online video game to report issues in the version of the game running on Linux and macOS. It contains heavily hostile comments aimed at the developers of the game – e.g., (paraphrased for anonymity): “*How can you lazy devs not fix this f***ing bug for over 10 months?*” and “*Why ban me for over 2 years? F*** you.*” This repository is also the 6th most hostile repository in D_{repo} in terms of raw number of hostile comments, indicating that it skews the average hostility per hostile category heavily upwards. Notably, this repository is not tagged with any other topic from the top 10 list, although it is associated with video games, further supporting our previous observation of video game repositories exhibiting high hostile behavior and aligning with previous studies highlighting the hostile nature of video game communities (Zsila et al. 2022; Pujante Jr 2021; Blackburn and Kwak 2014).

Another case study motivated from Table 5 relates to the “Discord” topic, which has significantly higher relative hostility than the other topics, and is tied with “Linux” for the highest Toxicity. There are 20 repositories tagged with “Discord” in D_{repo} , with 276 out of 6,173 comments classified as hostile. The hostile activity in these repositories appears much more evenly distributed than those of “Linux”,

Topic	Toxicity	Severe Toxic	Identity Attack	Insult	Threat	% Hostile Comms
Hacktoberfest	0.90	0.34	0.09	0.70	0.10	0.17%
Minecraft	0.96	0.51	0.25	0.84	0.07	0.21%
Python	0.94	0.58	0.27	0.88	0.08	0.21%
Linux	0.98	0.81	0.60	0.92	0.52	0.16%
Javascript	0.89	0.48	0.24	0.79	0.10	0.19%
Windows	0.97	0.62	0.22	0.91	0.07	0.15%
Game	0.89	0.38	0.13	0.69	0.14	0.18%
Nodejs	0.95	0.55	0.21	0.83	0.03	0.16%
Android	0.96	0.59	0.23	0.86	0.08	0.16%
Discord	0.98	0.76	0.34	0.91	0.10	0.32%

Table 5: Average hostility dimensions of comments from D_{repo} repositories for the top 10 most frequent topics. In the right-most column, we also report the percentage of hostile comments for each topic.

with the most hostile repositories having 58 (21.0%) and 57 (20.7%) of the hostile comments, respectively. Interestingly, the repositories here generally do not develop any “Discord” related tools, but in fact tag this topic to bring attention to a linked Discord (Discord 2025) where the developers engage with their user community while using the repository for, e.g., bug reports or feature requests. For instance, the most hostile “Discord” repository is “UseInterstellar/interstellar,” which is dedicated to developing a web proxy browser and links to a Discord server with over 45K members. However, in the issue threads of this repository, users are frequently met with hostility: for example, one user replying to a feature request with (paraphrased for anonymity): “*Are you mentally challenged?*” or another comment being answered with: “*Hmm you seem like a black child.*”

Notably, these hostile communications originated from non-contributor GitHub users, and so we recommend future developers of “Discord”-tagged projects to implement careful moderation techniques to avoid such hostile activity. Moreover, we anecdotally find a number of misspelled instances of the “n-word.”

6 Related Work

We now review prior work on detecting and studying hostility on GitHub as well as on other platforms.

Detecting Hostility on GitHub and OSS. Previous research has experimented with different tools to detect hostility in the context of Open Source Software (OSS). Raman et al. (2020) present a preliminary empirical evaluation of an SVM model to detect toxic issue comments on an old (to the best of our knowledge, no longer available) dataset of GitHub repositories known as GHTorrent (Gousios and Spinellis 2012). Also, Ferreira, Rafiq, and Cheng (2024) compare classic machine learning (ML) approaches to BERT (Devlin et al. 2019) to detect incivility in code review comments and in 404 GitHub issues locked for being “too heated” (Ferreira, Adams, and Cheng 2022). Sarker et al. (2023) use ML to detect toxicity in code review comments and identify specific toxic phrases, while Ehsani, Rezapour, and Chatterjee (2025) use ensemble learning.

Mishra and Chatterjee (2024) present preliminary results on the potential of a zero-shot ChatGPT-based approach for toxicity detection on GitHub comments from GHTorrent. Additionally, some work has focused on detecting gender-based discrimination in issue comments (Sultana et al. 2023) and pull request comments (Sultana 2022) as well as assessing the effectiveness of LLMs on these tasks (Sultana and Begum Kali 2024). Overall, while preliminary work points to the potential of using LLMs to detect hostility on GitHub, our work is the first to do so definitively, using an efficient, accurate, and scalable pipeline on a reasonably large dataset.

Characterizing Hostility on GitHub. Prior work has presented qualitative or mixed-methods studies of hostile activity on GitHub. Ehsani, Rezapour, and Chatterjee (2023) associate different types of toxic behaviors with corresponding moral principles, showing how moral foundations theory can be applied to understand online incivility. Miller et al. (2022) manually analyze 100 GitHub issue threads to look into the prevalence and types of toxic interactions, while Ferreira, Adams, and Cheng (2022) study locked GitHub threads to determine whether they actually contain uncivil discussion and what triggers escalation. Relatively larger-scale studies include (Ehsani et al. 2024), which releases an annotated dataset of 5,961 comments from 404 locked GitHub issues to categorize different types of incivility.

Researchers have also analyzed the relationship between users’ genders and ethnicities with the likelihood of experiencing toxicity on GitHub, with preliminary work pointing at disparities in exposure to hostile interactions (Sultana, Uddin, and Bosu 2024). Zhou et al. (2024) show how emoji usage in GitHub issues can reduce resolution time and increase user participation. Finally, studies on open-source miscommunication complement this line of work, showing how misunderstandings in GitHub issues can escalate conflicts (Hasan et al. 2024). Overall, prior work on GitHub toxicity is predominantly catered to specific subsets of OSS communication, as opposed to our broader, large-scale detection and profiling of hostile behavior on GitHub.

Hostility on Other Platforms. Hostile behavior has also been studied on other software development platforms, including Gitter and Stack Overflow. Researchers have evaluated toxicity detectors on Gitter (Sarker, Turzo, and Bosu 2020) and established a fine-grained toxicity detection approach using multiple toxicity types (Tian et al. 2024) or a social exclusion taxonomy (Savarimuthu et al. 2023). Some studies compare the effectiveness of off-the-shelf tools (Novielli et al. 2020) and pre-trained language models (Shafikuzzaman et al. 2024) in detecting sentiment in software projects, which has been shown to be positively correlated with developer productivity (Graziotin, Wang, and Abrahamsson 2014; Ortu et al. 2015).

Finally, seminal prior work has presented numerous studies analyzing and/or detecting hostile content on various social network platforms, e.g., Twitter (Silva et al. 2016; Mondal, Silva, and Benevenuto 2017; Davidson et al. 2017; Olteanu et al. 2018), YouTube (Otoni et al. 2018), Instagram (Hosseinmardi et al. 2015), Reddit (Olteanu et al. 2018), 4chan (Hine et al. 2017), and so on.

7 Conclusion

This paper presented an exploratory study of hostility on GitHub. We introduced *ASTRO*, a systematic framework for classification and profiling of hostile GitHub user-generated content, and instantiated it on a dataset comprising 2.3k GitHub users, 1.3k repositories, and 735k comments.

Our work started from the two research questions in Section 1, which we addressed as follows:

- **(RQ1)** Our two-layer method achieves good performance and cost, reporting an F1 score of 0.85 on our annotated ground-truth dataset, while reducing the cost of using an LLM on the entire dataset by a factor of roughly 29 through a PerspectiveAPI-based pre-filtering.
- **(RQ2)** We successfully identify interesting patterns of hostile behavior. First, we shed light on the relation between different hostility dimensions, such as identity attacks and threats. Second, we focus on communities that are more conducive to hostile behavior, including projects associated to video game and Linux topics.

Outlook and Community Implications. We are confident that our work will help detect, model, and reduce hostility on GitHub and other OSS platforms to encourage collaboration and improve open-source communities. Arguably, hostility can have devastating consequences both on contributors and new users. Hostile communication can demotivate contributors, leading to withdrawal, particularly among maintainers who volunteer time (Ehsani, Rezapour, and Chatterjee 2025), and the expertise loss can deteriorate the quality and continuity of projects (Raman et al. 2020).

Moreover, hostile behavior can derail technical discussions and reduce the likelihood of resolving issues, integrating contributions effectively, etc. In particular, newcomers may be especially sensitive to unhealthy interactions in issue discussions, pull requests, and bug reports (Miller et al. 2022) – hostility can function as a barrier to entry, reducing diversity and limiting the pool of potential contributors. In fact, harmful language can disproportionately affect underrepresented groups, as we indeed find evidence of racist (e.g., misspelled “n-word” slurs) in a few comments.

In this sense, our work not only highlights the need for better/more proactive intervention and moderation techniques, but it also provides a ready-to-use tool to detect/flag potentially problematic users, allowing OSS platforms and maintainers to intervene quickly. We are also confident that work building on *ASTRO* will be able to provide additional data points shedding light on the topics, projects, and issues that are more likely to generate hostile behavior, potentially pre-alerting maintainers that an increased level of attention and/or interventions might be needed.

Limitations. Our dataset is not necessarily representative of all GitHub users and repositories, primarily because GitHub data was not easily accessible to us. To obtain a truly representative sample, we would either need access to GitHub APIs for random user/repository samples, which GitHub does not provide, or collect the entire population of users/repositories on the platform and randomly sample ourselves, which is prohibitive.

While a relatively large GitHub dataset was once available via GHTorrent (Gousios and Spinellis 2012), to our knowledge, authors have stopped providing access, and had anyway stopped updating the data in 2019; with modern culture exhibiting significant change in linguistics (Witalisz 2011), that would represent a significantly limited representation of today’s GitHub ecosystem. Overall, we opt to provide and study a snapshot of hostility that exists today on GitHub and showcase the potential for our *ASTRO* framework to be used as more/larger datasets become available.

Future Work. In the future, we plan to generalize *ASTRO* to work on other OSS platforms and provide *ASTRO* as a standalone platform, allowing users to run it over their dataset and view results on a dashboard-like page. We also plan to expand *ASTRO* to incorporate languages other than English. Consequently, we plan to conduct a more extensive and comprehensive evaluation study to determine the effect different languages have on LLM-based hostility detection. In addition, we will conduct a mirrored evaluation study where we first translate all non-English messages to English and observe the change in hostility detection performance.

References

- Betley, J.; Tan, D.; Warncke, N.; Szyber-Betley, A.; Bao, X.; Soto, M.; Labenz, N.; and Evans, O. 2026. Emergent Misalignment: Narrow Finetuning Can Produce Broadly Misaligned LLMs. *Nature*, 649(8097): 584–589.
- Blackburn, J.; and Kwak, H. 2014. STFU NOOB! Predicting Crowdsourced Decisions on Toxic Behavior in Online Games. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW ’14, 877–888.
- Davidson, T.; Warmusley, D.; Macy, M.; and Weber, I. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. In *ICWSM*.
- Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HTL*.
- Discord. 2025. <https://discord.com/>.
- Duncan, S. C. 2011. Minecraft, beyond construction and survival. *Well Played—A journal on video games, value and meaning*, 1.
- Ehsani, R.; Imran, M. M.; Zita, R.; Damevski, K.; and Chatterjee, P. 2024. Incivility in open source projects: A comprehensive annotated dataset of locked github issue threads. In *MSR*.
- Ehsani, R.; Rezapour, R.; and Chatterjee, P. 2023. Exploring moral principles exhibited in OSS: A case study on github heated issues. In *ESEC/FSE*.
- Ehsani, R.; Rezapour, R.; and Chatterjee, P. 2025. Analyzing Toxicity in Open Source Software Communications Using Psycholinguistics and Moral Foundations Theory. In *NLBE*.
- FAIR Data Principles. 2024. <https://force11.org/info/the-fair-data-principles/>.
- Ferreira, I.; Adams, B.; and Cheng, J. 2022. How heated is it? Understanding GitHub locked issues. In *MSR*.
- Ferreira, I.; Rafiq, A.; and Cheng, J. 2024. Incivility detection in open source code review and issue discussions. *Journal of Systems and Software*, 209.
- Gebbru, T.; Morgenstern, J.; Vecchione, B.; Vaughan, J. W.; Wallach, H.; Iii, H. D.; and Crawford, K. 2021. Datasheets for datasets. *Communications of the ACM*, 64(12).
- Gervais, B. T.; Dye, C.; and Chin, A. 2025. Incivility or Invalidity? Evaluating Perspective API Scores as a Measure of Political Incivility. *American Politics Research*, 53(3).
- Gousios, G.; and Spinellis, D. 2012. GHTorrent: GitHub’s data from a firehose. In *MSR*.
- Graziotin, D.; Wang, X.; and Abrahamsson, P. 2014. Happy software developers solve problems better: psychological measurements in empirical software engineering. *PeerJ*, 2.
- Hasan, K. A.; Loc Mai, V. T.; Wang, C.; Tian, Y.; and Ding, S. H. 2024. A First Look at Self-Admitted Miscommunications in GitHub Issues. In *ASE Workshops*.
- Hine, G.; Onaolapo, J.; De Cristofaro, E.; Kourtellis, N.; Leontiadis, I.; Samaras, R.; Stringhini, G.; and Blackburn, J. 2017. Kek, cucks, and god emperor trump: A measurement study of 4chan’s politically incorrect forum and its effects on the web. In *ICWSM*.
- Hosseinmardi, H.; Mattson, S. A.; Rafiq, R. I.; Han, R.; Lv, Q.; and Mishra, S. 2015. Analyzing Labeled Cyberbullying Incidents on the Instagram Social Network. In *SocInfo*.
- Hotelling, H. 1933. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6).
- Jay, T. 2009. Do offensive words harm people? *Psychology, public policy, and law*, 15(2).
- Jay, T.; and Janschewitz, K. 2008. The pragmatics of swearing. *Journal of Politeness Research: Language, Behavior, Culture*, 4(2).
- Lees, A.; Tran, V. Q.; Tay, Y.; Sorensen, J.; Gupta, J.; Metzler, D.; and Vasserman, L. 2022. A new generation of Perspective API: Efficient multilingual character-level transformers. In *ACM KDD*.
- Macqueen, J. 1967. Some methods for classification and analysis of multivariate observations. In *Berkeley Symposium on Mathematical Statistics and Probability*.
- Malmasi, S.; and Zampieri, M. 2017. Detecting hate speech in social media. In *RANLP*.
- Miller, C.; Cohen, S.; Klug, D.; Vasilescu, B.; and Kaustner, C. 2022. “Did you miss my comment or what?” understanding toxicity in open source discussions. In *ICSE*.
- Mishra, S.; and Chatterjee, P. 2024. Exploring ChatGPT for toxicity detection in GitHub. In *ICSE: New Ideas and Emerging Results*.
- Mondal, M.; Silva, L. A.; and Benevenuto, F. 2017. A Measurement Study of Hate Speech in Social Media. In *HT*.
- Moore, H.; Qi, S.; Hogade, N.; Milojevic, D.; Bash, C.; and Pasricha, S. 2025. Sustainable Carbon-Aware and Water-Efficient LLM Scheduling in Geo-Distributed Cloud Datacenters. *arXiv preprint arXiv:2505.23554*.

- Nakka, N. 2025. An Evaluation of the Google Perspective API by Race and Gender. In *ACM WebSci*.
- Novielli, N.; Calefato, F.; Dongiovanni, D.; Girardi, D.; and Lanubile, F. 2020. Can we use se-specific sentiment analysis tools in a cross-platform setting? In *MSR*.
- Olteanu, A.; Castillo, C.; Boy, J.; and Varshney, K. R. 2018. The Effect of Extremist Violence on Hateful Speech Online. In *ICWSM*.
- Ortu, M.; Adams, B.; Destefanis, G.; Tourani, P.; Marchesi, M.; and Tonelli, R. 2015. Are bullies more productive? Empirical study of affectiveness vs. issue fixing time. In *MSR*.
- Otoni, R.; Cunha, E.; Magno, G.; Bernadina, P.; Meira, W.; and Almeida, V. 2018. Analyzing Right-wing YouTube Channels: Hate, Violence and Discrimination. In *WebSci*.
- Pamungkas, E. W.; Basile, V.; and Patti, V. 2023. Investigating the role of swear words in abusive language detection tasks. *Language Resources and Evaluation*, 57(1).
- Pujante Jr, N. T. 2021. Speech for fun, fury, and freedom: Exploring trash talk in gaming stations. *Asian Journal of Language, Literature and Culture Studies*, 4(1).
- Raman, N.; Cao, M.; Tsvetkov, Y.; Kästner, C.; and Vasilescu, B. 2020. Stress and burnout in open source: Toward finding, understanding, and mitigating unhealthy interactions. In *ICSE: New Ideas and Emerging Results*.
- Rosenblatt, L.; Piedras, L.; and Wilkins, J. 2022. Critical perspectives: A benchmark revealing pitfalls in PerspectiveAPI. In *Workshop on NLP for Positive Impact*.
- Sap, M.; Card, D.; Gabriel, S.; Choi, Y.; and Smith, N. A. 2019. The risk of racial bias in hate speech detection. In *ACL*.
- Sarker, J.; Turzo, A. K.; and Bosu, A. 2020. A benchmark study of the contemporary toxicity detectors on software engineering interactions. In *APSEC*.
- Sarker, J.; Turzo, A. K.; Dong, M.; and Bosu, A. 2023. Automated identification of toxic code reviews using toxicr. *ACM Transactions on Software Engineering and Methodology*, 32(5).
- Savarimuthu, B. T. R.; Zareen, Z.; Cheriyan, J.; Yasir, M.; and Galster, M. 2023. Barriers for social inclusion in online software engineering communities-a study of offensive language use in gitter projects. In *EASE*.
- Sayago-Heredia, J.; Sailema, G. C.; Pérez-Castillo, R.; and Piattini, M. 2025. Analyzing the Correlation Between Toxic Comments and Code Quality. *Journal of Software: Evolution and Process*, 37(2).
- Shafikuzzaman, M.; Islam, M. R.; Rolli, A. C.; Akhter, S.; and Seliya, N. 2024. An Empirical Evaluation of the Zero-shot, Few-shot, and Traditional Fine-tuning Based Pre-trained Language Models for Sentiment Analysis in Software Engineering. *IEEE Access*.
- Shahapure, K. R.; and Nicholas, C. 2020. Cluster quality analysis using silhouette score. In *IEEE DSAA*.
- Silva, L.; Mondal, M.; Correa, D.; Benevenuto, F.; and Weber, I. 2016. Analyzing the Targets of Hate in Online Social Media. In *ICWSM*.
- Sultana, S. 2022. Identifying Sexism and Misogyny in Pull Request Comments. In *ASE*.
- Sultana, S.; and Begum Kali, M. 2024. Exploring ChatGPT for identifying sexism in the communication of software developers. In *PETRA*.
- Sultana, S.; Sarker, J.; Israt, F.; Paul, R.; and Bosu, A. 2023. Automated Identification of Sexual Orientation and Gender Identity Discriminatory Texts from Issue Comments. *arXiv:2311.08485*.
- Sultana, S.; Uddin, G.; and Bosu, A. 2024. Assessing the influence of toxic and gender discriminatory communication on perceptible diversity in OSS projects. *arXiv:2403.08113*.
- Tian, J.; Bao, L.; Pan, S.; and Hu, X. 2024. Analyzing and detecting toxicities in developer online chatrooms: A fine-grained taxonomy and automated detection approach. In *APSEC*.
- Torvalds, L. 2013. 'Re: [00/19] 3.10.1-Stable Review' - MARC. <https://marc.info/?l=linux-kernel&m=137391223711946&w=2>.
- Tyson, M. 2025. Milestone one billionth Github repo is just the word "sh*t". <https://www.tomshardware.com/software/milestone-one-billionth-github-repo-is-just-the-word-sh-t>.
- Witalisz, A. 2011. Linguistic globalization as a reflection of cultural changes. In *Annual Conference of the Global Awareness Society International*.
- Zhao, L.; and Deek, F. P. 2004. User collaboration in open source software development. *Electronic Markets*, 14(2).
- Zhou, Y.; Lu, X.; Gao, G.; Mei, Q.; and Ai, W. 2024. Emoji promotes developer participation and issue resolution on GitHub. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 18, 1833–1846.
- Zsila, Á.; Shabahang, R.; Aruguete, M. S.; and Orosz, G. 2022. Toxic behaviors in online multiplayer games: Prevalence, perception, risk factors of victimization, and psychological consequences. *Aggressive Behavior*, 48(3).

Paper Checklist

1. For most authors...
 - (a) Would answering this research question advance science without violating social contracts, such as violating privacy norms, perpetuating unfair profiling, exacerbating the socio-economic divide, or implying disrespect to societies or cultures? **We only use publicly available data in our study that does not require signing in to any account or platform, as described in Section 2.**
 - (b) Do your main claims in the abstract and introduction accurately reflect the paper's contributions and scope? **Yes. We provide a summary of contributions at the end of the Introduction in Section 1.**
 - (c) Do you clarify how the proposed methodological approach is appropriate for the claims made? **Yes, see Section 3.2.**
 - (d) Do you clarify what are possible artifacts in the data used, given population-specific distributions? **Yes, see Section 2.**
 - (e) Did you describe the limitations of your work? **Yes, see the Limitations paragraph in Section 7.**
 - (f) Did you discuss any potential negative societal impacts of your work? **Yes, see the Implications paragraph in Section 7.**
 - (g) Did you discuss any potential misuse of your work? **Yes, see the Implications paragraph in Section 7.**
 - (h) Did you describe steps taken to prevent or mitigate potential negative outcomes of the research, such as data and model documentation, data anonymization, responsible release, access control, and the reproducibility of findings? **Yes. We anonymize all reported results to prevent reverse-engineering users' identities discussed in our work. Additionally, all of the data used in our work is publicly available, allowing for reproducibility.**
 - (i) Have you read the ethics review guidelines and ensured that your paper conforms to them? **Yes.**
2. Additionally, if your study involves hypotheses testing...
 - (a) Did you clearly state the assumptions underlying all theoretical results? *N/A.*
 - (b) Have you provided justifications for all theoretical results? *N/A.*
 - (c) Did you discuss competing hypotheses or theories that might challenge or complement your theoretical results? *N/A.*
 - (d) Have you considered alternative mechanisms or explanations that might account for the same outcomes observed in your study? *N/A.*
 - (e) Did you address potential biases or limitations in your theoretical framework? *N/A.*
 - (f) Have you related your theoretical results to the existing literature in social science? *N/A.*
 - (g) Did you discuss the implications of your theoretical results for policy, practice, or further research in the social science domain? *N/A.*
3. Additionally, if you are including theoretical proofs...
 - (a) Did you state the full set of assumptions of all theoretical results? *N/A.*
 - (b) Did you include complete proofs of all theoretical results? *N/A.*
4. Additionally, if you ran machine learning experiments...
 - (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? *N/A.*
 - (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? *N/A.*
 - (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? *N/A.*
 - (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? *N/A.*
 - (e) Do you justify how the proposed evaluation is sufficient and appropriate to the claims made? *N/A.*
 - (f) Do you discuss what is "the cost" of misclassification and fault (in)tolerance? *N/A.*
5. Additionally, if you are using existing assets (e.g., code, data, models) or curating/releasing new assets, **without compromising anonymity**...
 - (a) If your work uses existing assets, did you cite the creators? **Yes, such as with our use of PerspectiveAPI (Lees et al. 2022).**
 - (b) Did you mention the license of the assets? *N/A.*
 - (c) Did you include any new assets in the supplemental material or as a URL? *N/A.*
 - (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? **Yes. We only use publicly available data that does not require logging in to an account to view.**
 - (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? **Yes. A significant amount of our data contains offensive content, as that is the main theme of our work. Additionally, we anonymize the data and report results so that they cannot be used to reverse-engineer user identities.**
 - (f) If you are curating or releasing new datasets, did you discuss how you intend to make your datasets FAIR (see FAIR Data Principles (2024))? *N/A.*
 - (g) If you are curating or releasing new datasets, did you create a Datasheet for the Dataset (see Gebru et al. (2021))? *N/A.*
6. Additionally, if you used crowdsourcing or conducted research with human subjects, **without compromising anonymity**...
 - (a) Did you include the full text of instructions given to participants and screenshots? *N/A.*
 - (b) Did you describe any potential participant risks, with mentions of Institutional Review Board (IRB) approvals? *N/A.*

- (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? N/A.
- (d) Did you discuss how data is stored, shared, and de-identified? N/A.