CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Mining Ethical Concerns from Open-Source Project Repositories

Master's Thesis 2026

Master's Thesis in Computer science and engineering

Ruixuan Li

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2026

Master's Thesis 2026

Mining Ethical Concerns from Open-Source Project Repositories

Master's Thesis 2026

Ruixuan Li

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2026

Mining Ethical Concerns from Open-Source Project Repositories
Ruixuan Li

Supervisor: Irum Inayat, Department of Computer Science and Engineering
Examiner: Farnaz Fotrousi, Department of Computer Science and Engineering

Master's Thesis 2026
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2026

Ruixuan Li
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

In the process of open-source software development, ethical concerns arise when development decisions involve conflicting values or the interests of stakeholders. Although previous research has identified certain ethical issues in open-source repositories, such as license violations and incorrect attribution, broader ethical issues remain under-researched. This study employs a mixed-methods approach to examine these issues. In the first phase, the researcher conducted an inductive thematic analysis of 134 comments from GitHub issues and pull requests from multiple open-source repositories, identifying 18 types of ethical issues and grouping these issues into six themes. The researcher used Cohen's kappa and Fleiss' kappa coefficients to assess inter-rater reliability, and the results indicated substantial agreement. In the second phase, the researcher trained supervised machine learning classifiers using a labeled dataset containing 3,084 comments to enable automatic detection of these types of comments. The best-performing binary classifier achieved a macro F1 score of 0.68 in cross-validation, and the theme-level classifier achieved 0.79. External validation on an unseen repository (ansible/ansible) yielded a binary macro-F1 of 0.75 and a restricted theme-level macro-F1 of 0.53. In the third phase, the classifier trained in the RQ2 phase was applied to 54,605 comments. These comments were collected from five repositories over a 36-month period. The results of the third-phase analysis indicate that ethical concerns are a persistent feature of the open-source development process. Furthermore, significant governance events can lead to a measurable short-term increase in the expression of ethical concerns, and the findings suggest that ethical concern frequency may be more closely associated with structural community factors than with short-term activity fluctuations.

Keywords: ethics in software engineering, open-source software, mining software repositories, thematic analysis, text classification, machine learning

Acknowledgements

I would like to thank my supervisor, Irum Inayat, for her guidance throughout this project. There were moments when I lost sight of where the research was heading, or simply doubted whether I was on the right track. She always knew what to suggest. Her encouragement during those uncertain stretches meant more than she probably realizes. I am also grateful to Farnaz Fotrousi for her thoughtful input at various stages of the thesis.

I owe a particular thank you to my two classmates who helped with the inter-rater reliability experiments. They agreed to take this on during a busy period in their own studies. Their patience and careful attention made that part of the research possible.

To my friends: thank you for the late nights you stayed up with me when the work ran long. Thank you for the meals and the walks when things got overwhelming. I am not sure I would have got through some of those stretches without you. To my family, thank you for the support that was always there in the background, quiet but constant.

Ruixuan Li, Gothenburg, June 2026

Contents

List of Figures

List of Tables

1

Introduction

Ethical concerns in this study refer to the tensions that happen when development decisions require compromises between different values or stakeholder interests. This is what makes them different from purely technical problems. For example, fixing a bug or designing an API is just about code. But if a maintainer closes a pull request without any explanation, contributors will question if the process is fair. We can clearly see this in real-world events. In 2018, the Python community went through a major crisis when a new proposal (PEP 572) caused a huge fight over who actually had the decision-making power. This argument became so bad that Guido van Rossum decided to step down permanently [1]. In the same year, the Linux community faced a similar challenge when Linus Torvalds had to apologize for his long history of cursing at contributors in emails [2]. These are not isolated cases. They show standard problems where technical choices conflict with respect, power, and fairness.

Managing these ethical issues is particularly difficult because open-source development works very differently from corporate software engineering. In a regular company, HR policies, codes of conduct, and clear management layers can easily step in and solve conflicts. Open-source maintainers, by contrast, cannot enforce community rules the same way. There is no boss to manage volunteers who are spread across different countries and cultures. Because of this, when fights get too serious, communities are forced to build new governance systems. For example, Python replaced its BDFL model with an elected Steering Council, and the Linux kernel formally adopted a Code of Conduct. Both changes are directly in response to the 2018 crises.

Despite how serious these problems are, we still do not have enough empirical research on them. Some studies look at toxicity in developer comments, but they do not provide a complete framework to cover all ethical issues in the open-source community. Win et al. [3] represent an important methodological step by proposing a taxonomy of 15 types of unethical behavior and developing Etor, an ontology-based tool that automatically detects six of these types using Semantic Web Rule Language (SWRL) rules over GitHub repository attributes. This study builds on this foundation by taking a different approach: rather than modeling ethical concerns through predefined ontology rules, it applies supervised machine learning trained on manually labeled examples, providing a data-driven method that can handle concern

types expressed in natural language.

To train this model, we applied a strict coding framework where coders only labeled concerns explicitly stated in the text. For instance, if a contributor directly wrote, "this decision was made without any community input," this was marked as a fairness concern. However, if a maintainer simply closed an issue without a comment, we did not assume it was unfair unless the contributor themselves brought up that issue in the thread. We avoided inferring hidden intentions and refrained from making personal judgments about whether someone's behavior was objectively "wrong." This approach keeps our data grounded in the actual words used by the developers.

1.1 Problem Description

Although open-source ethical concerns are getting more attention, current empirical research is still limited. Most studies on toxic behavior only look at obvious harmful things, like personal attacks or hostile language. They do not give us a complete framework for the wider ethical tensions in daily development. Win et al. [3] did great work by creating a taxonomy of 15 unethical behaviors and using rules to detect them automatically. But because their method relies on strict rules and predefined conditions, it struggles to catch concerns that are written in natural language. This is why we use supervised machine learning. It provides a data-driven approach instead of a rule-driven one, making it easier to handle a broader range of concerns.

At the same time, we still know very little about how these ethical concerns change over a project's lifecycle, or how they connect to project features like community size, contributor activity levels, and governance context. We need this context. Without it, we cannot predict when ethical tensions will emerge, or make active plans to keep communities healthy.

1.2 Purpose of the Study

The purpose of this study is to identify, categorize, and analyze ethical concerns expressed in open-source software repositories. The study aims to construct an empirically grounded taxonomy of ethical concern types through inductive qualitative analysis of developer communication, develop supervised machine learning models to support automated detection of these concern types, and investigate how ethical concerns relate to project lifecycle and repository characteristics.

1.3 Research Questions

RQ1: What types of ethical concerns are expressed in open-source software repositories?

RQ2: How can automated text-mining and machine-learning methods detect and categorize ethical concerns in developer communication?

RQ3: How do ethical concerns change over the lifecycle of open-source projects?

RQ4: Which project characteristics, patterns, or events are associated with the emergence or increase of ethical concerns?

1.4 Significance of the Study

While prior work has extracted ethical concerns from user-generated content such as app reviews [4] and used rule-based approaches to detect a predefined set of unethical behavior types in open-source projects [3], this study inductively derives a taxonomy from open-source developer communication, replaces handcrafted detection rules with a supervised machine learning classifier, and includes an exploratory temporal analysis of ethical concern patterns across open-source repositories.

This study therefore makes the following contributions to both research and practice. For researchers, this study provides an empirically grounded taxonomy of ethical concern types that can be applied to other open-source projects, as well as a supervised classification approach that extends analysis beyond what manual review alone can achieve. For practitioners, the results can help project maintainers identify when ethical tensions emerge and understand how they relate to project characteristics such as governance structure and contributor activity, informing moderation practices and efforts to build more inclusive open-source communities.

1.5 Thesis Outline

This thesis is structured as follows. Chapter 2 introduces the background of the study. Chapter 3 reviews related work. Chapter 4 presents the theoretical framing. Chapter 5 describes the research methods. Chapter 6 presents the results. Chapter 7 discusses the findings and limitations. Chapter 8 concludes the thesis.

2

Background

This chapter provides the conceptual foundations necessary to understand the study. Section 2.1 introduces the ACM Code of Ethics and explains how it is used in taxonomy construction. Section 2.2 describes GitHub as a communication platform and explains why its artifacts are suitable for studying ethical concerns. Section 2.3 reviews existing ethical classification frameworks and situates the present study within this prior work. Section 2.4 characterizes open-source development and the ethical challenges it presents.

2.1 ACM Code of Ethics

To ground this study, we use the ACM Code of Ethics and Professional Conduct (2018) as our primary framework [5]. The code outlines seven core principles: contributing to society, avoiding harm, being honest, being fair, respecting intellectual property, protecting privacy, and honoring confidentiality. While the IEEE/ACM Software Engineering Code covers similar values, it targets professional engineers working within formal organizational settings. Open-source development, by contrast, relies on volunteers and distributed participation outside of employment contracts. This unique environment makes the broader ACM Code a much better fit for our theoretical foundation. For instance, the second principle of the IEEE/ACM code explicitly requires software engineers to act in the best interests of their client and employer. However, in a typical open-source repository, volunteer contributors do not have a common employer or client to answer to. If a conflict happens between contributors from different countries, they cannot look at employer interests to solve the problem. Instead, the community must rely on broader ACM principles such as fairness and honesty to manage these voluntary relationships.

Several of these principles are particularly relevant to ethical tensions that arise in open-source development. First, the fairness and non-discrimination principles relate to how contributions are reviewed and decisions are made, including whether all contributors are treated consistently and whether decisions are explained. Honesty, meanwhile, relates to how project status, capabilities, and intentions are communicated to users and contributors. Next, the harm avoidance principle focuses on the responsible handling of security vulnerabilities and the consequences of releasing inadequately tested software. Regarding privacy, this principle relates to how user

data is collected and handled. Finally, the respect for intellectual property principle relates to attribution, licensing, and the use of others' code. These connections between ethical principles and development practices form the theoretical basis for the taxonomy developed in this study.

In this study, the ACM Code of Ethics serves two distinct roles. First, we use it alongside Win et al.'s OSS ethical principles before data collection to derive a set of sensitizing dimensions. These dimensions direct our attention toward potential areas of ethical tension during open coding without predetermining the specific concern types we will find. Second, once we inductively identify ethical concern types from developer communication and aggregate them into stable categories, we map each category back to the most relevant ACM principle. Related categories are then grouped into broader themes that correspond to these principles. This mapping directly links the empirically grounded first layer of our taxonomy to established ethical standards at the second layer. Crucially, our study does not apply the ACM framework to judge whether any individual behaved unethically. Instead, it serves purely as a reference vocabulary for organizing and naming the concern types that emerge from the data.

2.2 GitHub as a Communication Platform

GitHub is the dominant hosting platform for open-source software projects, supporting version control, issue tracking, and collaborative code review. Our study focuses on two central types of communication within this platform. Specifically, issues are discussion threads used to report bugs, request features, or raise concerns about a project. Pull requests, meanwhile, are structured proposals to merge code changes into a repository, typically involving discussion between the contributor and reviewers before a change is accepted or rejected. Both artifact types support threaded comment discussions. In this study, individual comments within these threads constitute our primary unit of analysis [6].

We chose these discussion threads for studying ethical concerns because they offer several key advantages. To begin with, they serve as the primary channel where project decisions are made and communicated. Maintainers use them to explain why contributions are accepted or rejected, contributors use them to raise concerns about project direction, and community members use them to respond to one another's proposals. Because decisions with ethical dimensions are fully documented here, these threads provide direct evidence of how ethical tensions are expressed and managed in practice. Furthermore, these public artifacts are permanently archived, making them highly accessible for systematic analysis. Finally, each comment carries valuable metadata, including a timestamp and author information. This metadata supports both qualitative analysis and the temporal analyses we planned for RQ3 and RQ4.

2.3 Existing Ethical Classification Frameworks

Several prior studies have proposed taxonomies or classification frameworks for ethical concerns in software related contexts. These works differ in their scope, methodology, and the types of ethical scenarios they examine, and together they motivate the need for an empirically grounded taxonomy focused on open-source development communication.

To look at distributed work, Kocsis and de Vreede [7] developed a taxonomy of ethical considerations in crowdsourcing. Their deductive approach was grounded in Value Sensitive Design and transparency literature, integrating ethical dilemmas, crowdsourcing models, and affected stakeholders into a structured classification for design and governance. This context shares some characteristics with open-source development due to its distributed, collaborative work involving volunteer participants. Their taxonomy, however, remains conceptual rather than empirically derived. Furthermore, its scope is strictly limited to crowdsourcing as an organizational model rather than open-source software communities.

From a different angle, Biable et al. [8] investigated ethical concerns in software requirements engineering through interviews and focus group discussions with industry practitioners. Their qualitative analysis successfully identified five categories of concerns, including requirements identification problems, quality related problems, and knowledge gaps. Although this framework is grounded in real practitioner experience, its focus stays on the requirements engineering phase of corporate software development. Consequently, it leaves out the collaborative and community driven dynamics that define open-source projects, and it does not examine how ethical concerns are actually expressed in developer communication artifacts.

Win et al. [3] represent the closest prior work to the present study. They developed a taxonomy of 15 types of unethical behavior in open-source projects through analysis of 316 GitHub issues, organized around six ethical principles. Alongside the taxonomy, they developed Etor, an ontology-based detection tool that uses predefined logical rules applied to GitHub repository attributes. While this work is directly relevant, its detection approach relies on handcrafted rules for each concern type, which limits its ability to handle concern types expressed primarily in natural language. Their dataset was collected exclusively through ethical meta-language keyword search, that is, terms that explicitly label a situation as ethically problematic such as "unethical" or "not ethical", without systematic browsing of repository discussions. This risks underrepresenting concern types expressed without explicit ethical terminology, for example, a comment such as "did you even read the documentation?" may constitute Participation Shaming without any participant labeling it as ethically problematic. It is important to note that this does not mean the present study codes implicit or inferred concerns; rather, the concern must still be explicitly expressed or clearly framed in the text, but need not be necessarily accompanied by ethical vocabulary such as "unethical" or "unfair".

Taken together, these prior frameworks either construct categories deductively from

existing ethical principles or rely on predefined rules to detect specific concern types. The present study addresses these limitations in two ways. First, the taxonomy is developed inductively from open-source developer communication collected through a combination of systematic browsing and broad keyword-assisted search, allowing concern types to emerge from what developers actually express rather than from theoretical frameworks imposed in advance. Second, supervised machine learning is applied to enable automated detection across all identified concern types, providing a data-driven approach that does not require handcrafted rules for each category.

2.4 Open-Source Development and Its Ethical Context

Open-source software development is characterized by distributed, voluntary collaboration across organizational and geographical boundaries. Contributors participate based on personal motivation or organizational affiliation rather than employment obligation, and governance structures vary considerably across projects. Some projects operate under formal governance boards with documented decision-making procedures; others rely on informal norms and the judgment of a small number of core maintainers [9]. Maintainers are contributors who have been granted authority to review and accept or reject contributions, and they play a central role in shaping both the technical direction and the social culture of a project.

This context creates ethical challenges that differ from those in corporate software development. In organizational settings, ethical issues are typically managed through formal mechanisms such as HR policies and managerial accountability. In contrast, open-source communities lack employment relationships and formal institutional oversight.

As a result, project maintainers often have limited formal authority to enforce behavioral standards, even though they control access to the codebase. This creates power asymmetries between maintainers and contributors, who depend on maintainer decisions for their work to be accepted. In addition, contributors from different cultural backgrounds may have different expectations about what constitutes respectful communication or fair decision-making. When ethical tensions arise, communities must rely on a combination of informal norms, codes of conduct, and maintainer judgment [9, 10].

3

Related Work

This chapter reviews three strands of prior work. While Section 3.1 examines research on software engineering ethics with a focus on integrating ethical principles into development practices, Section 3.2 shifts to empirical studies on ethical concerns and toxic behavior in open-source communities. Section 3.3 covers text-mining and machine-learning approaches for analyzing developer communication.

3.1 Ethics in Software Engineering

Research on ethics in software engineering has begun to explore how ethical principles can be integrated into development practices. Biable et al. [8] proposed an ethical framework for requirements engineering based on the ACM/IEEE codes of ethics, identifying five categories of concerns and validating them with industry experts. Similarly, Halme et al. [11] introduced Ethical User Stories as a way to incorporate ethical reflection into agile development, demonstrating through empirical evaluation that ethical considerations can be embedded into existing workflows.

Other work has explored the possibility of identifying ethical concerns from unstructured text. For example, prior studies have shown that ethical considerations can be extracted from user-generated content such as reviews [4]. However, these approaches have not been systematically applied to developer communication in open-source repositories, where ethical concerns are often embedded in technical discussions.

Overall, while existing studies demonstrate that ethical frameworks can inform development practices and that ethical signals can be identified in textual data, there remains a lack of empirical work examining how ethical concerns emerge and are expressed in open-source development discussions.

3.2 Ethical Concerns and Toxic Behavior in Open-Source Communities

Empirical research on harmful behavior in open-source communities has established that personal attacks, exclusionary language, and hostile exchanges are present in

development discussions and have measurable effects on community health. Miller et al. [12] analyzed GitHub discussions and identified behaviors including arrogance, antagonistic exchanges, and dismissive communication that discourage contributor participation. Sarker et al. [13] conducted a large-scale investigation of toxicity on GitHub, confirming that toxic patterns are widespread and associated with reduced contributor retention. While these studies establish the social dimensions of ethical failure in open-source communities, they focus on overtly harmful behavior and do not provide a structured framework for the full range of ethical tensions that arise in collaborative development.

Win et al. [3] represent the most directly related prior work. They conducted a study of 316 GitHub issues and pull requests to develop a taxonomy of 15 types of unethical behavior in open-source projects, organized around six ethical principles. They also developed Etor, an ontology-based tool that uses Semantic Web Rule Language (SWRL) rules to automatically detect six of these types by modeling GitHub repository attributes as a knowledge graph and applying logical inference rules. Their evaluation on 195,621 GitHub issues from 1,765 repositories demonstrates the feasibility of automated detection. The present study extends this line of work in two ways: it applies supervised machine learning rather than ontology-based rules, providing a data-driven approach that does not require handcrafted detection logic for each concern type; and it expands the taxonomy to 18 types through inductive analysis of a dataset that was not restricted to issues containing ethical meta-language keywords.

Research on codes of conduct in open-source projects highlights the limitations of formal mechanisms for managing ethical issues. Singh et al. [10] found that codes of conduct have mixed effectiveness in promoting equal treatment, suggesting that formal policies alone are insufficient without consistent enforcement. Similarly, Hsieh et al. [9] examined moderation practices, including the use of automated tools, and showed that maintaining community norms remains challenging in distributed, volunteer-driven settings. Together, these findings indicate that ethical tensions in open-source projects cannot be fully addressed through formal rules alone.

3.3 Text Mining and Machine Learning for Developer Communication

Prior research has established that text mining and machine learning can identify behavioral and attitudinal patterns in software engineering artifacts at scale, providing a methodological foundation for the automated classification approach used in RQ2.

Lin et al. [14] conducted a systematic literature review of 185 studies on opinion mining in software engineering, documenting how researchers have applied computational text analysis to issues, pull requests, commit comments, and app reviews to identify sentiments, feature requests, and developer emotions. Their review demonstrates that text classification is an established approach for extracting meaningful

signals from the kinds of artifacts examined in the present study. Importantly, the review also highlights that tools trained on general-purpose text often perform poorly on software engineering content, motivating the use of domain-specific labeled datasets for training classifiers.

This concern is also reflected in research on toxicity detection in open-source communities. Sarker et al. [13] applied a domain-specific toxicity detector trained on software engineering interactions to classify over 100 million GitHub pull request comments, demonstrating both the feasibility of automated classification at scale and the importance of domain-specific training data. Kallis et al. [15] further showed that text-based classifiers can generalise reliably across heterogeneous repositories: their Ticket Tagger tool achieved F-measures above 80% for all three issue categories when tested on 30,000 issues drawn from over 12,000 projects.

For sentence-level representations, this study uses all-MiniLM-L6-v2, a lightweight model from the Sentence Transformers library developed by Reimers and Gurevych [16], which produces fixed-size embeddings optimised for semantic similarity. Since all-MiniLM-L6-v2 is trained on general-purpose text rather than software engineering content, it is unclear whether semantic similarity in a general domain corresponds to meaningful distinctions in ethical concern language. This uncertainty motivates the comparative evaluation of TF-IDF and sentence embeddings reported in Section 6.2.

Kalliamvakou et al. [6] showed that GitHub repositories vary enormously in their purpose, size, and community structure, ranging from personal and experimental projects to large collaborative communities. This diversity has implications for cross-repository generalisation: a classifier trained on comments from one set of communities may not transfer well to repositories with different governance structures or communication norms, a limitation that is directly relevant to the external validation results discussed in Section 7.2.4.

Taken together, these studies demonstrate that supervised machine learning applied to labeled datasets of GitHub comments is a feasible approach for identifying behavioral patterns in open-source development. Unlike prior work that focuses on binary toxicity classification, the present study applies this approach to a finer-grained set of 18 ethical concern types, for which no prior labeled training data exists, requiring the construction of the dataset described in Section 5.3.

4

Theory

This chapter introduces the concepts and methods that the study builds on. Sections 4.1 and 4.2 cover the qualitative side: how categories are built from data using thematic synthesis, and how inter-rater reliability is used to check that the coding is consistent. Sections 4.3 through 4.7 cover the machine learning side: how text is prepared and turned into features, what classifiers are used, how class imbalance is handled, and how the models are evaluated.

4.1 Thematic Synthesis as a Basis for Taxonomy Construction

In this study, thematic synthesis is used to build conceptual categories directly from the collected data. Instead of starting with fixed categories and checking if the data fits, the researcher reads the text, gives descriptive labels to the observations, and groups similar labels together until stable categories form [17]. This approach works well because our goal is to discover what types of ethical concerns exist in the data, rather than testing an existing framework.

Our coding procedure follows the five steps from Cruzes and Dybå [18]: data extraction, open coding for concept translation, descriptive theme formation, mapping themes to broader analytical categories, and reliability assessment. Crucially, the 18 specific concern types in our taxonomy were established from the data before connecting them to any theoretical framework. Only after a category became stable did the researcher map it to the most relevant ethical principle from the ACM Code of Ethics. This sequencing matters because it prevents the theory from shaping what gets coded. As Thomas [19] discussed, inductive approaches let categories emerge from data, while deductive approaches start from an existing framework. Our study clearly follows an inductive approach for the first layer of the taxonomy.

Before reading any data, a set of sensitizing dimensions was prepared based on the ACM code of ethics framework and on Win et al.'s prior work [3]. These dimensions are not categories; they are more like reminders of what to pay attention to. For example, knowing in advance that fairness concerns might appear in the data makes it less likely that a relevant comment will be overlooked during reading. But the dimensions do not determine what will be found. If something appears in the data

that does not fit any dimension, it still gets coded. And if a dimension does not appear frequently enough in the data to form a real category, it is dropped. Three dimensions, Self-Interest, Welfare, and Sustainability, were excluded for this reason.

During coding, we used negative case analysis to check the boundaries of our categories. For each emerging category, the researcher actively looked for cases that contradicted the definition. We then refined the definition until no more contradicting cases were found. This step stops categories from being defined too broadly or too loosely.

4.2 Inter-Rater Reliability as a Validity Criterion

When coding qualitative data, different people reading the same text may reach different conclusions. Inter-rater reliability (IRR) is a way of measuring how much agreement there is between independent coders applying the same scheme to the same data [20]. High agreement suggests that the coding guidelines are clear and that the categories are well-defined enough to be applied consistently. Low agreement suggests the opposite and usually points to specific boundaries that need to be clarified.

This study uses two statistics. Cohen's Kappa [21] measures pairwise agreement between two coders. Fleiss' Kappa [22] extends this to three or more coders at once. Both statistics range from 0 to 1, where higher values indicate better agreement. Following the interpretation scale of Landis and Koch [23], values between 0.61 and 0.80 indicate substantial agreement, and values above 0.80 indicate almost perfect agreement.

We assess IRR at two points. In RQ1, it checks whether the taxonomy is clear enough for independent coders to apply reliably. In RQ2, it checks whether the same coding framework can be applied consistently to a different dataset collected through keyword search rather than systematic browsing. A separate IRR assessment is needed for RQ2 because comments retrieved through category-specific keywords may look quite different from those found through open browsing, and it is not safe to assume that reliability transfers automatically [20].

For RQ2, we designed a two-stage process. The first stage (IRR-A) checks a simple binary choice: does a comment have an ethical concern or not? The second stage (IRR-B) focuses on choosing one of the 18 fine-grained categories for the comments that do have concerns. Separating these two steps helps us see where disagreements come from. If coders agree on the presence of a concern but disagree on the category, it means specific category boundaries need sharpening. This design follows the same logic as Gachechiladze et al. [24], who separately assessed whether a comment expressed anger and what direction that anger was directed in.

The external validation set was labeled by the same primary researcher using the same coding framework, following the same keyword-based retrieval and labeling procedure applied to the RQ2 development set. Therefore, we did not run a separate

IRR check for this set. The reason is that the purpose of external validation is to test whether the classifier generalizes to an unseen repository, not to evaluate the coding framework itself, which had already been validated through the IRR assessments described above.

4.3 Text Preprocessing for Natural Language Classification

Machine learning classifiers work with numbers, not raw text. Before any features can be extracted, the text needs to be cleaned and standardized. The goal is to remove tokens that carry little useful information while keeping those that are relevant to the classification task.

The preprocessing steps used in this study are lowercasing, removal of non-alphabetic characters, and lemmatization. Lemmatization maps different forms of the same word to a shared base form, so that "ignoring", "ignored", and "ignores" are all treated as the same token [25]. This reduces the vocabulary size without losing meaning. Lemmatization is performed using the WordNet Lemmatizer from NLTK.

Stop words, common function words like "the", "a", and "of", are removed because they appear everywhere and do not help distinguish one category from another [25]. There is one exception: negation words like "not", "never", and "nor" are kept. In ethical language, negation often changes the meaning entirely. "This decision is fair" and "this decision is not fair" are very different, and removing "not" would make them look the same to the classifier.

GitHub comments also contain tokens that are specific to the platform and not useful for classification. At-mentions (@username), issue reference numbers such as #1234, URLs, and code blocks are all removed. Keeping them would add noise to the feature space without contributing any signal about ethical concerns.

4.4 Feature Representations

After cleaning the text, we convert it into numerical vectors. This study evaluates two representation types: TF-IDF vectors and sentence embeddings.

TF-IDF stands for term frequency-inverse document frequency. It gives each word in a document a weight based on how often it appears in that document compared to how often it appears across all documents [25]. Words that appear frequently in one document but rarely in others get high weights. The reason is that such words are more likely to be specific to that document and therefore more useful for distinguishing it from others. In this study, TF-IDF is computed over individual words and two-word phrases. The vocabulary size is treated as a hyperparameter, with values of 20,000 and 50,000 explored during hyperparameter search. The two values were selected to examine whether a smaller vocabulary (20,000) that

filters out lower-frequency terms or a larger vocabulary (50,000) that retains more domain-specific expressions performs better on this dataset; the optimal value was determined through hyperparameter search. Two-word phrases are included because expressions like “no response” or “without consent” are more informative than either word on its own [26]. TF-IDF representations are expected to work well for categories that tend to be expressed using a fairly consistent set of terms, such as Irresponsible Disclosure, where phrases like “responsible disclosure” and “before the patch” appear regularly.

For sentence embeddings, we used a pre-trained language model, specifically all-MiniLM-L6-v2 from the Sentence Transformers library [16]. This model transforms text into a 384-dimensional vector that captures semantic meaning instead of just exact words. Two sentences with the same meaning but different words will stay close in this vector space. This is useful for categories like Responsibility Evasion, where the same concern can be said as “works on my machine” or “that is a user error”. TF-IDF treats these as completely different, but sentence embeddings put them close together.

Evaluating both representations makes it possible to see which works better overall and whether some categories are better served by one approach than the other.

4.5 Linear Classifiers for Text Classification

We evaluate two linear classifiers: logistic regression and a linear support vector machine (LinearSVC). Both are linear models, meaning they learn a weight for each feature and make predictions based on a weighted sum of the input. Linear models are a natural fit for text classification because text data tends to be high-dimensional and sparse, and linear models handle this well without needing large amounts of training data [26]. Since we only have around 1,000 labeled concern examples, deep neural networks would easily overfit, which aligns with prior findings on small datasets [27].

Logistic regression works by estimating the probability that a given comment belongs to each class, and then predicting the class with the highest probability. It has a regularization parameter C that controls how closely the model fits the training data. A smaller C makes the model more conservative, which helps it perform better on new data it has not seen before. Logistic regression is also a well-established baseline for text classification tasks and has been shown to perform competitively even against more complex models when training data is limited [26].

LinearSVC finds the decision boundary that maximizes the margin between classes. It does not produce probabilities natively, so a CalibratedClassifierCV wrapper is applied to enable probability output through Platt scaling [28]. LinearSVC is often more stable than logistic regression when the feature space is very sparse, which is the case with TF-IDF representations.

Both classifiers use balanced class weights. This means that examples from less

frequent categories contribute more to the loss during training. This prevents the model from simply ignoring rare categories in favor of the more common ones. In total, we test four configurations: TF-IDF combined with each classifier, and sentence embeddings combined with each classifier. We selected the best-performing configuration at each stage based on macro-averaged F1 across the outer cross-validation folds.

4.6 Class Imbalance and Evaluation Metrics

The dataset used in this study is highly imbalanced. Most GitHub comments do not contain any ethical concern, so the None class is much larger than any concern category. Also, among the concern categories, some appear frequently while others only appear a few times. Class imbalance is a well-known challenge in machine learning because standard evaluation metrics can be misleading when class sizes differ substantially [29].

Accuracy measures the proportion of all predictions that are correct. On an imbalanced dataset, a classifier that always predicts the most common class can still achieve high accuracy while being completely wrong about every minority class. This makes accuracy a misleading primary metric here. Instead, macro-averaged F1 is used [30].

F1 is the harmonic mean of precision and recall. Precision is the fraction of instances predicted as a given class that actually belong to it. Recall is the fraction of true instances of a class that the classifier correctly identifies. A classifier that is cautious and rarely predicts a class will have high precision but low recall; one that predicts a class too often will have high recall but low precision. F1 balances the two.

Macro-averaged F1 computes F1 separately for each class and then takes the simple average, giving every class equal weight regardless of how often it appears. This means that performing poorly on a rare category hurts the score just as much as performing poorly on a common one, which reflects the goal of detecting all concern types reliably. Accuracy is still reported alongside macro-F1 for completeness, and per-class precision, recall, and F1 are also reported so that performance on rare categories can be examined individually.

4.7 Nested Cross-Validation and External Validation

Evaluating a machine learning model on the same data used to tune it produces overoptimistic results, because the model has been implicitly fitted to that data [31]. Nested cross-validation avoids this by keeping model selection and performance estimation separate.

The outer loop uses Repeated Stratified 5-Fold cross-validation with three repeti-

tions, giving 15 test folds in total[32]. Stratification means that each fold preserves the class distribution of the full dataset, so no fold ends up with an unusually high or low proportion of any category. Repeating the process three times with different random splits reduces the influence of any single split on the final result. Within each outer training set, a separate inner loop searches over hyperparameter values using a 3-fold grid search, selecting the configuration that maximizes macro-F1 on the inner validation folds. The winning configuration is then retrained on the full outer training set and tested on the held-out outer fold. Because the outer test folds are never touched during the inner search, the reported performance scores reflect how the model would perform on genuinely unseen data. Out-of-fold predictions are aggregated across all 15 folds to produce a single confusion matrix, which gives a more stable picture of misclassification patterns than any individual fold would provide.

Stage 2 is evaluated at two levels of granularity: the 18 fine-grained concern types and the 6 broader themes. Comparing the two reveals whether classification performance degrades at finer granularity and whether confusions are concentrated within themes, such as between Gatekeeping and Participation Shaming, or spread across them. In this study, the 6-theme setting produced better macro-F1 scores, and the theme-level classifier was therefore used for the large-scale labeling in RQ3 and RQ4.

Cross-validation tests how well the model generalizes to new comments from the same repositories it was trained on. External validation tests something harder: whether the model works on a completely different repository it has never seen. In this study, the repository ansible/ansible is held out for this purpose. It was not used in any part of the RQ1 or RQ2 development process. The external validation is applied on this repository, and the results are compared against both a majority-class baseline and the internal cross-validation scores. For Stage 2 in the external set, a restricted macro-F1 is also computed that excludes themes with fewer than two true instances, since per-class F1 scores are unreliable when support is near zero. The difference between internal and external performance is examined to assess cross-repository generalizability. A performance drop is expected to some extent, since the external repository may have different linguistic conventions, governance structures, and concern distributions from the training repositories.

Because the classifier is structured as a two-stage pipeline, Stage 1 (binary detection) and Stage 2 (category or theme assignment) are evaluated separately. This makes it possible to tell whether errors come from failing to detect that a concern is present or from assigning it to the wrong theme or category.

5

Methods

This study adopts a mixed-methods empirical research design that combines inductive qualitative analysis with supervised machine learning and exploratory quantitative analysis. The methods are presented in the order of the research questions, as each phase builds on the results of the previous one.

5.1 Research Design Overview

The overall research design is summarised in Figure 5.1. The study proceeds in three sequential phases. In the first phase (RQ1), thematic analysis of manually collected GitHub comments is used to construct a taxonomy of ethical concerns and develop a coding framework. This taxonomy-building step follows an abductive logic: sensitizing dimensions derived from the ACM Code of Ethics and prior work were used to guide attention during open coding, but all categories were established empirically from the data itself, and only after a category became stable was it mapped to the most relevant ethical principle. In the second phase (RQ2), a larger labeled dataset is constructed using keyword-assisted data collection via the GitHub API, and supervised machine learning classifiers are trained and evaluated. In the third phase (RQ3 and RQ4), the best-performing classifier from RQ2 is applied to additional GitHub data to identify ethical concerns on a larger scale, which then enables temporal and repository-level analysis. This sequential design ensures that the coding framework used in RQ2 is grounded in the empirical observations from RQ1, and that the classifier developed in RQ2 provides the basis for the large-scale analysis conducted in RQ3 and RQ4.

5.2 RQ1: Qualitative Analysis and Taxonomy Construction

5.2.1 Repository Selection

Three main repositories were selected for the qualitative analysis phase: `nodejs/node`, `hashicorp/terraform`, and `stedolan/jq` (The repository was originally hosted at stedolan/jq and subsequently transferred to jqlang/jq; data collection used the current location at jqlang/jq.). These repositories represent the variation in the gover-

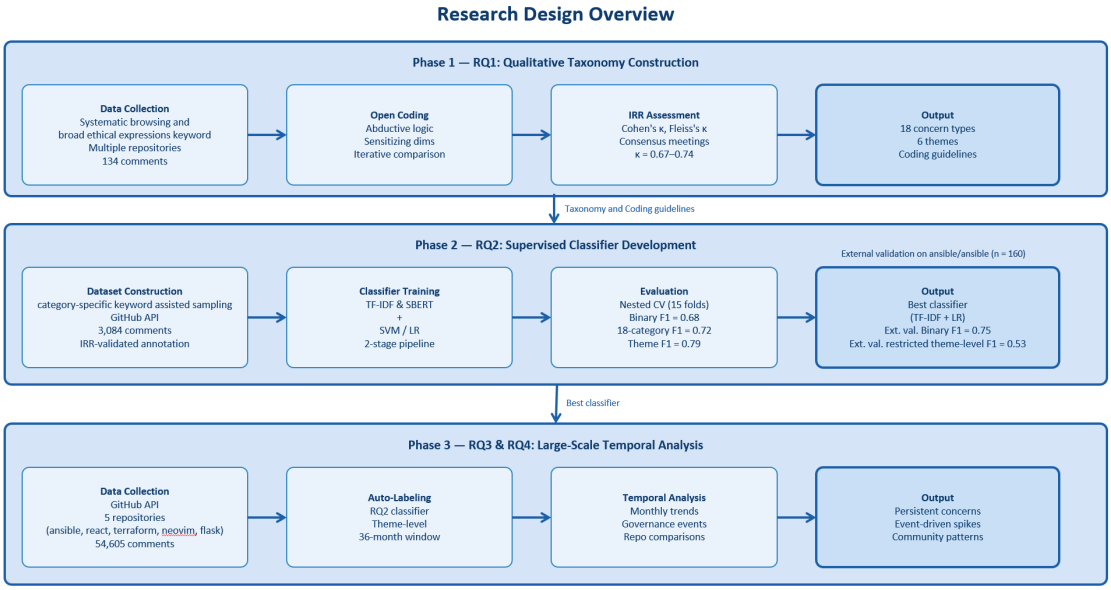

Figure 5.1: Overview of the research design and data flow across the three phases.

nance structure and the size of the community. Node.js is a large project with formal governance structures and documented Codes of Conduct. Terraform is a mid-sized project with active maintenance participation. jq is a smaller project with a less formalized governance structure. This variation enables the study to identify ethical concerns in different types of open-source development contexts rather than being confined to a single community type.

5.2.2 Unit of Analysis

The unit of analysis is an individual comment within a GitHub issue or pull request thread. These comments take various forms, such as a contributor complaining that a pull request was closed without explanation, a maintainer dismissing a question in a way that discourages further participation, or a community member raising concerns about how a decision was made without community input. This differs from an approach that treats the entire thread as one unit. Three reasons motivate this choice. First, a single issue or pull request thread can span many separate interactions involving different participants and different ethical dimensions; a single label for the whole thread would be inaccurate and would obscure the distribution of concern types. Second, comment-level units produce more consistent and comparable coding objects, which is important for inter-rater reliability. Third, for the machine learning phase in RQ2, comment-level text is more uniform in structure and length than full threads, which simplifies preprocessing and reduces variability in training data.

5.2.3 Data Collection

Data collection for RQ1 used two complementary strategies informed by the methodological guidance of Cruzes and Dybå [18] on purposive sampling in qualitative

software engineering research.

For the primary repositories (`nodejs/node`, `hashicorp/terraform`, and `jqlang/jq`), comments were collected manually by browsing GitHub issues and pull request threads sequentially without any keyword pre-filtering, following the immersive reading approach recommended by Cruzes and Dybå [18]. Each comment was read in full and considered for inclusion if it met the following criteria: the comment expressed a concern, objection, or dissatisfaction directed at a person, a decision, or a behavior rather than a purely technical issue; the concern was explicitly stated or clearly framed in the text rather than merely implied; and an identifiable ethical subject was present, meaning the concern was directed toward a person, a group, or a responsibility such as a maintainer, a contributor, the community, or project governance. Comments consisting only of technical discussions, bot-generated messages, very short acknowledgments such as "+1" or "thanks", and comments consisting only of links were excluded. In borderline cases where it was unclear whether a comment met these criteria, the comment was assigned the label None. This conservative labeling approach follows the principle that false positives, that is, incorrectly labeling a comment as containing an ethical concern, pose a greater threat to the validity of the taxonomy than false negatives, since including ambiguous cases could introduce noise into the coding framework and undermine the reliability of the inter-rater agreement assessment. Approximately 50 artifacts were collected through systematic browsing.

Because systematic browsing of large repositories is time-consuming, keyword-assisted search was used as a supplementary strategy for smaller repositories. The keywords were selected through an initial exploratory inspection of GitHub discussions, during which recurring expressions used by developers to signal dissatisfaction or ethical tension were identified. The final keyword included broad ethical expressions such as *unethical*, *not ethical*, *unfair*, *disrespectful*, and *complaint*. These terms flag a situation as problematic without specifying what type of concern is involved; for example, a comment stating "this is not ethical" reflects a general ethical judgment, whereas a comment such as "this feature violates user privacy" describes a specific concern type. Using broad expressions rather than category-specific terms was a deliberate choice: category-specific keywords would risk directing sampling toward pre-defined concern types and could influence the categories that emerge from inductive analysis. Broad keywords locate discussions where at least one participant has recognized a situation as problematic, while leaving open what the nature of that concern actually is.

During collection, comments were filtered using the same inclusion and exclusion criteria described above. Each retained comment was recorded as a separate entry in a structured Excel dataset with the following fields: artifact ID, repository name, repository type, artifact type, URL, comment text, creation date, primary open codes, taxonomy label, ethics-related flag, and notes.

For the keyword-sampled portion of the dataset, it is worth noting that the coding unit is the individual comment rather than the issue thread. A discussion flagged

through a keyword in the issue title or body may contain comments that demonstrate ethical tensions without using any ethical terminology. For example, a maintainer's reply to a complaint may constitute gatekeeping or participation shaming without containing any ethical keywords, and such comments remain identifiable through manual analysis of the comment content.

Thematic saturation was reached after approximately 119 of the 134 artifacts, with the remaining 15 introducing no new categories or themes, suggesting that the sample was sufficient to capture the range of ethical concern types present in the data. The 134 artifacts were selected from a much larger number of comments that were read and considered during data collection: many additional comments, such as those saying "thanks for the fix" or asking purely technical questions, were read but not included as they did not contain observable ethical concerns. Since the goal of RQ1 is to inductively derive a taxonomy from comments that express ethical concerns, including large numbers of non-concern comments would not contribute to the analysis. Only a small number of non-concern comments were retained to support the inter-rater reliability assessment.

The diversity of the data sources reduces the risk that the taxonomy reflects the particular patterns of any single community: systematic browsing covered three repositories of different scales and governance structures, while keyword-assisted search drew from a broader range of repositories across GitHub. The combination of these two strategies also reduces the risk of missing concern types that might not surface through either approach alone: systematic browsing captures naturally occurring concerns that do not contain explicit ethical vocabulary, while keyword-assisted search ensures that less frequent concern types are represented in the sample.

5.2.4 Coding Procedure and Taxonomy Development

Following the data collection and organisation process described in Section 5.2.3, each collected artifact was manually coded by the researcher using an iterative thematic analysis procedure following the five-step process described by Cruzes and Dybå [18]. This produced a two-layer taxonomy structure: the first layer consists of 18 concrete ethical concern types grounded in observable communication patterns; the second layer maps these types to broader ethical themes derived from the theoretical framework.

Prior to coding, the ACM Code of Ethics [5] and the six OSS ethical principles identified by Win et al. [3] were used to derive a set of nine sensitizing dimensions. The ACM framework provided general ethical principles applicable across computing contexts, while Win et al.'s principles offered a more specific grounding in open-source software development, covering accountability, attribution, autonomy, informed consent, privacy, and trust. Together, these two sources directed attention toward potential areas of ethical tension during open coding without predetermining what specific concern types would emerge or how many categories the final taxonomy would contain. For each sensitizing dimension, a set of anticipated behaviors was identified to further guide attention during coding. Some anticipated

behaviors were derived directly from the relevant ACM code of ethics principles, such as releasing untested or vulnerable software, which was derived from Principle 1.2 Avoid harm. Others were further informed by empirical findings from prior studies on toxic behavior and governance in open-source communities, including work on exclusionary communication and contributor marginalization [12], the limitations of codes of conduct in enforcing fair treatment [10], and moderation challenges in distributed volunteer communities [9]. In practice, the anticipated behaviors functioned as sensitizing cues rather than predetermined categories. Before reading each artifact, the coder was aware that certain types of behavior, such as insulting language or unjustified exclusion, might appear. This awareness helped direct attention during open reading, making it less likely that relevant communicative acts would be overlooked. However, the anticipated behaviors did not constrain what could be coded: if an artifact contained an ethical tension that did not correspond to any anticipated behavior, it was still coded. Conversely, if an anticipated behavior did not appear in the data with sufficient frequency to form a stable category, it was not included in the final taxonomy. A complete mapping of anticipated behaviors to their sources is provided in Table A.2 in the Appendix. The full list of sensitizing dimensions, their corresponding ACM code of ethics principles, anticipated behaviors, and their relationship to the final taxonomy is provided in Appendix A (Table A.1).

Each artifact was first read openly to identify any expressed ethical tension, without reference to the sensitizing dimensions. An initial code was assigned based on the language and content of the comment itself. For example, a comment such as "you shouldn't be making decisions without consulting the community" was first coded as "excluding community from decisions". Initial codes of this kind were then compared with other codes describing similar communicative acts, such as "decision made without community input" and "contributors not consulted on project direction", and aggregated into a candidate category. The boundaries of this candidate category were refined through repeated comparison until a stable definition emerged, in this case Governance Fairness, covering concerns about the legitimacy and fairness of decision-making processes. Only at this stage did the coder consider which sensitizing dimension the confirmed category most closely resembled, in this case Fairness. This process ensured that the specific concern types forming the first layer of the taxonomy were established from the data before being mapped to the second layer of broader themes.

To further illustrate how open codes were aggregated into categories, two additional examples are provided below.

The first example concerns the category *Personal Attacks and Demeaning Communication*. Initial open coding produced a range of descriptive labels across different comments, including `direct_insult`, `demeaning_language`, `competence_demeaning`, `personal_attack`, `personal_attack_language`, `personal_attack_on_competence`, `professional_legitimacy_attack`, and `mockery`. These codes were assigned to comments with different surface expressions: "You will perish in flames." was coded as `direct_threat`, "You're just a clear example of someone who spews nonsense"

was coded as `personal_attack_language`, and "You contributed to a spec that doesn't do in 2018 what superagent did in 2012" was coded as `competence_demeaning`. Despite these differences, repeated comparison revealed that all codes shared a common underlying pattern: the comment directed negative evaluation at a person rather than engaging with the technical substance of the discussion. This shared characteristic became the defining criterion of the aggregated category.

Negative case analysis was conducted to test the boundaries of this category. For instance, a comment such as "Is it so hard to have fun these days?" was initially considered for this category, but closer examination revealed that while the comment was dismissive and used mocking language, it did not constitute a direct attack on any individual. It was therefore retained as None, on the grounds that the problematic behavior was not sufficiently directed at a person to meet the category definition.

The second example illustrates a different aggregation pattern, one in which the initial codes described varied surface situations rather than a single recurring behavior. Open coding of comments related to contribution recognition produced labels including `attribution`, `lack_of_attribution`, `code_attribution_violation`, `aggregating_datasets_`, `misattribution`, and `authorship_ethics`. The specific situations differed considerably: one comment concerned a manufacturer modifying an upstream library without crediting the original authors; another described a project owner replacing a contributor's donation username with their own; a third raised the need for dataset citation fields where licenses require mandatory attribution; and a fourth identified a workflow flaw in which the translation approval process allowed code substitution that placed the wrong authorship on the final work. Despite these surface differences, repeated comparison revealed that all codes shared a common concern: a contributor's work was not being properly recognized or credited. This shared concern became the defining criterion of the aggregated category *Attribution and Credit Issues*.

This aggregation process followed the procedure recommended by Cruzes and Dybå [18]. Throughout the aggregation, negative case analysis was conducted to test the boundaries of emerging categories. For each proposed grouping, the coder actively searched for codes that challenged the grouping, and category definitions were refined accordingly until no further negative cases could be identified. For example, during the aggregation of *Gatekeeping* and *Participation Shaming*, several codes appeared to fit both categories. Closer examination revealed a consistent distinction: *Gatekeeping* involved challenging a person's right or standing to participate, whereas *Participation Shaming* involved criticizing or discouraging a legitimate way of participating, such as asking questions or leaving comments. For instance, a comment stating "I don't see why this discussion needs to happen on GitHub... it should be locked to collaborators who are going to be working on the code" was coded as *Gatekeeping* because it challenged certain community members' standing to participate in the discussion at all, rather than criticizing the way they were participating. This boundary was identified through repeated comparison of borderline cases and was subsequently incorporated into the coding framework as a decision rule.

Once all initial codes were aggregated into candidate categories forming the first layer, these categories were grouped into broader themes constituting the second layer. The balance between the two layers was achieved by keeping the sensitizing dimensions and the coding process strictly separate in time: the theoretical framework was consulted only after an initial code had already been assigned from the data, ensuring that the first-layer categories and their boundaries were established inductively before being mapped to the second-layer themes. The theme names represent a post-hoc refinement of the original sensitizing dimension labels, reflecting the specific patterns that were actually observed in the data. For example, the dimension “Fairness” evolved into the theme "Fairness in Participation and Decision-Making" after the data revealed that fairness concerns in this context were specifically concentrated around participation and governance processes. Three sensitizing dimensions, namely Self-Interest, Welfare, and Sustainability, did not receive sufficient empirical support and were therefore not included in the final taxonomy.

5.2.5 Inter-Rater Reliability

To assess the reliability of the coding framework, two additional annotators were recruited to independently code a subsample of the data. Both annotators were master’s students in software engineering with prior experience as research assistants, and neither had been involved in the development of the taxonomy, which helped reduce potential bias in the coding process. A structured annotation protocol was established prior to independent coding. Both additional annotators were provided with the same coding guidelines, which specified the coding unit definition, category definitions with illustrative examples, decision rules for borderline cases where multiple categories might apply, and a step-by-step coding process. Each annotator independently coded a stratified subsample of 58 artifacts drawn from the full dataset of 134 artifacts using Excel spreadsheets, which were distributed to each annotator separately. Each annotator completed the coding independently within one week, without consultation with the other annotators. This was because the purpose of the inter-rater reliability assessment was to evaluate the clarity, consistency, and reproducibility of the coding framework, rather than to duplicate the full coding process for all artifacts. Using a substantial but smaller subsample made the additional annotation effort feasible while still allowing agreement to be assessed across the taxonomy as a whole. This approach is consistent with methodological guidance on intercoder reliability in qualitative research, which notes that reliability is often assessed on a subset of the data rather than the full dataset, particularly under resource constraints, and that such a subset may be selected using random or other justifiable criteria, including stratified sampling [20].

Stratification was based on coding category and was designed to ensure coverage of all 18 ethical concern categories as well as the `None` label. As a general allocation rule, the subsample included 10 `None` cases, 5 cases from the largest category, 2–3 cases from medium-frequency categories, and 2 cases from low-frequency categories. These target quotas were determined pragmatically rather than through a formal sampling formula. The minimum quota of two cases was used because several categories had only two available artifacts in the full dataset, and retaining both cases

allowed those rare categories to be represented as fully as possible. A slightly larger quota of two to three cases was used for medium-frequency categories, while the largest category was capped at five cases so that it would be represented without dominating the reliability subset. Because the full dataset was highly imbalanced across categories, simple random sampling was not used. The aim of this allocation was not to reproduce the prevalence distribution of the full dataset, but to ensure adequate category coverage for assessing agreement across the taxonomy as a whole. Table 5.1 reports the category composition of the pre-consensus reliability subset.

Cohen's Kappa [21] was calculated for each pairwise comparison, and Fleiss' Kappa [22] was calculated across all three annotators. Following the reliability assessment, the three annotators held an online consensus meeting to review disagreements line by line, clarify category boundaries, and reach agreement on borderline cases. The resulting agreement statistics are presented in Chapter 6.

Table 5.1: Category composition of the 58-artifact reliability subset (pre-consensus)

Category	**Count**
None	10
Personal Attacks and Demeaning Communication	5
Attribution and Credit Issues	3
Discrimination	3
Governance Fairness	3
Intellectual Property Violation	3
Lack of Responsiveness	3
Lack of Transparency	3
Power Abuse or Authority Dominance	3
Privacy Violation	3
User Autonomy and Control Violation	3
Gatekeeping	2
Impersonation	2
Irresponsible Disclosure	2
Irresponsible Release	2
Misrepresentation	2
Open Source Obligation Violation	2
Participation Shaming	2
Responsibility Evasion	2
Total	58

5.3 RQ2: Automated Classification

5.3.1 Dataset Construction

Building on the taxonomy from RQ1, a larger labeled dataset was constructed for the machine learning phase. In this study, an artifact refers to a unit of data collected from GitHub, specifically an individual comment within an issue or pull request discussion. Artifacts are collected using GitHub API search with keywords derived

from each of the 18 concern categories in the taxonomy. For each category, approximately 1,500 to 2,000 candidate artifacts are retrieved through keyword search. Keywords were constructed based on how these concerns are actually expressed in GitHub discussions, rather than through abstract synonym expansion using lexical tools. For each category, keywords capture both functionally equivalent expressions and distinct contextual patterns. For example, for Lack of Responsiveness, keywords include expressions such as "no response", "never replied", and "radio silence", which reflect different ways developers describe being ignored in practice. For Lack of Transparency, keywords include expressions such as "no explanation", "silently changed", and "kept in the dark", capturing different communication patterns associated with the same concern. The full list of keywords used for each category is provided in Appendix A.

Following keyword-based retrieval, candidate artifacts were reviewed using the coding framework developed in RQ1, which specifies inclusion criteria, category definitions, and decision rules for borderline cases. Comments that did not express an observable ethical concern directed at a person, a decision, or a behavior were excluded without detailed reading, including purely technical discussions such as bug reports and feature requests, as well as duplicate comments and non-English comments, as these are typically identifiable at a glance. For example, a comment saying *"There is no response from the server when the timeout exceeds 30 seconds."* would be excluded, while a comment saying *"this issue has been ignored for months despite multiple requests"* would be retained as a potential Lack of Responsiveness concern.

For borderline cases where it was unclear whether a comment met the inclusion criteria, the decision rules documented in the coding framework were consulted to ensure consistent labeling across categories. The number of reviewed candidates varied by category based on data density. For several categories where sufficient positive examples were collected early, review was stopped before exhausting the full retrieval pool; these included *Participation Shaming* (108 positive examples), *Attribution and Credit Issues* (78), *Open Source Obligation Violation* (82), *Intellectual Property Violation* (78), and *Personal Attacks and Demeaning Communication* (70). For rarer categories such as *Irresponsible Disclosure* (13 positive examples) and *Misrepresentation* (26), all retrieved candidates were reviewed to maximize coverage. The final dataset comprises 3,084 labeled artifacts, of which 1,062 are confirmed ethical concern examples, and 2,022 are labeled None, including both clearly non-ethical comments and borderline cases that did not meet the inclusion criteria, thereby providing the machine learning models with realistic and challenging boundaries. The entire labeling process was conducted by the primary researcher over approximately one month.

The use of category-specific keywords for RQ2 data collection differs from the approach used in RQ1. This is intentional: in RQ1, the taxonomy had not yet been established, so category-specific keywords would have risked shaping the results. In RQ2, the taxonomy is fixed, and keywords serve only as a retrieval tool to locate candidate examples efficiently. Whether a retrieved artifact actually instantiates the

category is determined by manual review using the coding framework.

5.3.2 Inter-Rater Reliability for RQ2 Annotation

To address subjectivity in the manual labeling process, inter-rater reliability was evaluated in a staged manner, reflecting the hierarchical structure of the labeling task. The staging separates two distinct sources of disagreement. The first is whether a comment contains an ethical concern at all, for example, whether a dismissive reply from a maintainer should be coded as Participation Shaming or labeled as None. The second is which specific category should be assigned once a concern has been identified, for example, whether a comment should be coded as Gatekeeping rather than Participation Shaming. Assessing these two decisions separately makes the interpretation of agreement scores more precise and allows category-boundary problems to be identified more clearly. This logic is methodologically analogous to prior staged annotation work in software engineering. For example, Gachechiladze et al. [24] first assessed whether a comment expressed anger, and then assessed the direction of that anger, reporting agreement separately for each stage. Although the present study addresses ethical concerns rather than emotions, both designs share the same principle of separating detection of the target phenomenon from its fine-grained categorization.

Although the coding framework had already been validated through the IRR assessment conducted in RQ1, a separate IRR evaluation was conducted for RQ2 because the dataset was constructed using category-specific keywords rather than systematic browsing and broad ethical expressions keywords. This different retrieval strategy may surface comments with different linguistic characteristics from those encountered in RQ1, making it important to verify that the coding framework could be applied consistently to the new data. This approach is also consistent with broader guidance on IRR assessment, which recognizes that IRR may be assessed on a representative subset of the data when time and resource constraints apply [33].

In the first stage (IRR-A), a pilot sample of approximately 150 comments was constructed to evaluate whether annotators could reliably distinguish comments containing an observable ethical concern from comments that should be labeled as None. The sample included 50 concern-labeled comments drawn from across different themes, and 100 None-labeled comments, reflecting the approximate proportion of concern and non-concern cases in the full RQ2 dataset.

In the second stage (IRR-B), a separate sample was constructed from comments already identified as containing ethical concerns, with no overlap with the IRR-A sample. To ensure that all 18 fine-grained categories were assessable, quota-based sampling was used, with up to approximately 30 comments per category where available, and all available instances included for low-frequency categories. This was necessary because the category distribution is uneven, and purely random sampling would underrepresent rare categories and make category-level agreement difficult to interpret.

Across both stages, the same two annotators who participated in the RQ1 reliability assessment independently coded the samples using the same coding guidelines described in Section 5.2.5. The annotation was conducted using Excel spreadsheets, which were distributed to each annotator separately. Each annotator completed the coding independently within one week per stage, without consultation with the other annotators. Following each IRR assessment, the three annotators held an online consensus meeting to review disagreements line by line, identify recurring sources of ambiguity, and refine category boundary definitions. In this way, IRR served as a post-annotation reliability check on the consistency and interpretability of the coding framework, rather than as a pre-annotation gatekeeping step. Cohen's Kappa [21] was calculated for each pairwise comparison, and Fleiss' Kappa [22] was calculated across all three annotators. The resulting agreement statistics are presented in Section 6.2.1.

5.3.3 Preprocessing and Classification

5.3.3.1 Text Preprocessing

All labeled artifacts are preprocessed using a consistent pipeline before feature extraction. URLs are removed, as are GitHub-specific tokens including @mentions and issue reference numbers (e.g., #1234). Fenced code blocks delimited by triple backticks and inline code spans are stripped to prevent programming syntax from dominating the feature space. The remaining text is lowercased and non-alphabetic characters are removed. Tokens are then lemmatized using the WordNet Lemmatizer from the Natural Language Toolkit (NLTK).

Stop words are removed using the standard English stop word list, with one deliberate modification: negation terms (no, not, nor, never, neither, nobody, nothing, nowhere) are retained. These words carry semantically important information in ethical language, as they distinguish, for example, "this decision is fair" from "this decision is not fair", and their removal would discard a signal that is directly relevant to several concern categories. The same preprocessing pipeline is applied uniformly to both stages of the classifier.

5.3.3.2 Two-Stage Classification Architecture

The classification system is structured as a two-stage pipeline that mirrors the hierarchical structure of the annotation task. In Stage 1, a binary classifier distinguishes comments that contain no ethical concern (the None class) from comments that express ethical concern. In Stage 2, a multi-class classifier assigns each concern-positive comment to one of the ethical concern categories identified in the taxonomy. This decomposition is motivated by two considerations. First, the None class constitutes the large majority of GitHub comments in any repository, and treating the full problem as a single multi-class task would force the classifier to distinguish rare concern categories while simultaneously identifying a dominant non-concern majority, a task for which standard classifiers tend to perform poorly. Second, modular decomposition allows each stage to be evaluated and improved independently, mak-

ing it possible to attribute performance differences to detection failures in Stage 1 versus categorization failures in Stage 2.

Stage 2 is evaluated under two granularity settings, which are run separately and their results compared. In the 18-category setting, the classifier assigns each comment to one of the 18 fine-grained concern types in the taxonomy. In the 6-theme setting, the 18 categories are first mapped to their parent themes using the taxonomy structure, and the classifier operates on these six broader groupings: Respect and Inclusion; Fairness in Participation and Decision-Making; Responsibility and Accountability; Privacy and User Autonomy; Honesty; and Ownership, Attribution and Obligation. Comparing the two settings reveals whether classification performance degrades at finer granularity, and whether this degradation is concentrated in categories that are difficult to distinguish from each other, such as Gatekeeping and Participation Shaming, which belong to the same theme and therefore cannot be confused at the theme level.

5.3.3.3 Feature Representations

Two types of feature representations are evaluated for each stage. TF-IDF vectors and sentence embeddings were selected because they represent two fundamentally different approaches to text representation, allowing the study to assess whether lexical patterns or semantic similarity better captures the language of ethical concerns in this dataset. Simpler representations such as raw bag-of-words were not used because TF-IDF extends bag-of-words by down-weighting terms that appear frequently across all documents, which is important here since common words like "issue" or "change" would otherwise dominate the feature space without contributing to distinguishing concern types. Word embeddings such as Word2Vec were not used because they operate at the word level: to represent a full comment, word vectors must be aggregated, for example by averaging, which loses information about word order and context. Sentence embeddings avoid this problem by encoding the entire comment directly into a single vector.

The first is TF-IDF vectors, using unigram and bigram features, a maximum vocabulary of either 20,000 or 50,000 terms selected during hyperparameter search, and a minimum document frequency of two. The second is sentence embeddings produced by the all-MiniLM-L6-v2 model from the Sentence Transformers library.

TF-IDF is a natural choice for this dataset because GitHub developer communication contains a relatively stable domain-specific vocabulary: developers discussing ethical concerns tend to use recurring expressions that are characteristic of specific concern types, such as "responsible disclosure" or "before the patch" for Irresponsible Disclosure, and "no credit" or "removed my name" for Attribution and Credit Issues. Sentence embeddings represent meaning at the semantic level rather than the lexical level, making them potentially more robust to surface variation; for example, Responsibility Evasion can be expressed as "works on my machine", "that is a user error", or "feel free to fix it yourself", without sharing any common vocabulary. Evaluating both representations allows the study to determine which approach better

captures the language of ethical concerns in this dataset.

5.3.3.4 Classifiers

Logistic Regression and LinearSVC were selected as the two classifiers for evaluation. Both are well suited to high-dimensional sparse feature spaces such as TF-IDF representations, and both have been shown to perform competitively on text classification tasks in software engineering with limited training data. Logistic Regression estimates class probabilities through a sigmoid function, while LinearSVC maximizes the margin between classes; evaluating both provides a broader picture of how different linear approaches perform on this task.

Linear classifiers are used rather than deep learning approaches because the dataset size of 1,000 labeled concern examples is too small to reliably train deep neural networks without overfitting [27]. Naive Bayes, another common baseline for text classification, was also not included because it assumes that all features are independent of each other. In practice, words in GitHub comments tend to co-occur in meaningful patterns; for example, "closed" and "without explanation" frequently appear together in Gatekeeping concerns. Neither Logistic Regression nor LinearSVC makes this independence assumption, making them more appropriate for capturing the co-occurrence patterns present in this dataset.

Two linear classifiers are paired with each feature representation, yielding four model configurations per stage (Table 5.2): TF-IDF + LR, TF-IDF + SVM, SentEmb + LR, and SentEmb + SVM. Sentence embeddings were included alongside TF-IDF as a comparison condition to test whether capturing semantic similarity beyond bag-of-words would improve performance on categories whose surface expressions vary widely. Both classifiers use balanced class weights to compensate for the uneven class distribution. The SVM classifier is wrapped in a CalibratedClassifierCV so that it produces probability estimates, which are required for downstream threshold analysis in RQ3 and RQ4.

Table 5.2: Model configurations evaluated at each stage

Configuration	Features	Classifier	Abbreviation
1	TF-IDF bigrams	Logistic Regression	TF-IDF + LR
2	TF-IDF bigrams	Calibrated LinearSVC	TF-IDF + SVM
3	Sentence embeddings	Logistic Regression	SentEmb + LR
4	Sentence embeddings	Calibrated LinearSVC	SentEmb + SVM

5.3.3.5 Internal Validation Strategy

Model performance is estimated using nested cross-validation to ensure that hyperparameter selection does not inflate the reported scores. The outer loop applies Repeated Stratified 5-Fold cross-validation with three repetitions, producing 15 test folds in total. Stratification is used to preserve class proportions across folds, and the repetitions reduce sensitivity to any particular split of the data. Within each

outer training set, a Stratified 3-Fold grid search over the hyperparameter values listed in Table 5.3 selects the best hyperparameters by macro-F1; the winning configuration is then retrained on the full outer training set before being evaluated on the held-out fold.

Macro-averaged F1 is the primary metric, as accuracy would be misleading given the class imbalance: a classifier that ignores rare concern types entirely could still achieve high accuracy by predicting the majority class. All concern categories and themes are treated as equally important, which is why macro-F1 rather than weighted-F1 is used. The same procedure and the same four model configurations are applied to both the 18-category and the 6-theme settings of Stage 2, making the results directly comparable across granularity levels. Per-class precision, recall, and accuracy are also reported, and all scores are summarized as mean and standard deviation across the 15 folds with 95% confidence intervals.

The out-of-fold predictions are aggregated into a single confusion matrix across all folds and repetitions, giving a more stable picture of misclassification patterns than any individual fold would provide. The best configuration at each stage is chosen by mean macro-F1 across the 15 outer folds. The selected model is then retrained on the full development set for use in external validation and large-scale labeling in RQ3 and RQ4.

Table 5.3: Hyperparameter grids searched in the inner loop

Configuration	Hyperparameter	Values searched
TF-IDF + LR	n-gram range	(1,1); (1,2)
	max features	20,000; 50,000
	C (regularization)	0.01; 0.1; 1; 10
TF-IDF + SVM	n-gram range	(1,1); (1,2)
	max features	20,000; 50,000
	C (regularization)	0.1; 1; 10
SentEmb + LR	C (regularization)	0.01; 0.1; 1; 10
SentEmb + SVM	C (regularization)	0.1; 1; 10

5.3.3.6 External Validation

Following internal validation, the fixed two-stage pipeline is evaluated on the held-out repository *ansible/ansible*, which was not included in either the RQ1 qualitative analysis or the RQ2 development set. The external validation set was constructed using the same keyword-based retrieval and manual labeling procedure applied to the development set, with all labeling decisions made by the same primary researcher using the same coding framework. This ensures that the external labels are consistent with the development set labels and that the IRR results reported in Section 6.2.1 apply equally to the external annotation. Whereas cross-validation estimates how well the model generalizes to new comments drawn from the same repositories as the training data, external validation tests whether it generalizes to a completely

unseen repository with potentially different linguistic conventions, governance structures, and concern distributions.

On the external set, the final Stage 1 and Stage 2 models are each compared against two reference points: a majority-class baseline that predicts the most frequent class for every instance, and the internal cross-validation mean, which indicates the expected performance level on in-distribution data. Both the 18-category and the 6-theme settings of Stage 2 are evaluated on the external set, allowing the generalizability comparison to be made at both granularity levels. For Stage 2, a restricted macro-F1 is additionally computed that excludes classes with fewer than two true instances in the external set, in order to avoid reporting unreliable per-class F1 values for near-empty classes; both the full and restricted macro-F1 are reported. Any systematic performance drop between internal cross-validation and external evaluation is discussed as a potential indicator of repository-specific language patterns or limited cross-repository generalizability.

5.3.3.7 Supplementary Analysis

Three supplementary analyses are conducted alongside the main performance metrics. First, confusion matrices are produced for both stages, for both Stage 2 settings, and for both internal and external evaluation. Each matrix is saved in two forms: row-normalized to show the proportion of each true class that was correctly or incorrectly classified, and raw counts to show the absolute number of predictions. Second, per-class precision, recall, and F1 are reported for both Stage 2 settings, sorted by the number of true instances in ascending order, so that the performance on rare concern types is not hidden by averaging. Third, an error analysis is conducted by manually examining a sample of misclassified comments from both Stage 2 settings to identify recurring failure patterns, such as comments that sit on the boundary between two related classes or that use language typical of one class while expressing a concern from another.

5.4 RQ3 and RQ4: Temporal Patterns and Project-Level Associations

RQ3 and RQ4 apply the classifier developed in RQ2 to a larger collection of GitHub comments to conduct an exploratory analysis of how ethical concerns relate to project lifecycle and repository characteristics. This phase is designed to be exploratory rather than confirmatory: the goal is to identify patterns and associations that may inform future research, rather than to test pre-specified hypotheses.

5.4.1 Repository Selection

Five repositories were selected for this phase using purposive sampling: ansible/ansible, facebook/react, hashicorp/terraform, neovim/neovim, and pallets/flask. The selection was guided by the goal of maximizing variation across three dimensions:

governance structure, technical domain, and community scale. Regarding governance, facebook/react and hashicorp/terraform are backed by large corporations, while ansible/ansible, neovim/neovim, and pallets/flask are primarily community-governed. In terms of technical domain, the five repositories span infrastructure tooling (ansible, terraform), a text editor (neovim), a web framework (flask), and a UI library (react). Regarding community scale, ansible, react, and neovim represent larger projects with higher discussion volumes, while terraform and flask represent smaller communities with more moderate activity levels. Although flask's comment volume is substantially lower than the other four repositories, its inclusion provides a small-community contrast that is relevant to examining whether ethical concern patterns differ by community scale. This variation allows the analysis to examine whether temporal patterns and associations with repository characteristics differ across different types of open-source projects, rather than reflecting the idiosyncrasies of a single community. ansible/ansible was held out as the external validation repository in RQ2 and was therefore not used in classifier development; its inclusion here is appropriate because it is being used for analysis rather than training.

5.4.2 Time Window

The analysis covers a 36-month period from April 2023 to April 2026. This window is long enough to capture meaningful variation in project activity and ethical concern patterns over time while remaining computationally tractable. Covering three years also makes it possible to examine whether ethical concerns cluster around particular phases of project development or respond to specific events such as major releases or governance changes.

5.4.3 Data Collection

Issue comments and pull request comments were collected for each repository through the GitHub REST API. Two filtering steps were applied before classification. First, comments authored by known bot accounts were excluded based on account name pattern matching, since bot-generated content does not reflect community communication and would add noise to the ethical concern labels. Second, comments shorter than 30 characters were excluded, as these typically consist of acknowledgments such as "+1", "LGTM", or "thanks" that contain no substantive content. After filtering, the dataset comprises 54,605 comments in total, distributed across repositories as follows: ansible 19,840 comments, react 18,124 comments, neovim 18,035 comments, terraform 7,725 comments, and flask 881 comments. The substantially lower comment volume for flask reflects the smaller scale of that community over the study period.

Repository-level metadata was collected separately for each monthly time window to capture the activity context in which ethical concerns occur. These indicators reflect different aspects of project activity: issue counts and pull request counts measure the volume of community interaction and contribution attempts, while commit counts and active contributor counts reflect the pace of development and the size of the

actively contributing population. Together, they provide a broad picture of how busy and active a repository is in a given month, which is the basis for the association analysis in RQ4. Issue counts and pull request counts were obtained through the GitHub Search API. Commit counts and active contributor counts were obtained through the /stats/contributors endpoint, where an active contributor in a given month is defined as any account with at least one commit recorded in that month.

5.4.4 Classification

The two-stage classifier trained in RQ2 was applied to all 54,605 comments. Stage 1 distinguishes comments containing an ethical concern from those that do not. Stage 2 assigns each concern-positive comment to one of the six broader themes in the taxonomy. Because fine-grained 18-category classification produced lower macro-F1 scores than theme-level classification in both internal cross-validation and external validation, the theme-level classifier is used here to support more reliable large-scale labeling. Classification was performed at the comment level in batches of 512, which balances memory usage and processing speed without affecting the classification outputs.

5.4.5 Monthly Aggregation

Comment-level predictions were aggregated to the repository-month level to construct the main analysis table, which contains 180 rows, one for each of the five repositories across each of the 36 months. For each repository-month, the ethical comment ratio was computed as the number of comments classified as containing an ethical concern divided by the total number of comments in that month. This ratio rather than the raw count is used as the primary outcome measure, because it accounts for differences in overall discussion volume across months and repositories. The proportion of each theme among concern-positive comments was also computed for each repository-month. Repository-level metadata indicators were merged into this table, resulting in a dataset where each row combines the ethical concern measurements with the corresponding activity indicators for that month.

5.4.6 RQ3: Temporal Analysis

For RQ3, the ethical comment ratio for each repository was plotted as a time series over the 36-month study period. A three-month rolling average was applied to smooth short-term fluctuations and make longer-term trends more visible; a window of three months was chosen because it is short enough to preserve meaningful variation while reducing the influence of individual months with unusually high or low comment volumes. The composition of ethical concern themes was visualized as a stacked area chart to examine whether the relative prevalence of different concern types shifted over time. Both absolute counts and relative proportions are reported: absolute counts indicate the volume of ethical communication, while proportions normalize for differences in monthly activity levels.

5.4.7 RQ4: Association Analysis

For RQ4, the association between repository activity indicators and the ethical comment ratio was examined using Spearman rank correlation. Spearman correlation was chosen over Pearson correlation for two reasons. First, it does not assume a linear relationship between variables, which is appropriate here because the relationship between, for example, the number of active contributors and the ethical comment ratio is unlikely to be strictly linear. Second, it is less sensitive to outliers, which matters given that some months may have unusually high activity or concern levels that could disproportionately influence a Pearson coefficient. Four activity indicators were included: issue count, pull request count, commit count, and active contributor count.

Correlation analysis was chosen over regression-based approaches because the goal of RQ4 is exploratory: the analysis asks whether activity indicators and ethical concern ratios tend to move together, not whether one variable predicts another or whether a causal relationship exists. The reliability of the Spearman coefficients is further supported by the sample sizes: 36 monthly observations per repository and 180 in the pooled analysis are sufficient for Spearman correlation to produce stable estimates.

Correlation was examined at two levels. At the per-repository level, Spearman correlations were computed separately for each of the five repositories using its 36 monthly observations, asking whether busier months within a given project tend to coincide with more or fewer ethical concerns. This within-repository analysis controls for differences between projects, since each repository serves as its own baseline. At the pooled level, correlations were computed across all 180 repository-month observations combined, treating the dataset as a single collection regardless of which repository each observation comes from. This cross-repository analysis asks whether the pattern observed within individual repositories also holds when all five projects are considered together.

Because flask contributes substantially fewer comments than the other repositories, its monthly observations are less stable and may carry more measurement noise. A sensitivity analysis was therefore conducted by computing pooled correlations both with and without flask, and comparing the results to assess how much the flask observations influence the overall findings. The analysis is exploratory in nature and does not aim to establish causal relationships between activity indicators and ethical concern patterns.

All analyses were conducted in Python using pandas for data management, scipy.stats for correlation analysis, and matplotlib and seaborn for visualization.

5.5 Validity Considerations

This study addresses several validity concerns.

Construct validity is supported by developing the coding framework inductively from the data before mapping categories to the ACM Code of Ethics, ensuring that the taxonomy reflects what developers actually express rather than what the theoretical framework anticipates. Inter-rater reliability was assessed at multiple points, including separate evaluations for RQ1 taxonomy coding and RQ2 annotation, to confirm that the coding framework could be applied consistently across different datasets and retrieval strategies.

Internal validity is affected by the sampling strategy. The combination of systematic browsing and keyword-assisted search reduces but does not eliminate sampling bias. For RQ1, systematic browsing was used as the primary strategy to avoid directing sampling toward pre-defined concern types. For RQ2, category-specific keywords were used deliberately as a retrieval tool after the taxonomy was fixed, but this means the dataset is not a fully random sample of GitHub comments and may overrepresent comments containing characteristic ethical vocabulary. This keyword-based sampling bias has a direct implication for the classifier comparison in Section 6.2: comments retrieved through category-specific keywords are more likely to contain those keywords or closely related terms, which are precisely the features that TF-IDF weights most heavily. The relative advantage of TF-IDF over sentence embeddings reported in Section 7.2.3 should therefore be interpreted with caution, as it may partly reflect an artefact of the retrieval strategy rather than a generalizable property of the classification problem.

External validity is limited by the purposive selection of repositories and the language of the data. All repositories analyzed across all three phases are English-language projects hosted on GitHub, and the keywords used for data collection are exclusively in English. The taxonomy and trained classifiers may therefore not generalize well to open-source communities where discussions are conducted in other languages. Additionally, the findings are most applicable to projects hosted on GitHub, and may not extend to communities using other collaboration platforms such as GitLab or self-hosted systems.

A separate concern is ecological validity. The RQ2 dataset was constructed through keyword-assisted retrieval, meaning it does not represent a naturalistic sample of GitHub comments but rather a collection biased towards comments containing ethical concern vocabulary. The distribution of concern types in this dataset may therefore not reflect the true prevalence of ethical concerns in open-source development more broadly, and classifier performance on this dataset may not generalise to a randomly sampled corpus. The external validation on ansible/ansible provides some evidence of cross-repository generalizability at the theme level, but the validation set was limited in size and category coverage, and further validation across a broader range of repositories would be needed to support stronger generalizability claims.

A further dimension of validity concerns temporal scope. The 36-month window used in RQ3 and RQ4 was selected to balance data volume against feasibility, but it may be insufficient to detect long-term structural trends in the expression of ethical

concerns. Community norms, governance practices, and the prevalence of specific concern types may shift over multi-year timescales that a three-year window cannot capture. Stronger temporal inferences would require a longer observation period, which was beyond the scope of the present study.

Reliability is supported through documented coding procedures, structured annotation protocols, inter-rater agreement assessments at each phase, and consensus meetings to resolve disagreements and refine category boundary definitions.

6

Results

This chapter presents the results of the study across all four research questions. Section 6.1 reports the taxonomy of ethical concerns and the inter-rater reliability assessment for RQ1. Section 6.2 presents the automated classification results for RQ2, including inter-rater reliability for the labeled dataset, cross-validation performance, and external validation on ansible/ansible. Section 6.3 reports the temporal and repository-level analysis for RQ3 and RQ4, covering ethical concern patterns over the 36-month study period and the associations between repository activity indicators and ethical concern frequency.

6.1 Taxonomy of Ethical Concerns (RQ1)

The qualitative analysis produced a taxonomy of 18 ethical concern types organized under six broader themes. The taxonomy is grounded in observable communication patterns in GitHub issue and pull request comments and is supported by substantial inter-rater agreement.

6.1.1 Inter-Rater Reliability

Two additional annotators independently coded a sample of 58 artifacts. The results are summarized in Table 6.1.

Table 6.1: Inter-rater reliability results

Comparison	Kappa value
Annotator 1 vs Annotator 2	0.740
Annotator 1 vs Original coder	0.668
Annotator 2 vs Original coder	0.721
Fleiss' Kappa (all three annotators)	0.709

All pairwise Cohen's Kappa values fall in the range of 0.668 to 0.740, and Fleiss' Kappa across all three annotators is 0.709. According to the scale proposed by Landis and Koch [23], values in the range 0.61 to 0.80 indicate substantial agreement. These results indicate that the coding framework is sufficiently clear and consistent for reliable application by independent coders. After this reliability assessment, the

three annotators held a consensus meeting to resolve disagreements and refine our coding guidelines.

During the meeting, we first refined the Gatekeeping category. We decided that the key criterion is whether a comment challenges a person's right or standing to raise something, rather than critiquing the substance of what they said. For example, a comment such as "Quite honestly, I don't see why this discussion needs to happen on GitHub... it should be locked to collaborators who are going to be working on the code" was coded as Gatekeeping. This is because it questioned whether certain community members had the standing to participate in the discussion at all, rather than engaging with the content of what they said.

Secondly, We also agreed on a general principle: a category should only be applied when the problematic behavior is clearly present in the text; inferring ethical concerns from context without textual evidence is insufficient grounds for labeling. For instance, a comment asking Why are you scandalizing this?" was considered for Gatekeeping but ultimately retained as None, because the comment was criticizing the tone of participation rather than challenging the person's right to participate.

Additionally, the Impersonation category was clarified to require a direct claim of false authority or status in the text itself, rather than an inference from context. Fourth, the annotators agreed to default to None in cases of genuine uncertainty, on the grounds that under-labeling edge cases is preferable to over-labeling ambiguous ones.

6.1.2 Taxonomy Overview

The taxonomy is organized under six themes, each covering related types of ethical concerns. The themes are: (1) Respect and Inclusion, (2) Fairness in Participation and Decision-Making, (3) Responsibility and Accountability, (4) Privacy and User Autonomy, (5) Honesty (6) Ownership, Attribution and Obligation.

6.1.3 Theme 1: Respect and Inclusion

This theme covers ethical concerns that undermine the dignity and equal treatment of community members. It contains two categories. *Personal Attacks and Demeaning Communication* includes comments that attack individuals rather than addressing technical issues or ideas. For example: *"You're just a clear example of someone who spews nonsense and derails threads without a shred of link or evidence."* *Discrimination* covers bias or unfair treatment based on protected characteristics or group membership, such as: *"There is an Accessibility dimension to my request as currently the feature discriminates against people who are 'attentionally challenged'."* These two categories are distinguished by their target: Personal Attacks are directed at an individual's character or competence, while Discrimination targets someone because of their membership in a group.

Table 6.2: Summary of ethical concern categories

Category	Description
Respect and Inclusion	
Personal Attacks and Demeaning Communication	Comments that attack or demean individuals rather than addressing technical issues
Discrimination	Bias or unfair treatment based on protected characteristics or group membership
Fairness in Participation and Decision-Making	
Governance Fairness	Concerns about the legitimacy or fairness of decision-making processes
Lack of Transparency	Relevant information is not adequately shared or explained
Lack of Responsiveness	Legitimate questions, contributions, or concerns are ignored or left unanswered
Gatekeeping	Attempts to restrict another person's legitimacy to participate
Participation Shaming	Comments that shame legitimate participation
Responsibility and Accountability	
Responsibility Evasion	Avoidance of responsibility for problems or consequences
Power Abuse or Authority Dominance	Misuse of positions of authority or dominance over others
Irresponsible Release	Releasing software without adequate testing or consideration of harms
Irresponsible Disclosure	Inappropriate disclosure of security-related information
Privacy and User Autonomy	
Privacy Violation	Unauthorized collection, exposure, or sharing of user data
User Autonomy and Control Violation	Undermining users' control over their systems or data
Honesty	
Misrepresentation	False or misleading claims about status, capability, or authority
Impersonation	Pretending to be someone else or creating false identities
Ownership, Attribution and Obligation	
Attribution and Credit Issues	Disputes about proper recognition of contributions
Intellectual Property Violation	Unauthorized use of copyrighted code or license violations
Open Source Obligation Violation	Failure to meet obligations imposed by open-source licenses

6.1.4 Theme 2: Fairness in Participation and Decision-Making

This theme addresses concerns related to how community members participate in project governance and decision-making. It contains five categories. *Governance Fairness* covers concerns about the legitimacy or fairness of decision-making processes, for example: *"Your message signals that it isn't up for discussion, that you made your decision without community input, and you won't list your reasons." Lack of Transparency* covers situations where relevant information is not adequately shared, for example: *"This is a dupe of 365 which was closed, seemingly without any explanation." Lack of Responsiveness* captures patterns of failing to respond to legitimate contributions or concerns, for example: *"Why should we believe that an entity not capable of responding to the community for over a year is now able to?" Gatekeeping* covers attempts to restrict another person's right to participate, for example: *"Quite honestly, I don't see why this discussion needs to happen on GitHub... it should be locked to collaborators who are going to be working on the code." Participation Shaming* covers comments that shame or discourage legitimate participation, for example: *"90% of comments are like 'yeah me too' or 'that sucks'. Where the PRs guys? It's open source, not everybody can leech."* Gatekeeping and Participation Shaming are distinguished by their focus: Gatekeeping challenges a person's right to participate at all, while Participation Shaming criticizes how they are participating.

6.1.5 Theme 3: Responsibility and Accountability

This theme covers concerns about whether individuals and projects take appropriate responsibility for their actions and decisions. It contains four categories. *Responsibility Evasion* covers avoidance of responsibility for problems or consequences, for example: *"It was not fair to place blame for this on the SDK... please take responsibility for what is your responsibility without unfairly pointing fingers at other contributors in this open source ecosystem." Power Abuse or Authority Dominance* covers misuse of positions of authority over others, for example: *"The only 'abuse' I can see here is maintainers banning me for political reasons, which is an abuse of power." Irresponsible Release* covers releasing software without adequate testing or consideration of harms, for example: *"This software is so unstable it is unusable. I'm not sure why known bad code has been deliberately posted on GitHub, and I think that's unethical." Irresponsible Disclosure* covers inappropriate disclosure of security-related information, for example: *"Disclosure process was not followed, and CVE was created without contacting either me or TideLift, which is quite irresponsible."* Irresponsible Release and Irresponsible Disclosure are distinguished by their focus: Irresponsible Release concerns the harm caused by publishing unstable or dangerous software, while Irresponsible Disclosure concerns the premature or improper handling of security vulnerability information.

6.1.6 Theme 4: Privacy and User Autonomy

This theme covers concerns about the protection of user data and users' ability to control their own systems and data. It contains two categories. *Privacy Violation*

covers unauthorized collection, exposure, or sharing of user data, for example: *"We cannot add a plugin that silently spreads personal data in a free software, that would just be unethical." User Autonomy and Control Violation* covers situations where users' control over their systems or data is undermined, for example: *"I personally think installing Brew (or anything for that matter) without asking for the user's permission is a little unethical."* These two categories are distinguished by their focus: Privacy Violation concerns the handling of user data by the software, while User Autonomy and Control Violation concerns whether users are given meaningful choice and control over what the software does to their system.

6.1.7 Theme 5: Honesty

This theme covers concerns about honesty and accurate representation in open-source communication. It contains two categories. *Misrepresentation* covers false or misleading claims about status, capability, or authority, for example: *"Naming a published package 'aws' when you're not actually publishing on behalf of AWS is highly misleading and arguably unethical." Impersonation* covers situations where someone pretends to be someone else or claims false authority, for example: *"I am @daimon111 — the only account that speaks for daimon. @maiiko616 is impersonating me by using my wallet address and writing as if they are me. This is identity theft."* These two categories are distinguished by their focus: Misrepresentation involves making false or misleading claims about a project or its capabilities, while Impersonation involves falsely claiming to be another person or organization.

6.1.8 Theme 6: Ownership, Attribution and Obligation

This theme covers concerns about ownership, proper attribution, and obligations arising from open-source licenses. It contains three categories. *Attribution and Credit Issues* covers disputes about proper recognition of contributions, for example: *"It seems the manufacturer modified an upstream lib when they packaged it for the current firmware (without credit). This is confusing, and also a little unethical." Intellectual Property Violation* covers unauthorized use of copyrighted code or license violations, for example: *"What you have done with our code is totally unethical, you have copied our whole extension and uploaded it as your copyrighted work." Open Source Obligation Violation* covers failure to meet obligations imposed by open-source licenses, for example: *"There is still no written offer of source code alongside your application that I could find, which is a violation of the license."* Attribution and Credit Issues and Intellectual Property Violation are distinguished by their focus: Attribution and Credit Issues concern the recognition of contributions within a project, while Intellectual Property Violation concerns the unauthorized use or redistribution of code in violation of copyright or license terms.

6.1.9 Distribution of Ethical Concerns in the Sample

The distribution of the final consensus labels across the full 134 comments is shown in Figure 6.1. Personal Attacks and Demeaning Communication, Gatekeeping, and

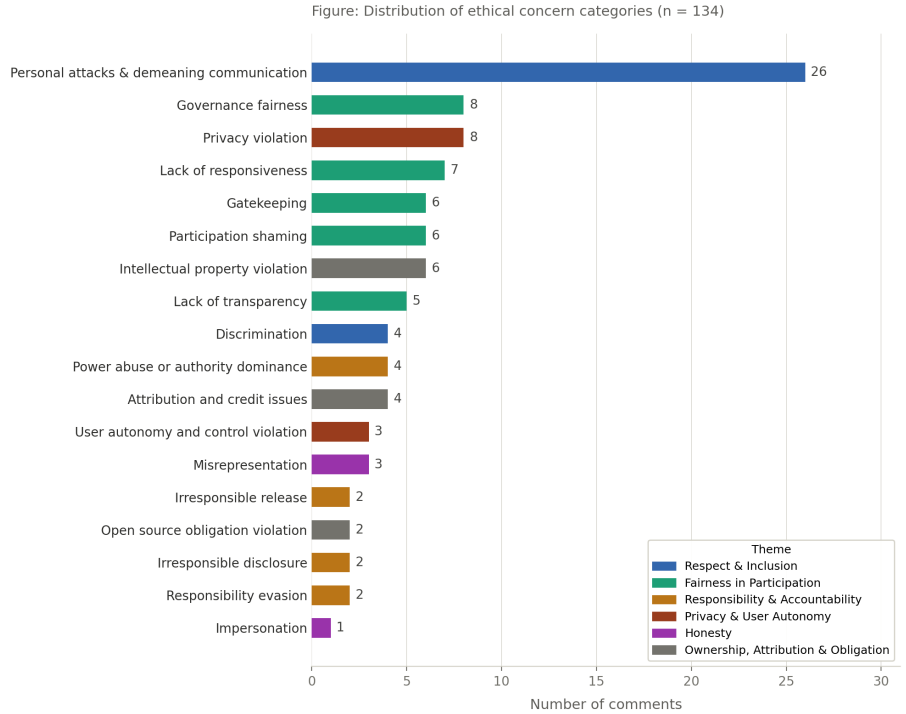

Figure 6.1: Distribution of ethical concern categories (n = 134)

Participation Shaming were the most frequently occurring categories. These categories together make up a large part of the observed cases. On the other hand, categories like Privacy Violation, Irresponsible Disclosure, and Open Source Obligation Violation do not appear very often. They are mostly found in the keyword-sampled part of our dataset. This uneven distribution shows that there is a substantial class imbalance across categories, which will affect our classification task in RQ2.

6.2 Automated Classification (RQ2)

6.2.1 Inter-Rater Reliability

Inter-rater reliability was assessed in two stages corresponding to the hierarchical structure of the annotation task.

In the first stage (IRR-A), three annotators independently coded a pilot sample of 150 comments to evaluate agreement on the binary distinction between concern and None. All pairwise Cohen's Kappa values ranged from 0.714 to 0.762, and Fleiss' Kappa across all three annotators was 0.741, as shown in Table 6.3. According to the scale proposed by Landis and Koch [23], values in this range indicate substantial agreement, suggesting that annotators could reliably distinguish ethical concern comments from non-concern comments.

In the second stage (IRR-B), we coded a separate sample of concern-positive comments to evaluate agreement on choosing one of the 18 fine-grained categories. For

Table 6.3: IRR-A inter-rater reliability results (binary classification)

Comparison	Kappa value
Annotator 1 vs Annotator 2	0.746
Annotator 1 vs Original coder	0.762
Annotator 2 vs Original coder	0.714
Fleiss' Kappa (all three annotators)	0.741

this stage, pairwise Cohen's Kappa values ranged from 0.622 to 0.668, and Fleiss' Kappa across all three annotators was 0.636. According to the scale, this indicates substantial agreement.

As shown in Table 6.4, agreement varied considerably across categories. Some categories were straightforward to code consistently. For example, Privacy Violation scored 0.952 and Attribution and Credit Issues scored 0.900. This is because their definitions refer to clearly distinct phenomena, and they have little conceptual overlap with other categories.

However, some categories were more difficult, particularly Gatekeeping (0.433) and Participation Shaming (0.424). These two categories are actually conceptually adjacent. Both of them involve comments directed at a contributor's participation, but they have a key difference: Gatekeeping challenges whether someone has the right to participate at all, while Participation Shaming criticizes the way they are participating. In practice, when comments challenge a person's standing, they often use dismissive language that looks like Participation Shaming. This makes the distinction depend on subtle differences in wording, which are not always easy to judge consistently. Because of this, these lower-agreement categories were discussed during the consensus meeting to clarify their boundaries.

Following each IRR assessment, the three annotators held a consensus meeting to review disagreements and refine category boundary definitions. During the discussions, we identified three recurring sources of ambiguity.

First, annotators sometimes disagreed between Intellectual Property Violation and Open Source Obligation Violation. The boundary was clarified as follows: Intellectual Property Violation applies when the concern is about unauthorized use of proprietary code without permission from the rights holder. Open Source Obligation Violation applies when the concern is about failing to fulfil the specific obligations imposed by an open source license. For example, a comment such as "if you do not have a contributors agreement that allows you to re-license contributed code... Just 'fixing' the license text now and thus changing the license in an AGPL incompatible way would be a license breach in itself" was coded as Open Source Obligation Violation rather than Intellectual Property Violation, because the concern was specifically about failing to fulfil the obligations imposed by the AGPL license, rather than about unauthorized use of proprietary code.

Table 6.4: IRR-B inter-rater reliability results (18-category classification)

Overall agreement	
Annotator 1 vs Annotator 2	0.668
Annotator 1 vs Original coder	0.622
Annotator 2 vs Original coder	0.625
Fleiss' Kappa (all three annotators)	0.636
Category-level Fleiss' Kappa	
Attribution and Credit Issues	0.900
Discrimination	0.799
Gatekeeping	0.433
Governance Fairness	0.665
Intellectual Property Violation	0.655
Impersonation	0.799
Irresponsible Disclosure	0.798
Irresponsible Release	0.697
Misrepresentation	0.704
Open Source Obligation Violation	0.718
Participation Shaming	0.424
Power Abuse or Authority Dominance	0.676
Privacy Violation	0.952
Responsibility Evasion	0.713
User Autonomy and Control Violation	0.850
Lack of Responsiveness	0.576
Lack of Transparency	0.713
Personal Attacks and Demeaning Communication	0.622

The second area of disagreement was between Governance Fairness and Lack of Transparency. We settled on the following boundaries: Governance Fairness applies when the concern is about whether a decision-making process was fair or inclusive, for example when certain contributors felt they had been excluded from a decision. Lack of Transparency applies when the concern is about whether sufficient information was shared about a decision, for example when a decision was made but no explanation was provided to the community. The key distinction is that Governance Fairness focuses on who was involved in making the decision, while Lack of Transparency focuses on whether the reasoning behind the decision was communicated.

The third issue involved Participation Shaming and Gatekeeping. The key distinction is that Gatekeeping applies when a comment challenges whether someone has the right or standing to participate at all, while Participation Shaming applies when a comment makes someone feel bad about the way they are participating, for example by implying that their question is stupid or a waste of time. Following the consensus meeting, the coding guidelines were updated to reflect these clarifications, and the relevant portions of the dataset were re-examined to ensure consistent application of the refined definitions.

Table 6.5: Internal cross-validation results for Stage 1 (binary classification)

Configuration	Macro-F1	Accuracy
TF-IDF + LR	**0.6801** ± 0.0254	0.7057
TF-IDF + SVM	0.6560 ± 0.0232	0.7197
SentEmb + LR	0.6613 ± 0.0154	0.6797
SentEmb + SVM	0.6413 ± 0.0126	0.7114

6.2.2 Dataset Statistics

The labeled dataset constructed for RQ2 comprises 3,084 artifacts in total. Of these, 2,022 are assigned the label None and 1,062 are labeled as ethical concerns distributed across the 18 categories. The development set contains 2,924 samples drawn from all repositories except ansible/ansible, and the external validation set contains 160 samples from ansible/ansible. Among the concern-positive examples in the development set, the most frequently represented categories are Participation Shaming (102 instances), Open Source Obligation Violation (94), and Attribution and Credit Issues (74), while the least represented are Irresponsible Disclosure (13), Privacy Violation (29), and Misrepresentation (25). At the theme level, Fairness in Participation and Decision-Making is the largest theme (346 instances), followed by Ownership, Attribution and Obligation (232) and Responsibility and Accountability (156), while Honesty (61) and Privacy and User Autonomy (74) are the smallest. This distribution reflects a substantial class imbalance across both the 18-category and 6-theme settings, which motivates the use of macro-averaged F1 as the primary evaluation metric and balanced class weights during training.

6.2.3 Stage 1: Binary Classification

Across all four model configurations, the best-performing Stage 1 classifier is TF-IDF + LR, achieving a mean macro-F1 of 0.6801 (±0.0254, 95% CI ±0.0128) across 15 outer cross-validation folds, as shown in Table 6.5. The remaining configurations perform somewhat lower: TF-IDF + SVM achieves 0.6560 (±0.0232), SentEmb + LR achieves 0.6613 (±0.0154), and SentEmb + SVM achieves 0.6413 (±0.0126). The consistently lower performance of sentence-embedding-based configurations relative to TF-IDF configurations at this stage is somewhat unexpected, given that embeddings are generally better at capturing semantic similarity. One possible explanation is that ethical concerns in this dataset tend to be expressed through specific recurring vocabulary, such as phrases like "without explanation", "no response", or "you idiot", which are strong lexical indicators that TF-IDF is well suited to capture by assigning them high weights. It should be noted, however, that the keyword-based data collection strategy used in RQ2 may have amplified this effect, since comments retrieved through category-specific keywords are more likely to contain those keywords or closely related terms. Whether TF-IDF's advantage over sentence embeddings would hold on data collected through a different strategy remains an open question, and is discussed further in Chapter 7.

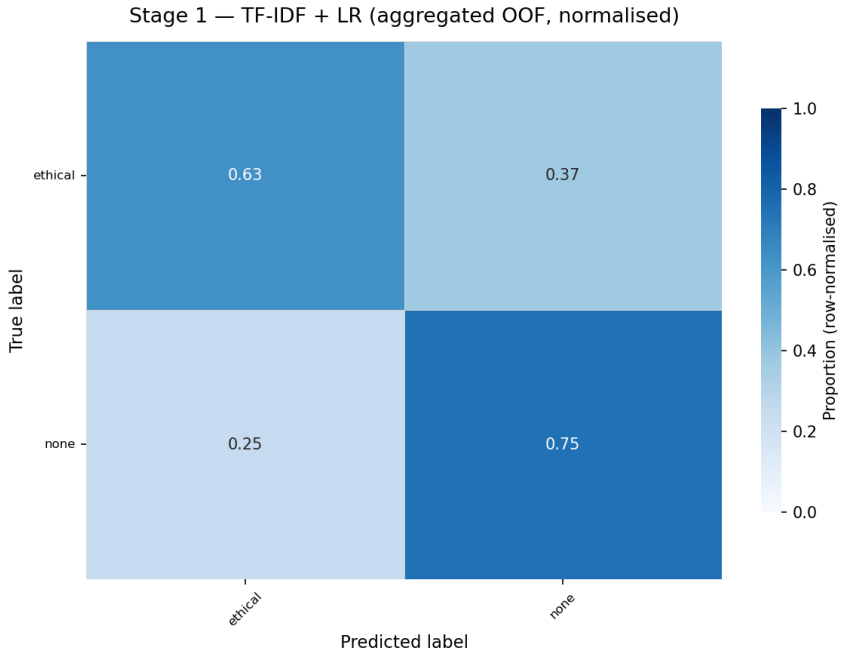

Figure 6.2: Aggregated out-of-fold confusion matrix for Stage 1 (TF-IDF + LR, normalised)

The aggregated out-of-fold confusion matrix for TF-IDF + LR (Figure 6.2) shows an asymmetry between the two classes. The classifier correctly identifies 63% of ethical concern comments but misses the remaining 37%, while correctly identifying 75% of None comments. This pattern indicates that the classifier is more conservative when predicting ethical concerns, which is consistent with the class imbalance in the dataset: with roughly twice as many None examples as ethical ones, the model is exposed to more non-concern language during training, making it more confident about the None class.

6.2.4 Stage 2: 18-Category Classification

For the 18-category classification task, the best-performing configuration is TF-IDF + SVM, achieving a mean macro-F1 of 0.7206 (±0.0470, 95% CI ±0.0238), as shown in Table 6.6. TF-IDF + LR achieves 0.7103 (±0.0470), while both sentence-embedding configurations perform substantially worse: SentEmb + LR achieves 0.5223 (±0.0415) and SentEmb + SVM achieves 0.5124 (±0.0414). The large gap between TF-IDF and sentence-embedding configurations at this stage suggests that fine-grained ethical concern categories in this dataset are largely distinguished by characteristic vocabulary rather than by paraphrase-level semantic similarity. This is plausible given that several categories are associated with distinctive terminology: for example, Irresponsible Disclosure comments frequently contain phrases such as "responsible disclosure" or "before the patch", and Attribution and Credit Issues comments frequently reference "no credit" or "original author".

As shown in Table 6.7, different categories have very different per-class performance

Table 6.6: Internal cross-validation results for Stage 2 (18-category and 6-theme settings)

Setting	Configuration	Macro-F1	Accuracy
18-category	TF-IDF + LR	0.7103 ± 0.0470	0.7196
	TF-IDF + SVM	**0.7206** ± 0.0470	0.7253
	SentEmb + LR	0.5223 ± 0.0415	0.5310
	SentEmb + SVM	0.5124 ± 0.0414	0.5310
6-theme	TF-IDF + LR	**0.7928** ± 0.0217	0.8096
	TF-IDF + SVM	0.7877 ± 0.0269	0.8077
	SentEmb + LR	0.6268 ± 0.0311	0.6588
	SentEmb + SVM	0.6120 ± 0.0347	0.6821

scores. Some categories achieve high performance, such as Attribution and Credit Issues (F1 = 0.836), Open Source Obligation Violation (F1 = 0.791), and User Autonomy and Control Violation (F1 = 0.792). This happens mostly because these categories usually use a very distinctive and consistent vocabulary, which makes them easy for the model to learn. In contrast, the model struggled much more with categories like Governance Fairness (F1 = 0.559), Intellectual Property Violation (F1 = 0.569), and Gatekeeping (F1 = 0.602).

The aggregated confusion matrix (Figure 6.3) confirms that the most frequent confusions involve conceptually adjacent categories. Gatekeeping is most often confused with Participation Shaming (11% of Gatekeeping instances predicted as Participation Shaming), which is consistent with the difficulty annotators also experienced distinguishing these two categories during the IRR assessment. Personal Attacks and Demeaning Communication shows notable confusion with Participation Shaming (10%) and Discrimination (7%). Governance Fairness is most often confused with Lack of Responsiveness (9%) and Power Abuse or Authority Dominance (11%), reflecting the conceptual overlap between concerns about unfair decision-making and concerns about authority being misused.

Intellectual Property Violation has a particularly high confusion rate with Attribution and Credit Issues. Specifically, 19% of Intellectual Property Violation instances were predicted as Attribution. This confusion makes sense because these two categories are conceptually very close. Both of them involve concerns about how others' work is used or credited, and the distinction between a legal rights issue and a recognition issue is not always clear from the text alone. Although we explicitly clarified this boundary in our coding guidelines after the IRR consensus meeting, the two categories share too many similar words. Therefore, surface-level features alone are not always enough to distinguish them reliably.

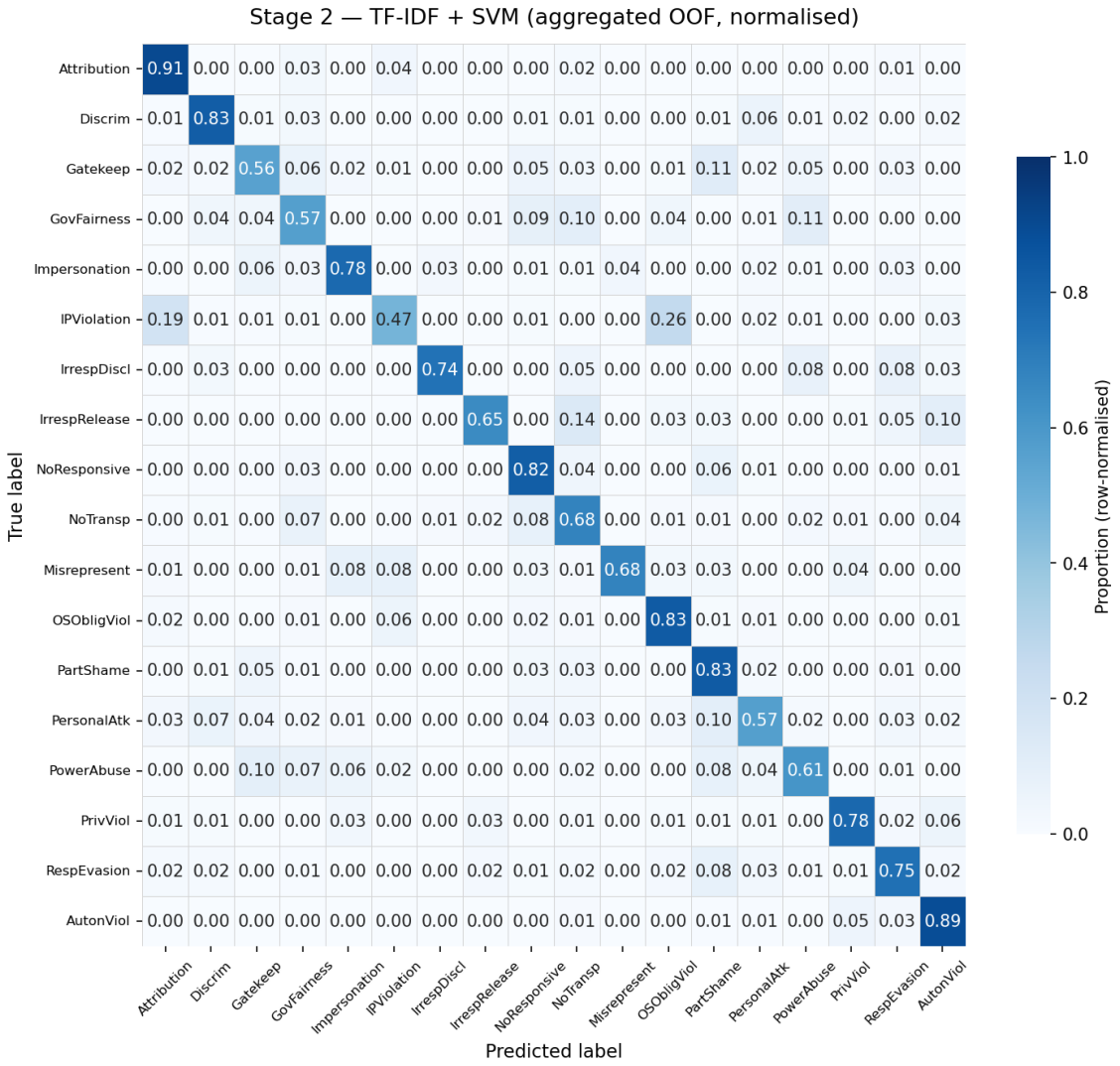

Figure 6.3: Aggregated out-of-fold confusion matrix for Stage 2, 18-category setting (TF-IDF + SVM, normalised)

6.2.5 Stage 2: 6-Theme Classification

When grouping the 18 fine-grained categories into six broader themes, the classification performance improves substantially. The best-performing configuration, TF-IDF + LR, achieves a mean macro-F1 of 0.7928 (±0.0217, 95% CI ±0.0110), compared to 0.7206 for the best 18-category configuration — an improvement of 0.0722 — and 0.7877 (±0.0269) for TF-IDF + SVM at the same granularity level, as shown in Table 6.6. Sentence-embedding configurations again perform considerably lower, with SentEmb + LR at 0.6268 (±0.0311) and SentEmb + SVM at 0.6120 (±0.0347). The improvement from 18-category to 6-theme classification is most pronounced for the sentence-embedding configurations, suggesting that when categories within the same theme are collapsed together, the semantic similarity that embeddings capture becomes more useful.

Per-theme performance is strong for Ownership, Attribution and Obligation (F1

Table 6.7: Per-class precision, recall, and F1 for Stage 2, 18-category setting (TF-IDF + SVM, aggregated OOF predictions, sorted by support)

Category	Precision	Recall	F1	Support
Irresponsible Disclosure	0.806	0.744	0.773	39
Misrepresentation	0.927	0.680	0.785	75
Privacy Violation	0.764	0.782	0.773	87
Impersonation	0.785	0.778	0.781	108
Irresponsible Release	0.837	0.649	0.731	111
Power Abuse or Authority Dominance	0.664	0.611	0.636	126
User Autonomy and Control Violation	0.714	0.889	0.792	135
Governance Fairness	0.552	0.567	0.559	150
Gatekeeping	0.655	0.557	0.602	174
Discrimination	0.815	0.828	0.821	186
Intellectual Property Violation	0.711	0.474	0.569	192
Responsibility Evasion	0.783	0.750	0.766	192
Lack of Responsiveness	0.709	0.824	0.762	204
Lack of Transparency	0.621	0.681	0.650	204
Personal Attacks and Demeaning Comm.	0.733	0.568	0.640	213
Attribution and Credit Issues	0.776	0.905	0.836	222
Open Source Obligation Violation	0.753	0.833	0.791	282
Participation Shaming	0.731	0.833	0.779	306

= 0.930) and Fairness in Participation and Decision-Making (F1 = 0.811), which are the two largest themes and benefit from more training examples (as shown in Table 6.8). Honesty achieves F1 = 0.772 and Privacy and User Autonomy achieves F1 = 0.808, both with relatively small sample sizes. The most challenging themes are Respect and Inclusion (F1 = 0.745) and Responsibility and Accountability (F1 = 0.693).

To find out where the model went wrong, we can check the confusion matrix in Figure 6.4. It reveals that the most common cross-theme errors involve Respect and Inclusion being predicted as Fairness (20% of Respect instances) and Responsibility and Accountability being predicted as Fairness (21% of Responsibility instances). Both patterns reflect the conceptual proximity of these themes: a dismissive comment directed at a person, for example, can simultaneously signal a respect concern and a fairness concern about how participation is being managed. Honesty also shows some confusion with Fairness (10% of Honesty instances), consistent with cases where a misleading statement is framed as a governance or transparency concern rather than a straightforward honesty issue.

6.2.6 External Validation

The external validation was conducted on 160 comments from ansible/ansible, including 60 ethical concern comments and 100 None comments. The distribution of concern categories and themes in the external set is shown in Table 6.9 and Ta-

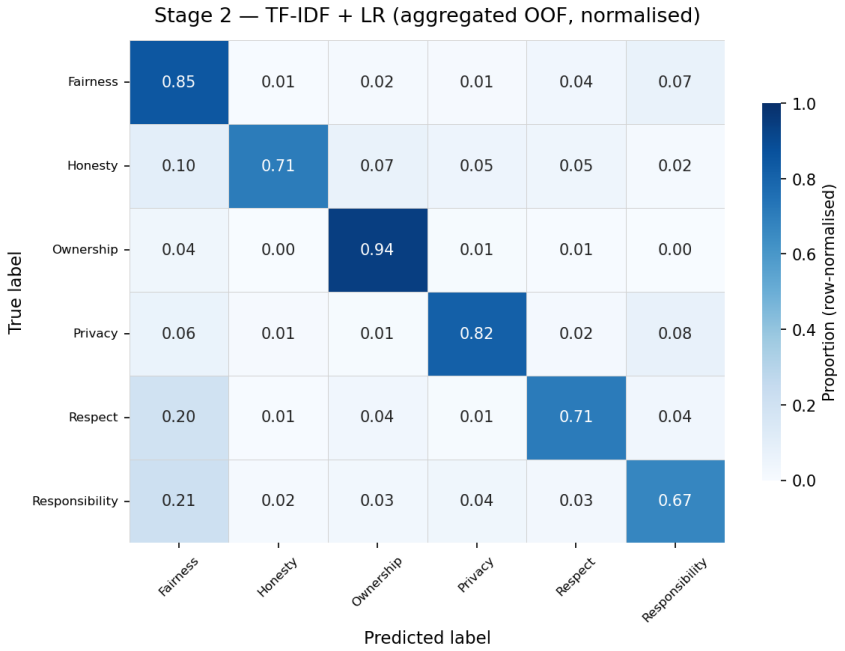

Figure 6.4: Aggregated out-of-fold confusion matrix for Stage 2, 6-theme setting (TF-IDF + LR, normalised)

Table 6.8: Per-class precision, recall, and F1 for Stage 2, 6-theme setting (TF-IDF + LR, aggregated OOF predictions, sorted by support)

Theme	Precision	Recall	F1	Support
Honesty	0.854	0.705	0.772	183
Privacy and User Autonomy	0.810	0.806	0.808	222
Respect and Inclusion	0.788	0.707	0.745	399
Responsibility and Accountability	0.744	0.647	0.693	468
Ownership, Attribution and Obligation	0.917	0.943	0.930	696
Fairness in Participation and Decision-Making	0.770	0.856	0.811	1038

ble 6.10. As the tables show, many fine-grained categories have very few instances, with three categories having zero instances entirely, which constrains the reliability of per-class evaluation at the 18-category level.

For Stage 1, our TF-IDF + LR classifier gets a macro-F1 of 0.7468 and an accuracy of 0.7812 on the external set. This result is substantially above the majority-class baseline, which is only 0.3846. Interestingly, it is also higher than our internal cross-validation mean of 0.6801 ($\pm$0.0254, 95% CI $\pm$0.0128).

If checking the external confusion matrix in Figure 6.5, it reveals how the classifier behaves on this new repository. It correctly catches 55% of the ethical concern comments here, which is lower than the 63% achieved in internal validation. This drop shows that recall for the ethical class gets a little worse when you test the model on an unseen repository. But at the same time, the precision for this ethical class

Table 6.9: Category distribution and per-class F1 in the external validation set (ansible/ansible, Stage 2 18-category setting)

Category	Support	F1
Personal Attacks and Demeaning Communication	11	0.143
Governance Fairness	6	0.800
Lack of Transparency	6	0.714
Misrepresentation	6	0.286
Lack of Responsiveness	5	0.909
Responsibility Evasion	5	0.500
Irresponsible Disclosure	4	0.667
Discrimination	3	0.286
Irresponsible Release	3	0.333
Participation Shaming	3	0.000
Power Abuse or Authority Dominance	3	0.000
Gatekeeping	2	0.222
Attribution and Credit Issues	1	1.000
Intellectual Property Violation	1	0.500
Open Source Obligation Violation	1	0.000
Impersonation	0	—
Privacy Violation	0	—
User Autonomy and Control Violation	0	—
Total concern-positive	60	
None	100	

remains high at 0.805, and the model correctly finds 92% of the None comments.

This whole pattern tells us that the classifier is more conservative rather than just noisy. Basically, when it decides to flag a comment as an ethical concern, it is usually right. So, why did our overall macro-F1 score look better than the internal cross-validation mean? This jump is mostly driven by how well the model handled the None class, which just reflects the class distribution of this external set. It does not mean the model achieved a genuine improvement in its detection capability.

For Stage 2 at the 18-category level, performance drops substantially on the external set. The TF-IDF + SVM classifier achieves a macro-F1 of 0.3533 on the full external set, compared to the internal CV mean of 0.7206 and the majority-class baseline of 0.0172, a drop of 0.3673 relative to internal validation. While the gap between internal and external performance is large, the classifier remains considerably above the baseline, indicating that it retains meaningful predictive signal on the unseen repository. This large gap is partly attributable to the extremely small support for many individual categories in the external set: several categories have only one to three instances, making per-class F1 scores unreliable and driving down the macro average. The external confusion matrix (Figure 6.6) shows that some categories with sufficient support are classified reasonably well, including Lack of Responsiveness

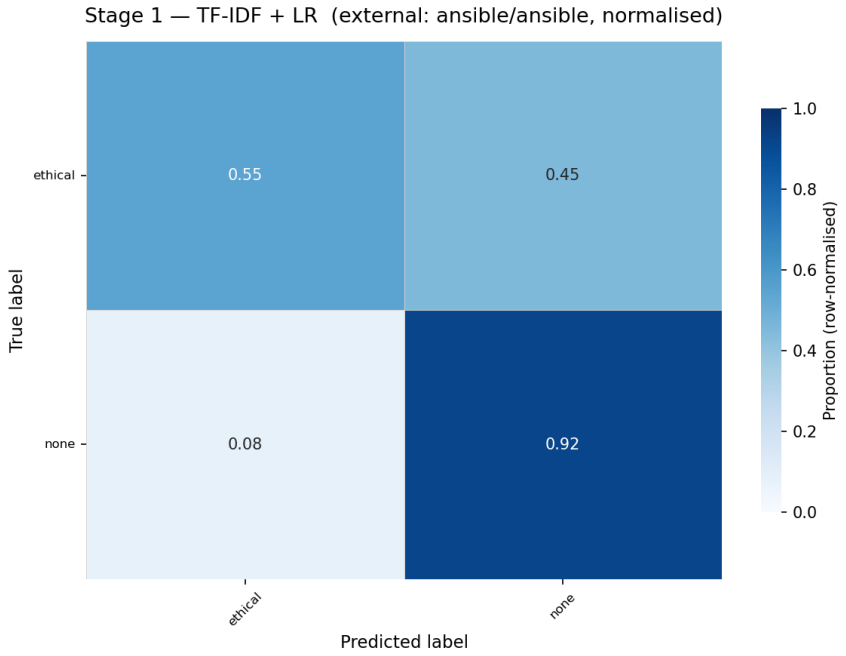

Figure 6.5: Confusion matrix for Stage 1 on the external validation set (ansible/ansible, normalised)

(F1 = 0.909), Governance Fairness (F1 = 0.800), and Attribution and Credit Issues (F1 = 1.000), while others with very small support such as Participation Shaming (F1 = 0.000, n = 3) and Personal Attacks and Demeaning Communication (F1 = 0.143, n = 11) perform poorly. These results are discussed further in Chapter 7.

Table 6.10: Theme distribution and per-theme F1 in the external validation set (ansible/ansible, Stage 2 6-theme setting)

Theme	Support	F1
Fairness in Participation and Decision-Making	22	0.667
Responsibility and Accountability	15	0.467
Respect and Inclusion	14	0.400
Honesty	6	0.286
Ownership, Attribution and Obligation	3	0.857
Privacy and User Autonomy	0	—
Total concern-positive	60	
None	100	

For Stage 2 at the 6-theme level, the performance drop is smaller but still notable. The TF-IDF + LR classifier achieves a restricted macro-F1 of 0.5352 on the external set (excluding Privacy and User Autonomy, which has zero instances), compared to the internal CV mean of 0.7928 and the majority-class baseline of 0.1073, a drop of 0.2524 relative to internal validation. The classifier again remains well above the baseline, suggesting that theme-level classification generalizes more re-

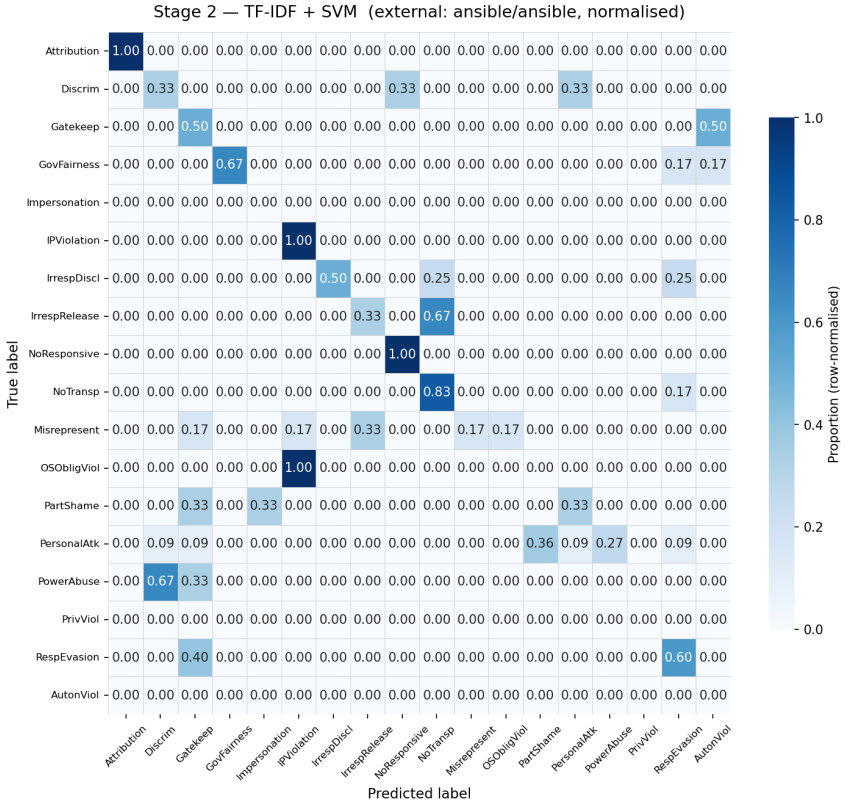

Figure 6.6: Confusion matrix for Stage 2, 18-category setting, on the external validation set (ansible/ansible, normalised). Categories with zero instances in the external set are included for completeness.

liably across repositories than fine-grained category classification. Among the five evaluable themes, Ownership, Attribution and Obligation achieves F1 = 0.857 and Fairness in Participation and Decision-Making achieves F1 = 0.667, while Honesty achieves only F1 = 0.286 due to low recall (0.167), with five of six Honesty instances misclassified as Fairness. Responsibility and Accountability achieves F1 = 0.467 and Respect and Inclusion achieves F1 = 0.400, as shown in Figure 6.7. These results are discussed further in Chapter 7.

6.2.7 Error Analysis

A manual examination of misclassified comments was conducted to identify recurring failure patterns in both the 18-category and 6-theme settings.

6.2.7.1 18-Category Setting

The best-performing Stage 2 model (TF-IDF + SVM) produced 815 errors out of 3,006 aggregated out-of-fold predictions, where each sample appears across multiple folds due to the repeated cross-validation design (3 repetitions × 5 folds) The categories with the highest absolute error counts are Intellectual Property Violation (101 errors), Personal Attacks and Demeaning Communication (92 errors), and Gatekeeping (77 errors). Notably, these error counts are high not only because these

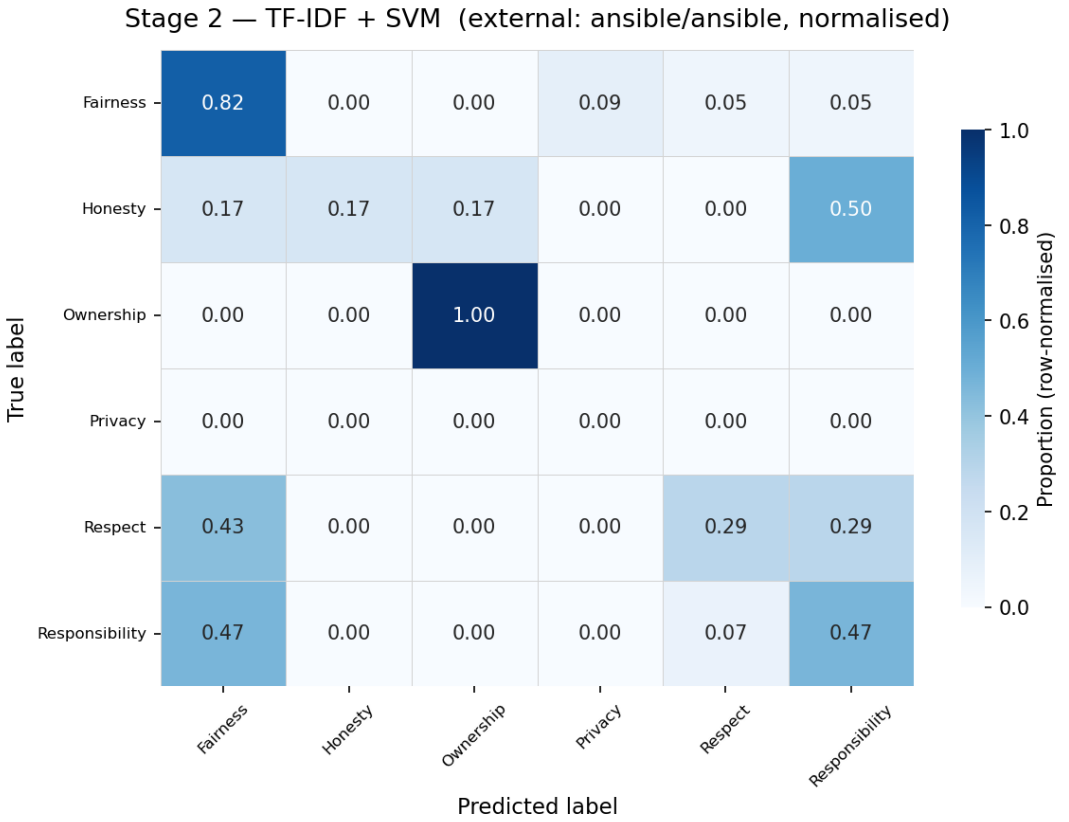

Figure 6.7: Confusion matrix for Stage 2, 6-theme setting, on the external validation set (ansible/ansible, normalised)

are large categories, but also because they are conceptually close to several other categories, making them harder for the classifier to distinguish reliably.

Several recurring confusion patterns are identifiable. The most frequent pattern involves categories that share surface vocabulary despite representing conceptually distinct concerns. Gatekeeping and Participation Shaming account for a large proportion of mutual confusions: comments that challenge a person's right to participate often use dismissive language similar to that found in Participation Shaming, making it difficult for the classifier to distinguish between questioning someone's standing to participate and criticizing the way they are participating. For example, a comment instructing someone to stop opening issues and read the title was predicted as Participation Shaming rather than Gatekeeping, even though the primary concern is the person's legitimacy to raise the issue at all.

A second recurring pattern involves Intellectual Property Violation and Open Source Obligation Violation, which share considerable lexical overlap around licensing terminology. Comments discussing license breaches are frequently misclassified between these two categories, reflecting the same boundary ambiguity that annotators encountered during the IRR assessment. Similarly, Attribution and Credit Issues and Intellectual Property Violation are frequently confused in both directions, consistent with comments that discuss code usage without crediting the original authors,

where the concern straddles recognition and legal rights simultaneously.

Governance Fairness is most often confused with Lack of Responsiveness, reflecting comments where a decision being made without community input is framed as a pattern of ignoring contributors rather than as a process fairness concern. Personal Attacks and Demeaning Communication shows notable confusion with Gatekeeping, because comments that aggressively tell someone they do not belong in a discussion can sound like a personal attack, even though the primary concern is about excluding them from participating rather than insulting them as a person.

6.2.7.2 6-Theme Setting

The best-performing Stage 2 model in the 6-theme setting (TF-IDF + LR) produced 568 errors out of 3,006 predictions. The dominant confusion pattern involves Fairness in Participation and Decision-Making acting as a catch-all prediction for several other themes. Responsibility and Accountability produces the highest number of errors (165), with many instances misclassified as Fairness, reflecting comments where a failure to take responsibility for a problem is framed in terms of how the project is being managed rather than who is accountable. Respect and Inclusion also shows substantial confusion with Fairness (117 errors), where dismissive or exclusionary comments are interpreted as governance or participation concerns rather than respect violations.

Honesty has a relatively small support size but produces 54 errors, most of which are misclassified as Fairness. The reason is straightforward: when people express concerns about being misled, they rarely say "this is dishonest." Instead, they describe what happened, for example "the issue was closed after falsely claiming a sponsor," which sounds more like a complaint about a decision than an accusation of deception. The classifier identifies this type of language and assigns it to Fairness, because the words used look more like a governance complaint than a honesty concern.

Privacy and User Autonomy also shows notable confusion with other themes. Comments in this category are sometimes misclassified as Ownership, Attribution and Obligation or Responsibility and Accountability because privacy concerns are often expressed in terms of what the project should or should not do for its users, rather than using explicit privacy-related vocabulary. For example, a comment arguing that an application should not use Google Analytics because it threatens user privacy may read more like a complaint about project obligations than a privacy violation, leading the classifier to assign it to the wrong theme.

Ownership, Attribution and Obligation performs well overall, but some errors occur when attribution disputes are expressed in a way that sounds like a governance complaint. For example, a comment arguing that a contributor's work was not acknowledged in a release may be written as "why was this contribution ignored by the maintainers", which the classifier reads as a Fairness concern about how decisions are made, rather than recognizing it as a concern about giving proper credit.

6.3 Temporal and Repository-level Analysis (RQ3 and RQ4)

Before examining ethical concern patterns, Figure 6.8 shows the monthly comment volume for each repository over the study period. ansible, react, and neovim consistently generate several hundred comments per month, while terraform shows a notable drop in activity from late 2023 onward, coinciding with the BSL license change and the emergence of the OpenTofu fork. flask remains at a much lower volume throughout, typically fewer than 50 comments per month, which explains the high variability in its ethical comment ratio observed in the following analysis.

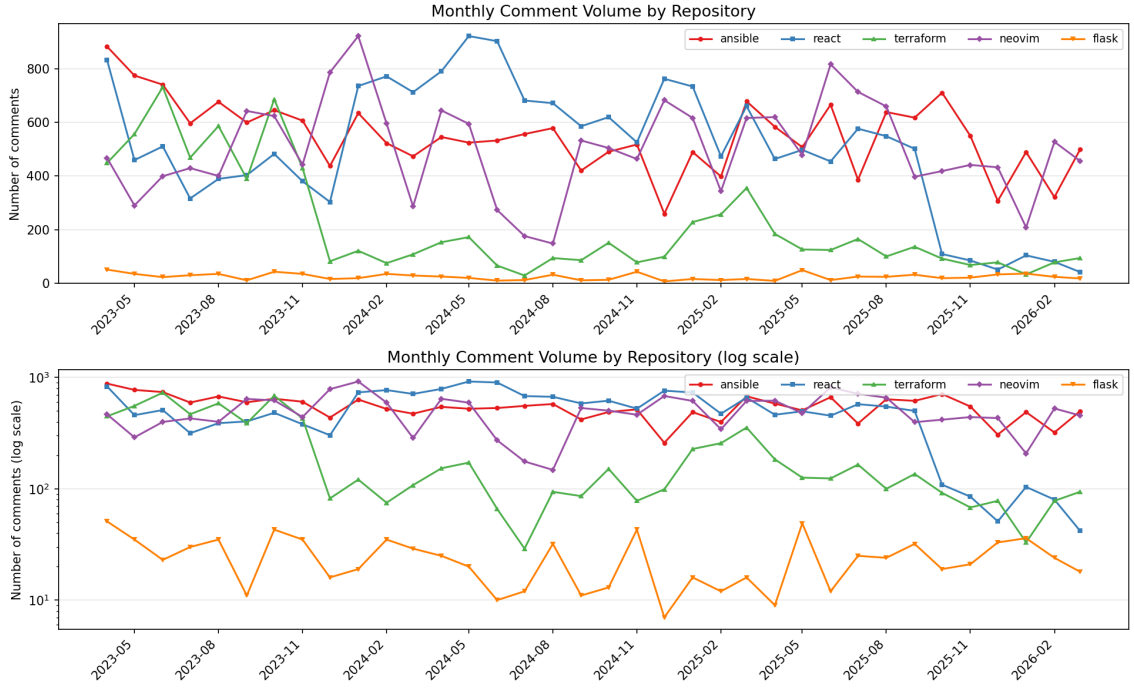

Figure 6.8: Monthly comment volume by repository over the 36-month study period, shown on linear scale (top) and log scale (bottom).

6.3.1 RQ3: Temporal Patterns of Ethical Concerns

Figure 6.9 shows the proportion of ethical comments aggregated across all five repositories over the 36-month study period. The overall ethical comment ratio remains relatively stable, fluctuating between approximately 0.06 and 0.11, with a mean of around 0.08. A modest downward trend is visible from mid-2023 onward, with the ratio gradually declining from around 0.09–0.10 in the early months to around 0.06–0.08 by early 2026, though month-to-month variation is considerable throughout.

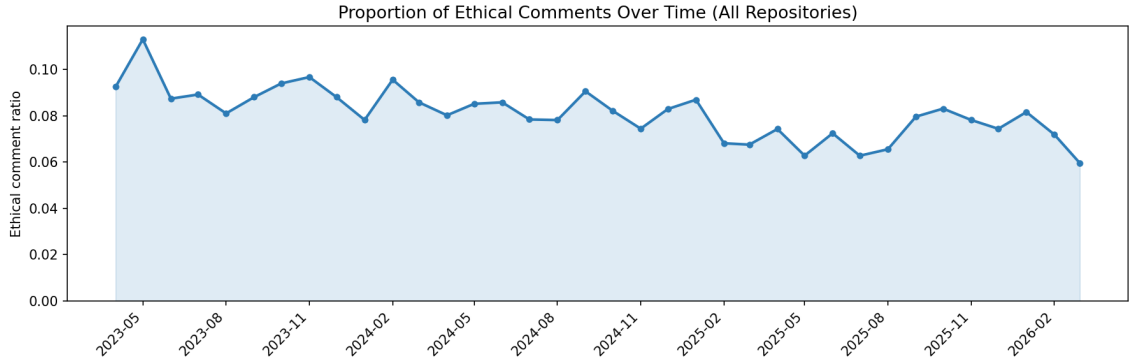

Figure 6.9: Proportion of ethical comments aggregated across all five repositories over the 36-month study period (April 2023 to April 2026).

Figure 6.10 presents the ethical comment ratio for each repository individually over the same period. The repositories differ considerably in both their mean ethical concern levels and the degree of month-to-month variability. ansible and react show relatively stable ratios with means of 0.086 and 0.085 respectively and modest standard deviations (0.016 and 0.028), suggesting consistent levels of ethical concern expression throughout the study period. neovim shows a similar level of stability with a mean of 0.070 (SD = 0.017). terraform exhibits greater variability (mean = 0.052, SD = 0.034), with a notable spike in ethical comment ratio around late 2023, followed by a sustained decline. flask shows the highest mean ethical comment ratio (0.192) and by far the greatest variability (SD = 0.093), with monthly values ranging from 0.031 to 0.444, which is consistent with its very low comment volume making the ratio unstable from month to month.

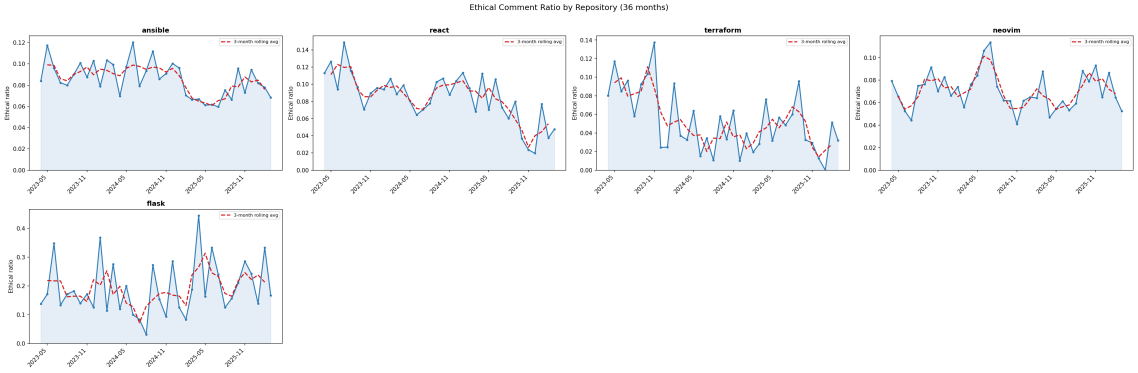

Figure 6.10: Ethical comment ratio by repository over the 36-month study period, with 3-month rolling average (dashed line).

A notable pattern in terraform is the elevated ethical comment ratio observed around October to December 2023, as shown in Figure 6.11. This period coincides with two significant events: HashiCorp's announcement of a Business Source License (BSL) change in August 2023, and the announcement of the OpenTofu fork in September 2023. The ethical comment ratio in terraform reached its highest point of 0.138

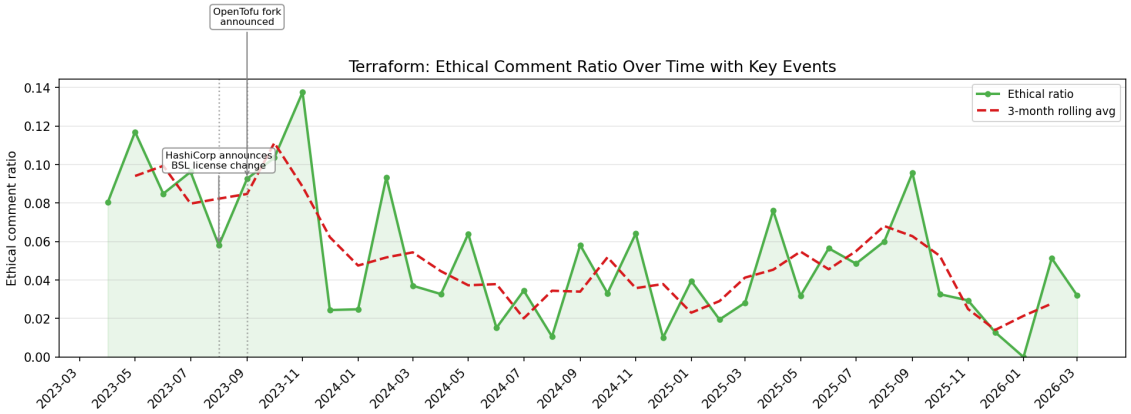

Figure 6.11: Ethical comment ratio for hashicorp/terraform over time, with key governance events annotated.

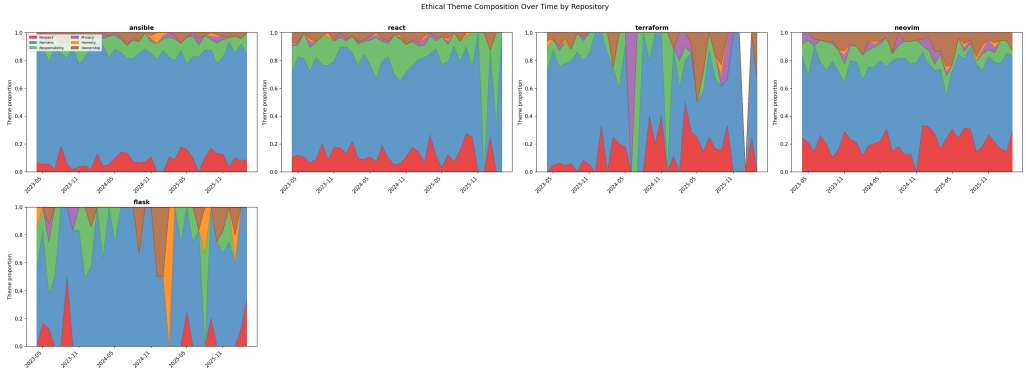

Figure 6.12: Theme composition of ethical comments over time by repository. Each colour represents one of the six ethical concern themes.

in November 2023 before declining sharply in the following months. This pattern suggests that governance-related events may be associated with short-term increases in ethical concern expression, though the causal relationship cannot be established from this analysis alone.

Figure 6.12 shows the theme composition of ethical comments over time for each repository. Across all four larger repositories, Fairness in Participation and Decision-Making (blue) consistently constitutes the largest share of ethical concern comments, typically accounting for more than half of all flagged comments in any given month. Ownership, Attribution and Obligation (brown) and Responsibility and Accountability (green) appear as secondary themes with relatively stable proportions. Respect and Inclusion (red) is present across all repositories but at lower proportions. Privacy and User Autonomy and Honesty appear infrequently and intermittently. In terraform, the theme composition shows greater fluctuation around the late 2023 period, with Ownership and Fairness themes becoming more prominent, which is consistent with the licensing and governance controversy during that period. flask's theme composition is highly variable due to its small sample size, and no stable pattern can be identified.

6.3.2 RQ4: Association Between Repository Characteristics and Ethical Concerns

Figure 6.13 presents the Spearman correlation coefficients between the four repository activity indicators and the ethical comment ratio, computed separately for each repository and for the pooled dataset. The per-repository and pooled results show strikingly different patterns, which are examined in turn.

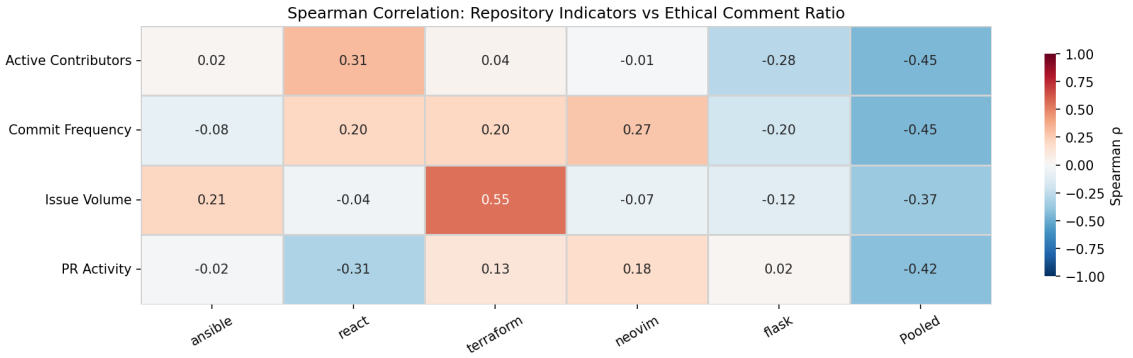

Figure 6.13: Spearman correlation coefficients between repository activity indicators and ethical comment ratio, computed per repository and for the pooled dataset (n = 180).

At the per-repository level, most correlations are weak and statistically non-significant. For ansible, react, neovim, and flask, all four indicators show correlations close to zero with p-values well above 0.05, indicating no detectable within-repository association between activity levels and ethical concern ratios for these projects. The one notable exception is terraform, where issue volume shows a moderate positive correlation with the ethical comment ratio ($\rho = 0.553$, p = 0.0005), meaning that months with more issues reported in terraform tend to coincide with higher proportions of ethical concern comments. This is the only statistically significant per-repository correlation across all 20 comparisons, and it is consistent with the terraform licensing controversy period, during which both issue volume and ethical concern expression were elevated.

At the pooled level, all four indicators show moderate negative correlations with the ethical comment ratio, all of which are statistically significant ($p < 0.001$): issue volume (ρ = -0.372), PR activity (ρ = -0.420), commit frequency (ρ = -0.453), and active contributors (ρ = -0.451). These pooled correlations suggest that repository-months with higher activity levels across all five repositories tend to have lower ethical comment ratios. Figure 6.14 illustrates these pooled associations, showing that the negative trend is driven largely by the contrast between flask, which has low activity and high ethical ratios, and the four larger repositories, which have high activity and lower ethical ratios.

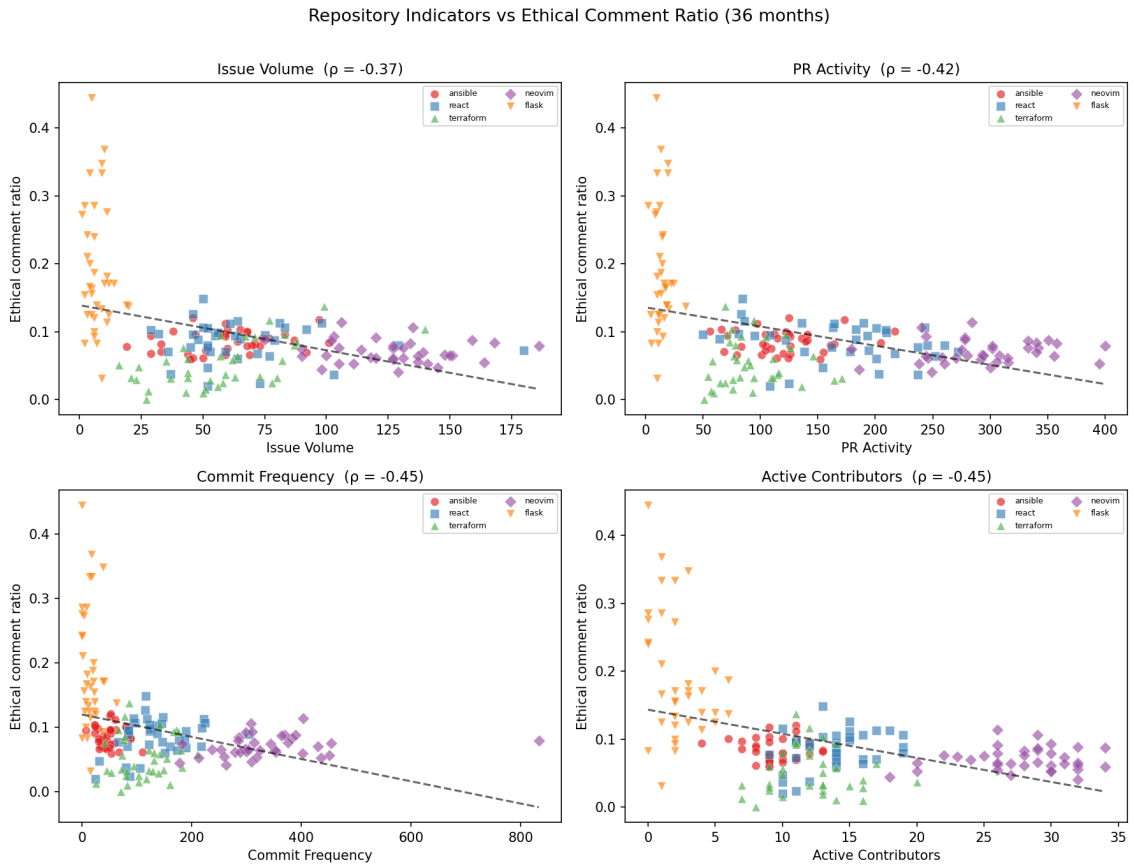

Figure 6.14: Scatter plots of repository activity indicators versus ethical comment ratio across all 180 repository-month observations. Each marker represents one repository-month, coloured by repository.

6.3.3 Sensitivity Analysis

The sensitivity analysis in Figure 6.15 examines how the pooled correlations change when flask is excluded from the data. The results show a substantial shift. Once flask is removed, the pooled correlations for all four indicators drop to near zero (issue volume: $\rho = +0.093$; PR activity: $\rho =$ -0.003; commit frequency: $\rho =$ -0.090; active contributors: $\rho =$ -0.061), and none of these would be statistically significant given n = 144 (36 months $\times$ 4 repositories). This indicates that the negative pooled correlations observed with all five repositories are driven primarily by flask's combination of low activity volume and high ethical comment ratio, rather than reflecting a consistent within-repository pattern.

Therefore, this sensitivity analysis suggests that the pooled negative correlations should be interpreted with caution. These scores actually reflect a between-repository contrast, rather than a genuine association between activity levels and ethical concern frequency within individual projects.

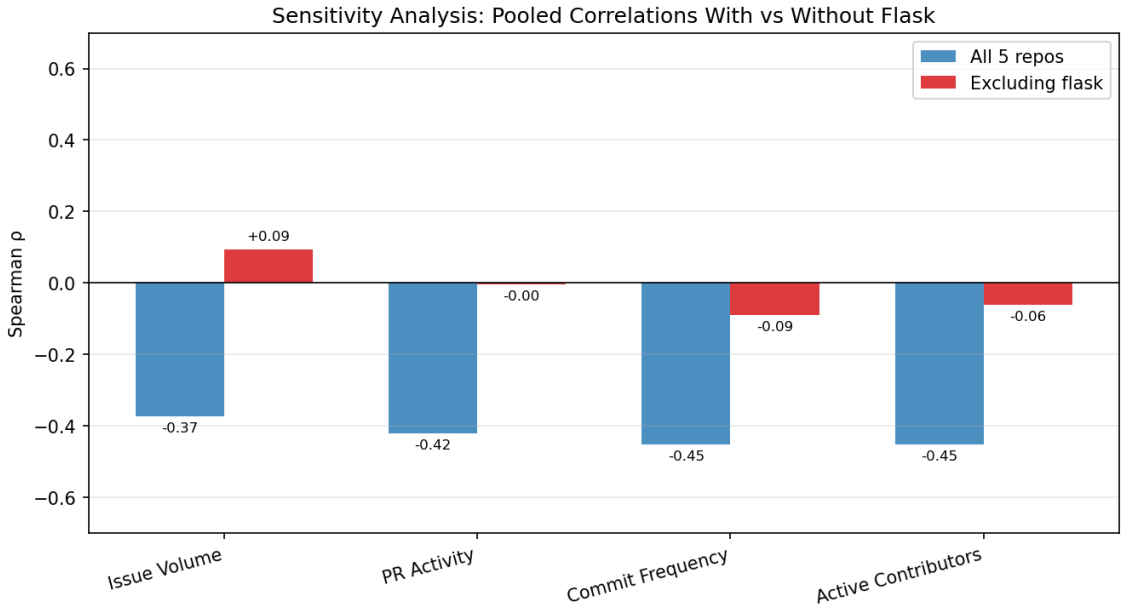

Figure 6.15: Sensitivity analysis: pooled Spearman correlations computed with all five repositories (n = 180) and with flask excluded (n = 144).

7

Discussion

7.1 The Taxonomy of Ethical Concerns in Open Source Development

7.1.1 Why Certain Concern Types Are More Prevalent Than Others

The distribution of ethical concern types in the dataset is not random. The most frequently occurring categories, including Participation Shaming, Open Source Obligation Violation, and Personal Attacks and Demeaning Communication, all arise from structural features of open source development rather than stemming purely from individual bad actors.

The high prevalence of Participation Shaming and Personal Attacks relates directly to the publicly accessible nature of open source projects. These spaces attract contributors with widely varying experience levels, ranging from seasoned developers to first-time contributors and end users. This diversity creates a persistent tension between the expectations of experienced community members and the behavior of newcomers, and this friction frequently surfaces as dismissive or demeaning responses. This is consistent with the findings of Tourani et al. [34], who observed that codes of conduct in open source projects frequently address interpersonal respect, suggesting that disrespectful communication is a recognized and persistent problem in these communities. The asymmetric power relationship between maintainers and contributors further amplifies this dynamic: maintainers hold gatekeeping authority over what gets merged, which makes dismissive responses from maintainers particularly impactful and more likely to be perceived as ethical violations worth commenting on.

Similarly, the frequent appearance of Open Source Obligation Violation reflects a growing awareness within open source communities regarding rights under open source licenses. High-profile cases of license violations by commercial organizations have received substantial public attention in recent years. Notable examples include the ongoing debates around companies building commercial products on top of MongoDB or Elasticsearch without contributing back. As a result, contributors are now much more likely to recognize and call out these violations. For instance, a

company incorporating a GPL-licensed library into a proprietary product without publishing the source code faces far more public challenges today than a decade ago. This heightened awareness ensures that Open Source Obligation Violation concerns are not only common in practice but are also explicitly raised in comment threads rather than ignored.

By contrast, categories such as Impersonation, Irresponsible Disclosure, and Privacy Violation are considerably less frequent. These behaviors tend to be more severe and more clearly prohibited by platform policies, and communities have established mechanisms for handling them outside of public comment threads. Irresponsible Disclosure, for example, involves security vulnerabilities that most communities handle through private channels by design, meaning that public comments about such issues are structurally suppressed rather than simply rare.

7.1.2 Concerns Not Captured by the Taxonomy

Another important point is that three dimensions from the early stages of this study, which are Self-Interest, Welfare, and Sustainability, did not have enough examples to become separate categories in the final dataset. However, this missing data does not mean that these concerns do not exist in open source communities. A better explanation is that these specific issues are usually expressed without using broad keywords like "unethical" or "unfair."

For example, a Self-Interest concern about a maintainer prioritizing features that benefit their employer might appear as a comment like "I notice all the recent changes are things Company X asked for", which raises the concern without using any ethical vocabulary.

The same thing happens with Sustainability and Welfare. Environmental impacts of software dependencies, which relate to Sustainability, are usually talked about as technical discussions about performance or resource use, rather than as ethical problems. Because of this, a simple keyword search will miss them. Similarly, a Welfare concern about software tracking people's locations might just be written as a plain technical observation: "this API could easily be used to monitor employees without their knowledge."

These gaps show that while the taxonomy successfully finds the ethical concerns that developers name directly, it likely misses problems that are talked about indirectly or that need more background information to understand.

7.1.3 Alignment and Gaps with Normative Frameworks

Turning to the relationship between the taxonomy and normative frameworks, the mapping onto the ACM Code of Ethics reveals both areas of alignment and notable gaps. Several of the most frequently occurring categories correspond directly to ACM principles: Personal Attacks and Demeaning Communication maps onto the principle of treating all people with respect and dignity, while Open Source Obli-

gation Violation corresponds to honoring intellectual property rights and fulfilling professional commitments.

However, the taxonomy also surfaces concerns that are less prominently addressed in normative frameworks, particularly those related to governance and participation. Governance Fairness and Participation Shaming are among the more frequently occurring categories in the dataset. At first glance, these concerns appear to be covered by ACM Principle 1.4, which calls for fair participation and prohibits harassment and abuse of power. However, the principle is written at a level of abstraction that does not fully reflect how these concerns actually manifest in open source communities.

For example, a Participation Shaming comment might say something like "just read the documentation" or "this has been answered a hundred times", which discourages participation in ways that ACM Principle 1.4 does not specifically address. Similarly, Governance Fairness concerns often arise when major decisions, such as changing a license or deprecating a feature, are made by a small group of maintainers without consulting the broader community, which may be perceived as unfair to contributors even though no formal rule has been broken. These are everyday social dynamics that fall into grey areas the ACM principle was not designed to adjudicate.

This suggests that normative frameworks like the ACM Code of Ethics are a useful starting point, but open source communities likely need more concrete, context-specific guidelines, for example explicit policies on how maintainers should communicate decisions, what counts as acceptable feedback on contributions, and how community members can raise concerns when they feel excluded. Codes of conduct that simply prohibit harassment and discrimination may not be sufficient to address the subtler everyday tensions that this study identifies.

7.1.4 Inter-Rater Reliability and Category Boundary Ambiguity

The inter-rater reliability results provide evidence that the taxonomy is sufficiently well-defined to support consistent annotation. The overall Fleiss' Kappa of 0.636 for 18-category classification falls within the substantial agreement range, which is a meaningful result given the difficulty of the task.

The variation in agreement across categories is itself informative. Categories with high agreement, such as Privacy Violation (0.952) and Attribution and Credit Issues (0.900), tend to have clear and distinctive definitions anchored in specific observable behaviors. Categories with lower agreement, such as Gatekeeping (0.433) and Participation Shaming (0.424), involve more interpretive judgment about the intent behind a comment and the social context in which it was made.

This pattern is consistent with the error analysis findings, where these same low-agreement categories also produced the most classifier confusions. Taken together, the IRR results and the error analysis suggest that participation-related categories

share closely adjacent conceptual boundaries, and that this adjacency reflects the fact that these concerns are often expressed using overlapping vocabulary in practice.

7.2 The Potential and Limitations of Automated Classification

7.2.1 Binary versus Fine-Grained Classification Performance

The classification results demonstrate that automated detection of ethical concerns in open source comments is feasible, but that performance varies considerably depending on the granularity of the task. The binary classifier achieves a cross-validation macro-F1 of 0.6801 and generalizes well to the external validation set (macro-F1 = 0.7468), suggesting that the presence or absence of an ethical concern can be detected with reasonable reliability across different repositories.

This is consistent with prior work demonstrating that automated classification of behavioral patterns in GitHub comments is feasible at scale. Sarker et al. [13] applied a domain-specific toxicity detector to over 100 million GitHub pull request comments, demonstrating that automated classification can handle volumes of data that would be impractical to review manually. The binary detection performance observed in the present study is lower, which is expected given that ethical concern detection covers a broader and more ambiguous set of phenomena than toxicity detection, and that the labeled dataset used here is considerably smaller than the training data available for toxicity classifiers.

The gap between binary and fine-grained classification performance is worth examining carefully. The 6-theme classifier achieves a cross-validation macro-F1 of 0.7928, while the 18-category classifier achieves 0.7206, and both drop substantially on the external validation set. This pattern reflects the fundamental difference between binary and fine-grained classification. At the binary level, the classifier only needs to detect that something is wrong, and comments containing ethical concerns often use recognizable complaint language that makes this detectable from surface words alone. At the fine-grained level, however, two comments can contain the same core phrase but belong to different categories depending on what else appears in the text. For example, a comment saying "this is not the place to raise this kind of issue, we only accept contributions from core team members" would be Gatekeeping because it explicitly challenges the person's right to participate. The same opening phrase in a comment like "this is not the place to raise this kind of issue, you clearly haven't read the contributing guidelines" would be Participation Shaming because the emphasis is on criticizing how the person is participating rather than denying their right to do so. The shared phrase carries the same surface signal, but the rest of the comment shifts the ethical category entirely. In practice, the words that distinguish these two cases, such as references to membership or standing on the one hand, and references to contribution quality or process on the other, may not appear consistently enough across all examples of each category for the classifier to learn them as reliable signals.

7.2.2 Why Theme-Level Classification Outperforms Category-Level Classification

The fact that the 6-theme classifier outperforms the 18-category classifier is not simply a consequence of having fewer classes to distinguish. A more substantive explanation is that collapsing 18 categories into 6 themes removes the most difficult classification boundaries while preserving the most meaningful ones. The categories that produce the most errors in the 18-category setting, such as Gatekeeping versus Participation Shaming, or Governance Fairness versus Lack of Responsiveness, all belong to the same theme, which means that when they are merged together, the classifier no longer needs to make these difficult within-theme distinctions. What remains are between-theme distinctions, such as distinguishing a Respect concern from an Ownership concern, which tend to be expressed using more clearly different vocabulary and are therefore easier to learn from text alone. This supports the decision to use theme-level predictions for the large-scale labeling in RQ3 and RQ4.

The error analysis further reveals that the most frequent misclassifications are not random but follow predictable patterns that reflect the way ethical concerns are expressed in practice. This has two implications.

First, it suggests that the classification errors are not primarily a technical failure of the model, but rather a reflection of the fact that some ethical concerns are expressed using overlapping vocabulary, even when the underlying concerns are conceptually distinct. For example, both Gatekeeping and Participation Shaming tend to appear in comments that use dismissive language directed at a contributor. Governance Fairness concerns are particularly prone to being misclassified as Lack of Responsiveness, because a contributor expressing that a decision was made without community input will often use language like "our concerns were ignored" or "maintainers never listened to us", which contains the same vocabulary as Lack of Responsiveness comments about maintainers failing to respond to issues or pull requests. The word "ignored" appears in both categories but means something different in each: in Governance Fairness it refers to community voices being excluded from a decision, while in Lack of Responsiveness it refers to technical contributions receiving no reply. A classifier working from surface words alone cannot distinguish between these two uses.

This is consistent with the lower IRR scores for the same categories: the classifier is struggling with the same linguistic overlap that human annotators also found challenging, which sets a realistic upper bound on what automated classification can be expected to achieve for these categories based on text alone.

Second, the error patterns suggest that improving classification for these boundary cases would likely require features that go beyond bag-of-words representations, such as information about the thread context or the relationship between the commenter and the maintainer, which are not available to the current classifier.

7.2.3 TF-IDF versus Sentence Embeddings

The consistent advantage of TF-IDF over sentence embeddings across both stages is somewhat counterintuitive, given that sentence embeddings are generally considered more semantically rich. One possible explanation is that ethical concerns in this dataset tend to be signaled by specific recurring vocabulary, which TF-IDF captures effectively by assigning high weights to distinctive terms.

This finding has broader implications for the text classification literature in software engineering. Lin et al. [14] demonstrated that opinion mining tools trained on general-purpose corpora consistently underperform on software engineering artifacts, attributing this gap in part to the domain-specific vocabulary that developers use. The present results extend this observation to the ethical dimension of developer communication: ethical concerns in GitHub discussions appear to be lexically rather than semantically distinctive, with comments expressing governance unfairness, attribution disputes, or responsibility evasion recurring around a stable set of domain-specific terms that TF-IDF captures effectively.

Sentence embeddings, by contrast, are optimised for semantic similarity across general-purpose text. Reimers and Gurevych [16] trained SBERT on natural language inference data drawn from general domains rather than specialist corpora, meaning a phrase carrying strong ethical weight in a software development context may be mapped close to superficially similar phrases from unrelated domains where that ethical weight is absent. This suggests that lexical models may be preferable to general-purpose semantic encoders for ethical concern classification unless the embeddings are fine-tuned on domain-specific data. It should be noted, however, that the keyword-based sampling strategy used in RQ2 may have amplified TF-IDF's advantage by over-representing comments containing the exact retrieval terms. Whether this advantage persists under a different sampling strategy remains an open question.

7.2.4 Cross-Repository Generalizability

The performance drop between internal cross-validation and external validation, particularly at the 18-category level, raises questions about cross-repository generalizability. One interpretation is that ethical concern language is sufficiently repository-specific that a classifier trained on one set of communities will struggle to generalize to an unseen community. An alternative explanation is that the external validation set is simply too small and too unevenly distributed across categories to provide a reliable estimate of generalization performance: with only 60 concern-positive instances spread across 15 categories, many categories have fewer than three instances in the external set, making per-class F1 scores highly sensitive to individual predictions.

Both explanations are likely to be partially true. Constructing a larger and more balanced external validation set would help disentangle them, but doing so would require either extending the data collection period or including additional repositories,

both of which were beyond the scope of this study given the time constraints.

The restricted macro-F1 of 0.5352 at the 6-theme level, which accounts for the uneven category distribution, provides a more reliable indication of generalization performance and suggests that theme-level classification generalizes more robustly than fine-grained category classification.

7.2.5 LLM-Based Classification as a Future Direction

The classification pipeline developed in this study relies on traditional supervised machine learning with TF-IDF features and linear classifiers. An alternative worth considering is the use of large language models (LLMs) such as GPT-4 or Claude, which have demonstrated strong performance on a wide range of text classification tasks without requiring task-specific training data.

LLMs offer several potential advantages for this task. First, they can handle linguistic variation more robustly than TF-IDF-based classifiers, since they represent meaning at a deeper semantic level and are not dependent on recurring keywords. This could improve performance on categories like Responsibility Evasion, where the same concern is expressed in many different ways. Second, LLMs can be prompted with the taxonomy definitions and decision rules from the coding framework, allowing them to classify comments in a zero-shot or few-shot setting without requiring a large labeled dataset. This would be particularly valuable for rare categories such as Irresponsible Disclosure, which have too few examples to train a reliable supervised classifier. Finally, LLMs may generalize more robustly across repositories, since their representations are not tied to the vocabulary of any specific training set.

However, LLMs also introduce important limitations and risks. Their outputs are not fully reproducible, since the same input can produce different classifications across runs. They are also computationally expensive and dependent on API access, making large-scale labeling of the kind conducted in RQ3 and RQ4 significantly more costly than running a local linear classifier. Perhaps most importantly, LLMs may introduce their own biases into the classification process: their training data reflects broad patterns in internet text, which may not align with the specific norms and vocabulary of open-source developer communication. Finally, applying an LLM to classify ethical concerns raises its own ethical questions about transparency and accountability, since the reasoning behind any classification decision is not directly inspectable.

7.3 Temporal Patterns of Ethical Concerns

The temporal and correlation findings reported in RQ3 and RQ4 are based on classifier-generated labels rather than manual annotations, which introduces an important limitation for the interpretation. The theme-level classifier achieved a restricted macro-F1 of 0.53 on the external validation set, meaning that a meaningful proportion of comments may be misclassified. Classification errors could affect the

observed trends in two ways. First, if certain themes are systematically over- or under-predicted, the observed theme compositions and their changes over time may not accurately reflect the true distribution of ethical concerns. Second, if misclassification rates vary across repositories due to differences in linguistic conventions or community-specific vocabulary, the ethical comment ratios may not be directly comparable across repositories. For these reasons, the temporal and correlation results should be interpreted as exploratory associations rather than definitive findings, and the specific patterns identified, such as the terraform spike, should be treated as possible signals that deserve further investigation with more reliable labeling in future work.

7.3.1 Overall Stability and Modest Decline Over Time

The most notable finding from the temporal analysis is that ethical concerns are a persistent and stable feature of open source development rather than an episodic phenomenon. Across all five repositories, the overall ethical comment ratio fluctuates within a relatively narrow band throughout the 36-month study period, with no evidence of a sustained increase over time. If anything, a modest downward trend is visible from mid-2023 onward, with the aggregate ratio gradually declining from around 0.09–0.10 in the early months to around 0.06–0.08 by early 2026.

This pattern is consistent with the view that ethical tensions in open source communities may be structurally embedded in how these communities operate, rather than being primarily driven by transient events or growing over time as communities scale. Prior work by Sarker et al. [13] found that toxic behavior on GitHub is widespread and persistent, and the present findings extend this observation to a broader range of ethical concern types: not only overtly toxic behavior but also governance concerns, attribution disputes, and responsibility evasion appear to be ongoing features of community life rather than problems that worsen or improve over time in any systematic way. The modest downward trend, if it reflects a real shift rather than measurement noise, could suggest that community health initiatives such as codes of conduct and moderation practices have had some gradual effect, though this interpretation would require further investigation to support.

It is also worth reflecting on the choice of a 36-month study window. Three years is long enough to observe meaningful variation and capture at least one significant governance event, as the terraform case demonstrates, but it is not long enough to capture the full lifecycle of a project or to observe how ethical concern patterns evolve as communities mature over decades. A longer time window extending back to the early years of these projects would risk introducing a confound that is difficult to control for: before codes of conduct became widespread in open source communities around 2015 to 2018, ethical tensions may have been expressed more frequently and more overtly, simply because there were fewer formal norms discouraging such behavior. By focusing on the 2023 to 2026 period, this study observes communities that already have established governance structures and behavioral norms, which means the ethical concern ratios observed here may be lower than they would have been in earlier periods. The three-year window also reflects practical constraints on

data collection and manual validation, and future work could extend this analysis to longer time horizons as automated classification tools improve.

7.3.2 Repository-level Variation and the Role of Governance Events

While the aggregate trend is stable, the repository-level analysis reveals considerably more variation and provides more interpretively rich findings. ansible and react show stable ethical comment ratios throughout the study period, with means of 0.086 and 0.085 respectively and low standard deviations, suggesting that the ethical culture of these communities is well established and relatively insensitive to month-to-month fluctuations in activity. neovim shows a similar pattern. terraform, by contrast, exhibits a sharp spike in ethical comment ratio around October to December 2023, coinciding with HashiCorp's announcement of the BSL license change in August 2023 and the subsequent announcement of the OpenTofu fork in September 2023, before declining sharply in the following months.

This terraform pattern is the most theoretically significant finding in the temporal analysis. It suggests that governance-level decisions, particularly those perceived as violating the open source social contract, can trigger short-term surges in ethical concern expression. This is consistent with the governance crises described in the Introduction of this thesis: just as the Python PEP 572 controversy and the Linux kernel conduct debate each generated intense community discussion before subsiding, the terraform license change appears to have produced a similar time-bounded spike in ethical tension. The ethical comment ratio nearly doubled in the months immediately following the license change before returning to baseline levels. This pattern most plausibly reflects a genuine community response to a perceived ethical violation, with the subsequent decline occurring as contributors who felt most strongly about the issue either migrated to the OpenTofu fork or disengaged from the project. Regardless of the exact mechanism, the terraform case suggests that ethical comment ratios may serve as a measurable signal of community stress around major governance decisions, which suggests that monitoring these ratios over time could provide an early indicator of community tension before it escalates into contributor attrition or project fragmentation.

7.3.3 Theme Composition and What It Reveals About Community Priorities

The theme composition analysis reveals two patterns worth discussing. The first is the remarkable stability of theme proportions within each repository over time: with the exception of terraform during the license controversy period, the relative contribution of each theme to the monthly ethical concern mix remains roughly consistent across all 36 months. This suggests that the ethical culture of a repository, in terms of what kinds of concerns tend to be raised, may be more closely associated with structural features of the community than with short-term fluctuations in activity. In other words, what a community tends to argue about ethically appears to be a

relatively stable property of that community, not something that varies from month to month depending on how busy the project is. Interestingly, this stability holds across repositories with very different governance structures and technical domains, from a corporate-backed UI library like react to a community-governed text editor like neovim, suggesting that the broad distribution of ethical concern types may be a general feature of open source development rather than something specific to particular project types.

The second pattern is the terraform exception. During the license controversy period, the theme composition shifted noticeably, with Ownership and Fairness themes becoming more prominent relative to their usual proportions. This suggests that governance events do not merely increase the volume of ethical concern expression but also change its character, with contributors shifting toward the specific themes most directly implicated by the governance change. This suggests that ethical comment ratios and theme compositions together may serve as a richer signal of community stress than either measure alone. From a practical standpoint, this implies that project maintainers and community managers could use shifts in theme composition as an early warning signal: a sudden increase in Ownership or Fairness concerns relative to the repository's usual baseline may indicate that a governance decision has been perceived as having ethical tensions.

Beyond the temporal patterns discussed above, the figure 6.12 also reveals differences in how themes are distributed across repositories. While Fairness in Participation and Decision-Making dominates in all four repositories, the secondary themes vary in meaningful ways. The most noticeable example is neovim, which shows a consistently higher proportion of Ownership, Attribution and Obligation concerns compared to ansible and react throughout the study period. One possible explanation is that this is connected to neovim's dual licensing situation: contributions predating the project's switch to Apache 2.0 remain under the Vim license, and code ported from Vim continues to carry the original Vim license, creating an ongoing layer of licensing ambiguity that has no equivalent in corporate-backed projects. React, by contrast, operates under Meta's governance with a formal contributor license agreement, where ownership and attribution are more uniformly defined from the outset. The fact that these differences remain stable across 36 months suggests that a repository's licensing history and governance structure shape not just how much ethical concern is expressed, but what kinds of concerns tend to surface.

Furthermore, all six themes are present across all four larger repositories throughout the study period, albeit in varying proportions, suggesting that the full range of ethical concern types identified in the taxonomy reflects genuine and ongoing tensions in open source communities rather than concerns that are specific to particular project types or governance structures. Flask is excluded from this observation due to its low comment volume, which makes monthly theme proportions unreliable.

7.4 Repository Characteristics and Ethical Concern Frequency

7.4.1 How Repository Characteristics Relate to Ethical Concern Frequency

The per-repository correlations reveal that within individual repositories, none of the four activity indicators, issue volume, PR activity, commit frequency, and active contributors, show a consistent or statistically significant association with the ethical comment ratio. The single exception is terraform's issue volume ($\rho = 0.553$, $p = 0.0005$), which is best understood as a governance event artifact rather than a general pattern: the spike in both issue volume and ethical comment ratio during the license controversy period drove this correlation, and it is unlikely to reflect a stable relationship between issue activity and ethical concern expression in terraform under normal circumstances.

This absence of within-repository associations is itself informative. One interpretation is that ethical concern expression in open source communities operates relatively independently of overall activity levels. Contributors raise ethical concerns when they perceive a violation of community norms, and this does not appear to happen more or less frequently simply because the project is having a particularly busy or quiet month. This is consistent with the theme composition stability finding in RQ3: if what a community argues about ethically is shaped by deeper features such as its governance structure, the power dynamics between maintainers and contributors, and the norms established early in the project's history, it would make sense that month-to-month fluctuations in how many issues are opened or how many commits are made do not meaningfully shift the ethical concern ratio.

7.4.2 Do the Correlations Imply a Causal Direction?

The pooled correlations across all five repositories show that all four activity indicators are negatively associated with the ethical comment ratio (ρ ranging from -0.372 to -0.453, all $p < 0.001$), meaning that repository-months with higher activity tend to have lower ethical comment ratios. Before interpreting this pattern, it is worth asking which direction the relationship runs, because correlation alone cannot establish causality.

One possibility is that high activity causes lower ethical concern ratios. For example, a project with many active contributors and frequent commits may have a healthier and more distributed community where no single maintainer dominates decisions, making ethical violations less likely to occur or less likely to go unaddressed. On this reading, being a busy and active project may itself serve as a protective factor against ethical tension.

The opposite possibility is equally plausible: projects with fewer ethical tensions may simply be more attractive to contributors, leading to higher activity as a con-

sequence. In other words, it is not that being busy reduces ethical concerns, but rather that having fewer ethical concerns makes a project a more welcoming place to contribute, which in turn drives up activity levels.

However, the sensitivity analysis makes both interpretations largely moot. When flask is excluded from the pooled analysis, all four correlations drop to near zero. This shift occurs because flask occupies a very distinctive position in the dataset: it has by far the lowest activity levels of the five repositories and simultaneously the highest mean ethical comment ratio (0.192, compared to 0.052 to 0.086 for the other four repositories). When flask is included, its data points cluster in the top-left corner of every scatter plot, high ethical ratio combined with low activity, which pulls the regression line into a negative slope. When flask is removed, the remaining 144 data points from the four larger repositories show no systematic pattern, and the correlations collapse to near zero. The pooled correlations therefore capture the contrast between flask and the other four repositories rather than a consistent within-repository or cross-repository relationship between activity levels and ethical concern frequency.

7.4.3 Protective Factors, Risk Factors, and Implications for Community Management

The results do not support a straightforward identification of protective or risk factors in the traditional sense, since no consistent within-repository associations were found. However, the findings do carry implications for how open source communities approach ethical concern management.

The absence of within-repository associations suggests that the commonly held intuition that busier projects experience more ethical conflict is not supported at the monthly level. Projects do not appear to enter higher-risk periods simply because they are receiving more issues or pull requests in a given month. This implies that reactive moderation strategies, such as increasing oversight during periods of high activity, may not be well targeted. A more effective approach may focus on the structural and governance features that are associated with a community's ethical culture over the long term, since these appear to be more stable than short-term activity fluctuations..

The terraform case illustrates that how major decisions are made and communicated matters more than how active the project is on a day-to-day basis. A unilateral license change that the community perceived as violating the open source social contract produced a more pronounced spike in ethical concern expression than any amount of routine activity variation, suggesting that governance transparency and community consultation are more relevant to ethical tension management than the volume of issues or commits in any given month. This implies that open source communities looking to reduce ethical tensions would be better served by investing in inclusive governance processes, such as public discussion periods before major decisions, than by monitoring activity metrics as proxies for community health.

8

Conclusion

This study investigated ethical concerns in open-source software repositories through a mixed-methods empirical research design combining inductive qualitative analysis, supervised machine learning classification, and exploratory temporal and repository-level analysis. The motivation for this work was that prior research had either relied on predefined ontological rules to detect specific concern types, or focused primarily on overtly toxic behavior, leaving the broader range of ethical tensions in everyday open-source development largely uncharted.

8.1 RQ1: Taxonomy of Ethical Concerns

The qualitative analysis of 134 GitHub issue and pull request comments from three repositories produced a taxonomy of 18 ethical concern types organized under six broader themes: Respect and Inclusion, Fairness in Participation and Decision-Making, Responsibility and Accountability, Privacy and User Autonomy, Honesty, and Ownership, Attribution and Obligation. Categories were derived inductively from the data before being mapped to the ACM Code of Ethics for theoretical grounding, ensuring that the taxonomy reflects what developers actually express rather than what existing frameworks anticipate. Inter-rater reliability assessment yielded a Fleiss' Kappa of 0.709, indicating substantial agreement across three independent annotators. The most frequently occurring concern types were Participation Shaming, Personal Attacks and Demeaning Communication, and Open Source Obligation Violation, all of which reflect structural features of how open-source communities operate rather than isolated incidents. Compared to Win et al., the taxonomy covers a wider range of concern types, including categories such as Governance Fairness and Participation Shaming that tend to be expressed without explicit ethical vocabulary and are not prominently addressed in existing normative frameworks.

8.2 RQ2: Automated Classification

A labeled dataset of 3,084 artifacts was constructed through keyword-assisted retrieval from the GitHub API and manual annotation. The binary classifier achieved a cross-validation macro-F1 of 0.6801, and the 18-category classifier achieved 0.7206. Aggregating the 18 categories into six broader themes improved macro-F1 to 0.7928,

which is consistent with the finding that within-theme category boundaries are more linguistically ambiguous than between-theme distinctions. TF-IDF representations outperformed sentence embeddings across all configurations, likely because ethical concerns in this dataset tend to be signaled by characteristic recurring vocabulary rather than by paraphrase-level semantic variation. External validation on ansible/ansible showed that binary classification generalized well to the unseen repository, with Stage 1 achieving a macro-F1 of 0.7468. Theme-level classification on the external set achieved a restricted macro-F1 of 0.5352, remaining well above the majority-class baseline and showing more robust generalization than fine-grained 18-category classification.

8.3 RQ3: Temporal Patterns

The theme-level classifier was applied to 54,605 comments across five repositories over a 36-month period from April 2023 to April 2026. The aggregate ethical comment ratio was relatively stable throughout, fluctuating between approximately 0.06 and 0.11, with a modest downward trend from mid-2023 onward. Most repositories maintained consistent ethical concern ratios and stable theme compositions over time, which suggests that the ethical culture of a community may be more closely associated with its governance structure and social norms than with short-term changes in activity levels. The most notable exception was hashicorp/terraform, where the ethical comment ratio nearly doubled following HashiCorp's Business Source License change in August 2023 and the announcement of the OpenTofu fork in September 2023, before returning to baseline over subsequent months. This pattern suggests that major governance decisions perceived as violating the open-source social contract can produce measurable short-term increases in ethical concern expression, and that monitoring the ethical comment ratio over time may provide a useful signal of community stress around such events.

8.4 RQ4: Project Characteristics and Associated Ethical Concerns

Spearman correlation analysis between four repository activity indicators (issue volume, pull request activity, commit frequency, and active contributor count) and the monthly ethical comment ratio found no consistent within-repository associations. For ansible, react, neovim, and flask, all per-repository correlations were weak and statistically non-significant. The one exception was terraform, where issue volume showed a moderate positive correlation ($\rho = 0.553$, $p = 0.0005$), which is most plausibly explained by the governance controversy period rather than a stable relationship between activity and ethical concern expression. At the pooled level, all four indicators showed moderate negative correlations with the ethical comment ratio, but a sensitivity analysis showed that removing flask caused all pooled correlations to drop to near zero. The pooled negative correlations therefore reflect the contrast between flask, which has low activity and a high ethical comment ratio, and the four

larger repositories, rather than a consistent pattern across projects. Taken together, these results suggest that ethical concern expression may be more closely associated with deeper structural features of the community than with how busy a project is in any given month.

8.5 Contributions and Implications

This study makes three contributions to research and practice.

For researchers, it provides an empirically grounded taxonomy of 18 types of ethical concerns that can be applied to other diverse open-source environments and used as a basis for future annotation efforts. it shows that supervised machine learning can support automated detection of ethical concerns at scale, with theme-level classification offering a practical balance between granularity and reliability, though external validation indicates that generalization to unseen repositories remains limited, particularly at the fine-grained category level.

For practitioners, the temporal analysis suggests that ethical comment ratios and theme compositions may serve as useful signals for community management. For example, sharp deviations in Ownership or Fairness themes relative to a project's baseline indicate rising tension from recent governance choices. The findings further suggest that investing in inclusive governance processes and clear communication around major decisions is more likely to reduce ethical tensions than tracking monthly activity metrics as proxies for community health.

8.6 Limitations and Future Work

Several limitations apply to this study. All repositories analyzed are English-language projects hosted on GitHub, and the data collection keywords are exclusively in English, so the taxonomy and classifiers may not transfer well to communities that discuss in other languages or use other platforms. The external validation set was small and unevenly distributed across categories, which limits the conclusions that can be drawn about fine-grained cross-repository generalization. The keyword-based data collection strategy for RQ2 may have favored TF-IDF over sentence embeddings, and whether this advantage holds under different sampling strategies remains an open question. The 36-month study window also covers a period when most of the selected repositories already had established governance structures and codes of conduct, so the findings may not reflect ethical concern patterns in earlier or less structured phases of project development.

Future research should deploy this classification pipeline to multilingual communities and alternative platforms to evaluate cross-context generalizability. Expanding the volume and balance of external validation data will allow for a more comprehensive assessment of cross-project classification limits. Integrating contextual indicators beyond raw text, such as commenter roles or local thread structures, could resolve the boundary issues currently impacting fine-grained classification for categories like

Gatekeeping and Participation Shaming. Extending the temporal analysis back to the period before codes of conduct became common in open-source communities would make it possible to examine whether and how formal governance mechanisms have affected the prevalence and character of ethical concerns over time. As open-source software becomes increasingly central to critical digital infrastructure, understanding the ethical dynamics of the communities that produce it is a problem that deserves continued empirical attention.

Bibliography

[1] LWN.net. Guido van Rossum resigns as Python leader. https://lwn.net/Articles/759654/, July 2018. Accessed: June 14, 2026.

[2] Noam Cohen. After years of abusive e-mails, the creator of Linux steps aside. *The New Yorker*, September 2018. Accessed: June 14, 2026. URL: https://www.newyorker.com/science/elements/after-years-of-abusive-e-mails-the-creator-of-linux-steps-aside.

[3] Hsu Myat Win, Haibo Wang, and Shin Hwei Tan. Towards automated detection of unethical behavior in open-source software projects. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2023)*, pages 644–656. Association for Computing Machinery, 2023. doi:10.1145/3611643.3616314.

[4] Aakash Sorathiya and Gouri Ginde. Ethical software requirements from user reviews: A systematic literature review. *arXiv preprint arXiv:2410.01833*, September 2024. doi:10.48550/arXiv.2410.01833.

[5] Association for Computing Machinery. ACM code of ethics and professional conduct. https://www.acm.org/code-of-ethics, 2018. Accessed: June 14, 2026.

[6] Eirini Kalliamvakou, Georgios Gousios, Kelly Blincoe, Leif Singer, Daniel M. German, and Daniela Damian. An in-depth study of the promises and perils of mining GitHub. *Empirical Software Engineering*, 21(5):2035–2071, October 2016. doi:10.1007/s10664-015-9393-5.

[7] David Kocsis and Gert-Jan de Vreede. Towards a taxonomy of ethical considerations in crowdsourcing. In *Proceedings of the 22nd Americas Conference on Information Systems (AMCIS)*. AIS Electronic Library, 2016.

[8] Seblewongel E. Biable, Nuno M. Garcia, and Dida Midekso. Proposed ethical framework for software requirements engineering. *IET Software*, 17(4):526–537, 2023. doi:10.1049/sfw2.12136.

[9] Jane Hsieh, Joselyn Kim, Laura Dabbish, and Haiyi Zhu. Nip it in the bud:

Moderation strategies in open source software projects and the role of bots. *Proceedings of the ACM on Human-Computer Interaction*, 7(CSCW2):1–29, 2023. doi:10.1145/3610092.

[10] Vandana Singh, Brice Bongiovanni, and William Brandon. Codes of conduct in Open Source Software—for warm and fuzzy feelings or equality in community? *Software Quality Journal*, 30(2):581–620, 2022. doi:10.1007/s11219-020-09543-w.

[11] Erika Halme, Marianna Jantunen, Ville Vakkuri, Kai-Kristian Kemell, and Pekka Abrahamsson. Making ethics practical: User stories as a way of implementing ethical consideration in software engineering. *Information and Software Technology*, 167:107379, 2024. doi:10.1016/j.infsof.2023.107379.

[12] Courtney Miller, Sophie Cohen, Daniel Klug, Bogdan Vasilescu, and Christian Kästner. "Did you miss my comment or what?": Understanding toxicity in open source discussions. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*, pages 710–722, 2022. doi:10.1145/3510003.3510111.

[13] Jaydeb Sarker, Asif Kamal Turzo, and Amiangshu Bosu. The landscape of toxicity: An empirical investigation of toxicity on GitHub. *Proceedings of the ACM on Software Engineering*, 2(FSE):623–646, June 2025. doi:10.1145/3715744.

[14] Bin Lin, Nathan Cassee, Alexander Serebrenik, Gabriele Bavota, Nicole Novielli, and Michele Lanza. Opinion mining for software development: A systematic literature review. *ACM Transactions on Software Engineering and Methodology*, 31(3):1–41, March 2022. doi:10.1145/3490388.

[15] Rafael Kallis, Andrea Di Sorbo, Gerardo Canfora, and Sebastiano Panichella. Ticket tagger: Machine learning driven issue classification. In *2019 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 406–409, 2019. doi:10.1109/ICSME.2019.00070.

[16] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 3982–3992. Association for Computational Linguistics, 2019.

[17] Virginia Braun and Victoria Clarke. Using thematic analysis in psychology. *Qualitative Research in Psychology*, 3(2):77–101, 2006. doi:10.1191/1478088706qp063oa.

[18] Daniela S. Cruzes and Tore Dybå. Recommended steps for thematic synthesis in software engineering. In *Proceedings of the 2011 International Symposium on Empirical Software Engineering and Measurement (ESEM 2011)*, pages 275–284. IEEE, 2011. doi:10.1109/ESEM.2011.36.

[19] David R. Thomas. A general inductive approach for analyzing qualitative evaluation data. *American Journal of Evaluation*, 27(2):237–246, 2006. doi:10.1177/1098214005283748.

[20] Cliodhna O'Connor and Helene Joffe. Intercoder reliability in qualitative research: Debates and practical guidelines. *International Journal of Qualitative Methods*, 19:1609406919899220, 2020. doi:10.1177/1609406919899220.

[21] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960. doi:10.1177/001316446002000104.

[22] Joseph L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971. doi:10.1037/h0031619.

[23] J. Richard Landis and Gary G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977. doi:10.2307/2529310.

[24] Daviti Gachechiladze, Filippo Lanubile, Nicole Novielli, and Alexander Serebrenik. Anger and its direction in collaborative software development. In *Proceedings of the 39th International Conference on Software Engineering: New Ideas and Emerging Technologies Results Track (ICSE-NIER 2017)*, pages 11–14. IEEE, 2017. doi:10.1109/ICSE-NIER.2017.18.

[25] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[26] Sida Wang and Christopher D. Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 90–94, 2012.

[27] Christian Janiesch, Patrick Zschech, and Kai Heinrich. Machine learning and deep learning. *Electronic Markets*, 31(3):685–695, April 2021. doi:10.1007/s12525-021-00475-2.

[28] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.

[29] Haibo He and Edwardo A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[30] Juri Opitz and Sebastian Burst. Macro F1 and macro F1. *arXiv preprint arXiv:1911.03347*, 2019.

[31] Sudhir Varma and Richard Simon. Bias in error estimation when using cross-validation for model selection. *BMC Bioinformatics*, 7(1):91, 2006. doi:10.1186/1471-2105-7-91.

[32] Remco R. Bouckaert and Eibe Frank. Evaluating the replicability of significance tests for comparing learning algorithms. In *Proceedings of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 3–12. Springer, 2004.

[33] Kevin A. Hallgren. Computing inter-rater reliability for observational data: An overview and tutorial. *Tutorials in Quantitative Methods for Psychology*, 8(1):23–34, 2012. doi:10.20982/tqmp.08.1.p023.

[34] Parastou Tourani, Bram Adams, and Alexander Serebrenik. Code of conduct in open source projects. In *Proceedings of the 2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 24–33. IEEE, 2017. doi:10.1109/SANER.2017.7884606.

A

Appendix

Table A.1: Sensitizing Dimensions, Anticipated Behaviors, and Confirmed Taxonomy Categories

Sensitizing Dimension	ACM Principle	Anticipated Behaviors	Confirmed Taxonomy Categories	Evolved Theme Name
Respect	1.4 Be fair and take action not to discriminate	Offensive or insulting language toward individuals; stereotyping or exclusionary language based on identity; harassment campaigns targeting specific contributors outside the repository	Personal Attacks and Demeaning Communication; Discrimination	Respect and Inclusion
Fairness	1.4 Be fair and take action not to discriminate	Biased or inconsistent decision-making; excluding contributors from discussion; restricting participation without justification; withholding project information; explicit vote manipulation or ballot stuffing in governance decisions	Governance Fairness; Lack of Transparency; Gatekeeping; Participation Shaming; Lack of Responsiveness[a]	Fairness in Participation and Decision-Making
Responsibility	1.2 Avoid harm	Deflecting blame for problems; misusing maintainer authority; releasing untested or vulnerable software; a maintainer explicitly abandoning a project without notifying dependent users; publicly attributing a security breach to a specific contributor by name	Responsibility Evasion; Power Abuse or Authority Dominance; Irresponsible Release; Irresponsible Disclosure[a]	Responsibility and Accountability
Privacy	1.6 Respect privacy	Unauthorized data collection or sharing; removing user control over settings or data; doxxing, defined as the public disclosure of a contributor's personal information without consent	Privacy Violation; User Autonomy and Control Violation	Privacy and User Autonomy
Honesty	1.3 Be honest and trustworthy	False claims about software capabilities or status; misrepresenting identity in project participation; fabricating contributor statistics or activity metrics	Misrepresentation; Impersonation	Honesty
Intellectual Property	1.5 Respect the work required to produce new ideas, inventions, creative works, and computing artifacts	Failing to credit contributors; using others' code without permission; violating software licenses	Attribution and Credit Issues; Intellectual Property Violation; Open Source Obligation Violation[a]	Ownership, Attribution and Obligation
Self-Interest	1.3 Be honest and trustworthy; 2.5 Give comprehensive and thorough evaluations	Undisclosed personal gain or bias in project decisions; accepting sponsorship or funding that creates undisclosed conflicts of interest in feature prioritization; steering project direction to benefit a contributor's employer without disclosure	—	—
Welfare	1.1 Contribute to society and human well-being	Negative impact of software on broader social well-being; concerns about software being used for surveillance or oppression of vulnerable populations	—	—
Sustainability	1.1 Contribute to society and human well-being	Concerns about software energy consumption or long-term environmental impact; concerns about excessive computational resource consumption in distributed systems; criticism of unnecessarily large dependencies that increase energy consumption	—	—

[a] Category emerged from data and was not anticipated prior to coding.
Dimensions shown in grey did not receive sufficient empirical support to warrant inclusion in the final taxonomy.

Table A.2: Anticipated Behaviors and Their Sources

Sensitizing Dimension	Anticipated Behaviors	Source
Respect	Offensive or insulting language toward individuals	Miller et al. [12]
	Stereotyping or exclusionary language based on identity	Miller et al. [12]; Sarker et al. [13]
	Harassment campaigns targeting specific contributors outside the repository	ACM Principle 1.4
Fairness	Biased or inconsistent decision-making; excluding contributors from discussion	Singh et al. [10]; Hsieh et al. [9]
	Restricting participation without justification	Hsieh et al. [9]
	Withholding project information	ACM Principle 1.4; Hsieh et al. [9]
	Explicit vote manipulation or ballot stuffing in governance decisions	ACM Principle 1.4
Responsibility	Deflecting blame for problems; misusing maintainer authority	ACM Principle 1.2
	Releasing untested or vulnerable software	ACM Principle 1.2
	A maintainer explicitly abandoning a project without notifying dependent users	ACM Principle 1.2
	Publicly attributing a security breach to a specific contributor by name	ACM Principle 1.2
Privacy	Unauthorized data collection or sharing	ACM Principle 1.6
	Removing user control over settings or data	ACM Principle 1.6
	Doxxing (public disclosure of a contributor's personal information without consent)	ACM Principle 1.6
Honesty	False claims about software capabilities or status	ACM Principle 1.3
	Misrepresenting identity in project participation	ACM Principle 1.3
	Fabricating contributor statistics or activity metrics	ACM Principle 1.3
Intellectual Property	Failing to credit contributors	ACM Principle 1.5; Win et al. [3]
	Using others' code without permission	ACM Principle 1.5; Win et al. [3]
	Violating software licenses	ACM Principle 1.5; Win et al. [3]
Self-Interest	Undisclosed personal gain or bias in project decisions	ACM Principles 1.3, 2.5
	Accepting sponsorship or funding that creates undisclosed conflicts of interest	ACM Principles 1.3, 2.5
	Steering project direction to benefit a contributor's employer without disclosure	ACM Principles 1.3, 2.5
Welfare	Negative impact of software on broader social well-being	ACM Principle 1.1
	Concerns about software being used for surveillance or oppression of vulnerable populations	ACM Principle 1.1
Sustainability	Concerns about software energy consumption or long-term environmental impact	ACM Principle 1.1
	Concerns about excessive computational resource consumption in distributed systems	ACM Principle 1.1
	Criticism of unnecessarily large dependencies that increase energy consumption	ACM Principle 1.1

Dimensions shown in grey did not receive sufficient empirical support to warrant inclusion in the final taxonomy.

Keywords Used for RQ2 Data Collection

The following table lists the keywords used to retrieve candidate artifacts from GitHub for each of the 18 ethical concern categories in the taxonomy. Keywords were constructed based on how these concerns are actually expressed in GitHub discussions, rather than through abstract synonym expansion using lexical tools.

Category	**Keywords**
Personal Attacks and Demeaning Communication	"you idiot", "you moron", "you are incompetent", "you are pathetic", "have no idea what you", "have no clue what you", "spews nonsense", "clearly don't understand", "you don't know what you're doing", "out of touch with reality", "whiny", "you are a troll", "fuck off", "fucking idiot"
Discrimination	"racist", "sexist", "transphobic", "homophobic", "ableist", "discriminatory", "discriminating against", "women can't", "females don't", "for a woman", "for a girl", "as a woman", "because you're a woman", "typical developer from", "doesn't belong here", "don't belong here", "not welcome here", "go back to", "your kind", "disabled people", "visually impaired", "disability", "accessibility", "chinese developers", "indian developers", "third world"
Governance Fairness	"without community input", "without consultation", "without discussion", "unilateral decision", "unilaterally", "not consulted", "excluded from", "shut out", "no say in", "ignored my", "overruled", "pushed through", "forced through", "cherry-picked", "rigged", "conflict of interest", "hidden agenda", "unfair decision", "unfairly", "biased decision", "dictator", "autocratic", "not democratic", "governance is broken", "governance failure"

Category	**Keywords**
Lack of Transparency	“behind closed doors”, “lack of transparency”, “not transparent”, “no transparency”, “without explanation”, “no explanation”, “without reason”, “no reason given”, “without justification”, “not documented”, “undocumented”, “kept secret”, “in secret”, “privately decided”, “private channel”, “without notice”, “without announcement”, “no changelog”, “no release notes”, “not communicated”, “silently changed”, “silently removed”, “silently deprecated”, “silently merged”, “hidden agenda”, “opaque process”, “kept in the dark”, “withheld”, “information withheld”
Lack of Responsiveness	“no response”, “never responded”, “never replied”, “not responded”, “ignored”, “being ignored”, “got no response”, “without response”, “pattern of ignoring”, “consistently ignored”, “repeatedly ignored”, “never acknowledged”, “unanswered”, “left unanswered”, “pr ignored”, “stale pr”, “stale issue”, “open for months”, “open for years”, “gathering dust”, “no feedback”, “no review”, “never reviewed”, “closed without explanation”, “closed without comment”, “dismissed”, “concerns ignored”, “maintainer unresponsive”, “radio silence”, “shouting into the void”, “talking to a wall”
Gatekeeping	“not welcome”, “unwelcome”, “don’t belong”, “not qualified”, “not experienced enough”, “not skilled enough”, “not good enough”, “gatekeeping”, “gatekeeper”, “go away”, “not your project”, “not your place”, “not your call”, “stay in your lane”, “outsider”, “you wouldn’t understand”, “none of your business”, “not your business”, “right to contribute”, “right to participate”, “earn your place”, “earn the right”, “prove yourself”, “not a real developer”, “just a user”, “just an outsider”, “no right to complain”, “no right to criticize”, “fork it”, “just fork”

Category	Keywords
Participation Shaming	“rtfm”, “read the docs”, “just google it”, “google is your friend”, “lmgtfy”, “stupid question”, “dumb question”, “obvious question”, “waste of time”, “wasting my time”, “wasting everyone’s time”, “stop asking”, “spam”, “noise”, “low quality”, “no effort”, “zero effort”, “try harder”, “did you even read”, “have you even tried”, “do your homework”, “do your research”, “learn to”, “not a support forum”, “not for beginners”, “closing as spam”, “closing as noise”, “not a real issue”, “wontfix”, “don’t bother”
Responsibility Evasion	“your fault”, “user error”, “you broke”, “your configuration”, “not our fault”, “not my fault”, “not our problem”, “not a bug”, “working as intended”, “by design”, “works on my machine”, “works for me”, “upstream issue”, “upstream bug”, “third-party issue”, “out of our control”, “not our responsibility”, “not my responsibility”, “PR welcome”, “feel free to fix”, “fix it yourself”, “not a security issue”, “won’t fix”, “security is your problem”, “user’s responsibility”
Power Abuse or Authority Dominance	“ban you”, “will ban”, “block you”, “revoke access”, “kicked out”, “refuse to merge”, “never merge”, “my project”, “my repo”, “final say”, “my rules”, “I decide”, “as the maintainer”, “as the owner”, “last warning”, “final warning”, “face consequences”, “not welcome here”, “leave this project”, “abusing your authority”, “abuse of power”, “power trip”, “acting like a dictator”, “toxic maintainer”, “bullying contributors”, “who do you think you are”, “you don’t get to”, “you have no right”
Irresponsible Release	“breaking change”, “breaking changes”, “released without testing”, “no testing”, “known bug”, “shipped with bugs”, “pushed to production”, “without adequate testing”, “without quality assurance”, “released prematurely”, “premature release”, “untested release”, “known vulnerability released”, “shipped vulnerable”

Category	**Keywords**
Irresponsible Disclosure	“responsible disclosure”, “irresponsible disclosure”, “coordinated disclosure”, “full disclosure”, “uncoordinated disclosure”, “disclosed before”, “disclosure policy”, “before fix”, “before patch”, “no patch available”, “without a fix”, “unpatched vulnerability”, “0-day”, “zero-day”, “proof of concept”, “published exploit”, “exploit code”, “publicly disclosed”, “premature disclosure”, “embargo”, “embargo period”, “90 days”, “private report”, “reported privately”, “security advisory”, “without notifying”, “without coordinating”
Privacy Violation	“without consent”, “without user consent”, “without user knowledge”, “secretly collecting”, “silently collecting”, “tracking users”, “covert tracking”, “hidden tracking”, “surveillance”, “fingerprinting”, “telemetry without”, “opt-out”, “selling user data”, “sharing user data”, “shared with third”, “privacy policy”, “violates privacy”, “privacy violation”, “privacy breach”, “data leak”, “data exposure”, “user data exposed”, “GDPR”, “CCPA”, “personally identifiable”, “PII”, “backdoor”, “spyware”, “plaintext password”, “unauthorized access”
User Autonomy and Control Violation	“forced update”, “force update”, “forced upgrade”, “auto update”, “no way to opt out”, “cannot opt out”, “removed the option”, “removed the ability”, “dropped support”, “overrides user”, “reset my settings”, “ignores user settings”, “cannot be disabled”, “no way to disable”, “cannot turn off”, “forced to”, “no choice”, “no way to prevent”, “without asking”, “takes away control”, “no control over”, “breaking change”, “no rollback”, “vendor lock-in”, “locked in”, “user autonomy”, “user control”, “against user”

Category	Keywords
Misrepresentation	“deliberately misleading”, “intentionally misleading”, “intentionally false”, “lied about”, “lying about”, “falsely claimed”, “false claim”, “advertised as”, “marketed as”, “promised feature”, “never delivered”, “overstated”, “exaggerated”, “misleading claims”, “misrepresented”, “misrepresentation”, “concealed”, “known limitation”, “hidden limitation”, “never mentioned”, “claimed actively maintained”, “production ready”, “false security”, “false sense of security”, “denied the bug”, “covered up”, “downplayed”, “misleading docs”
Impersonation	“impersonating”, “impersonation”, “pretending to be”, “posing as”, “masquerading as”, “fake account”, “fake profile”, “fake identity”, “sockpuppet”, “sock puppet”, “burner account”, “impersonating maintainer”, “fake maintainer”, “fake official”, “fake developer”, “typosquatting”, “name squatting”, “identity fraud”, “identity theft”, “social engineering”, “phishing”, “falsely claimed to be”, “multiple accounts”, “ban evasion”
Attribution and Credit Issues	“no attribution”, “without attribution”, “missing attribution”, “give credit”, “no credit”, “not credited”, “uncredited”, “took credit”, “stole credit”, “my work”, “original author”, “removed author”, “authorship”, “squashed”, “commit history”, “contributors list”, “AUTHORS file”, “CONTRIBUTORS”, “not mentioned”, “forgot to mention”, “plagiarism”, “plagiarized”, “copied without”, “used without credit”, “stole my”, “without giving credit”, “removed my name”, “original work”

Category	**Keywords**
Intellectual Property Violation	“license violation”, “violates license”, “license breach”, “license terms”, “copyright infringement”, “copyright violation”, “DMCA”, “takedown notice”, “intellectual property”, “IP violation”, “proprietary code”, “without permission”, “without a license”, “unlicensed”, “unauthorized use”, “stolen code”, “misappropriated”, “removed copyright”, “copyright notice”, “license header”, “missing license”, “license incompatible”, “conflicting license”, “relicense”, “CLA”, “contributor license agreement”, “redistribute”, “license compliance”
Open Source Obligation Violation	“must release source”, “release the source”, “open source the code”, “GPL obligation”, “copyleft obligation”, “GPL requires”, “AGPL requires”, “violates GPL”, “violates AGPL”, “copyleft violation”, “share-alike”, “viral license”, “must provide source”, “corresponding source”, “removed copyright notice”, “stripped copyright”, “must retain copyright”, “preserve copyright”, “must include license”, “license file missing”, “derivative work”, “modified version”, “changes must be”, “AGPL network”, “SaaS and GPL”, “not open sourced”, “should be open source”, “required to open source”, “open source obligation”, “license audit”