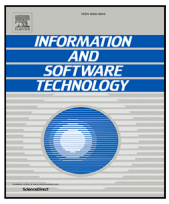
Information and Software Technology 191 (2026) 108003

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof



Emotional expression in open- source: How project function shapes communication

Matteo Vaccargiu [a,b], Silvia Bartolucci [c], Nicole Novielli [d], Marco Ortu [e], Roberto Tonelli [a], Giuseppe Destefanis [c,*]

[a] Department of Mathematics and Computer Science, University of Cagliari, Italy
[b] Department of Computer Science, Brunel University, United Kingdom
[c] Department of Computer Science, UCL, United Kingdom
[d] Dipartimento di Informatica, University of Bari "A. Moro", Italy
[e] Dipartimento di Scienze Economiche e Aziendali, University of Cagliari, Italy

ARTICLE INFO

Keywords:
OSS
Human aspects
Mining software repositories

ABSTRACT

Context: Open-source software (OSS) development is often studied as a decentralized process driven by technical goals. However, mature OSS projects operate under external constraints such as security advisories, release deadlines, and ecosystem dependencies. These pressures shape technical decisions and also communication patterns among contributors, including emotional expression.

Objective: This study investigates how emotional expression in OSS projects varies across different types of repositories, evolves over time, and relates to the activity of top contributors. The goal is to assess whether emotional dynamics are shaped more by project function than by technical domain or project size.

Methods: We analyzed issue comments from 14 OSS repositories spanning over ten years. A transformer-based classifier was used to detect emotions. Emotional patterns were quantified using a composite Emotional Index, and contextual activity. Contributor roles were assessed using a Contribution Index combining code activity, discussion engagement, and sustained involvement. Analyses were conducted at the repository, temporal, and contributor levels.

Results: The four most frequent emotions across all repositories were gratitude, curiosity, confusion, and approval. Emotional patterns tend to cluster by functional role rather than technical domain, with repositories converging toward stable emotional profiles over time. High-impact contributors show distinct expression patterns that reflect their role and stage of engagement.

Conclusion: Emotional expression in OSS projects follows recurring patterns linked to project function, contributor roles, and maturity. These findings can help anticipate communication challenges during project evolution and support interaction strategies among contributor groups with differing emotional tendencies.

1. Introduction

Open-source software (OSS) development is often portrayed as decentralized and loosely structured, with contributors working autonomously and guided primarily by technical interest [1]. This characterization fails to capture the conditions of many mature OSS projects that must respond to external constraints resembling those found in industrial settings. These projects operate under time-sensitive demands such as vulnerability disclosures [2,3], version deprecations [4–6], and dependency management across complex ecosystems [7]. Contributors are expected to coordinate quickly, resolve issues with minimal delay, and maintain project reliability while adapting to constant change.

Such constraints influence not only technical decision-making but also the way contributors interact. Communication becomes more deliberate, sometimes tense, and often shaped by the urgency or visibility of specific events [8–10]. While there is growing interest in the social and emotional aspects of software development [11–13], there is limited empirical evidence on how these dynamics manifest in open-source settings that operate under sustained external demands.

Existing research has examined emotional expression in software development, often focusing on specific projects, event-based snapshots, or limited timeframes. Prior work has explored connections between emotion and factors such as communication style [14–16], productivity [8], and conflict [17]. However, there is limited empirical evidence

* Corresponding author.
E-mail address: g.destefanis@ucl.ac.uk (G. Destefanis).

https://doi.org/10.1016/j.infsof.2025.108003
Received 10 July 2025; Received in revised form 15 December 2025; Accepted 16 December 2025
Available online 16 December 2025

on how emotional patterns (e.g., expressions of approval, confusion, curiosity, or gratitude) differ across project domains, how they evolve over time, or how they relate to the activity of contributors who drive project development.

These questions are especially relevant in projects subject to recurring external demands, where coordination and communication are shaped by urgency, technical risk, or stakeholder expectations. Examining emotional variation in such contexts may help clarify how contributors adapt their behavior as project conditions shift. This study examines how emotional expression varies across OSS projects that operate under sustained technical and organizational demands. It combines emotion detection on developer comments with a contribution index that captures the level and nature of participation over time.

The analysis spans 14 OSS repositories drawn from multiple functional categories, including developer tools (e.g., Hardhat), user-facing applications (e.g., MetaMask), core technical projects (e.g., Go-ethereum), scientific and infrastructure software (e.g., scikit-learn), and libraries or package managers (e.g., OpenZeppelin). It examines both cross-sectional and longitudinal variation, focusing on three levels: (I) project-level emotional characteristics, (II) temporal shifts in emotional expression, and (III) contributor-specific emotional profiles. By integrating these dimensions, we aim to provide a structured view of how emotional communication relates to project context and contributor role.

This study is guided by three research questions, each focusing on a different level of analysis.

RQ1: What are the predominant emotions in each repository? OSS projects vary in purpose and structure. Development of a security library may encourage formal, approval-driven communication, while the design and implementation of a developer tool may support more informal and exploratory exchanges. Identifying the dominant emotional characteristics in the contributors' communication traces of each repository helps explain how the project goals and context shape interaction style.

RQ2: How do emotions evolve over time within each repository? As repositories grow or respond to external demands, emotional expression may shift [18,19]. Tracking these changes reveals whether and how communication adapts during different phases, such as technical transitions or increased adoption.

RQ3: What is the emotional profile of the top contributors in each repository? A small number of contributors often drive both technical direction and communication norms. Analyzing their emotional profiles may enable highlighting how individual expression patterns relate to influence, consistency, and project cohesion, particularly across projects with different demands.

Together, these three research questions support the broader objective of the study: to provide a structured understanding of how emotional communication relates to both project context and contributor role. RQ1 focuses on repository-level emotional patterns, RQ2 captures their temporal evolution, and RQ3 connects emotional expression to developer participation. This layered approach allows us to examine emotional dynamics at multiple levels of granularity and interpret them in relation to project function, maturity, and contribution structure.

By examining emotional expression across multiple projects, time periods, and contributor roles, this study offers a structured view of how communication patterns vary in OSS development under sustained external constraints. The analysis spans 14 repositories across five functional categories (developer tools, user-facing applications, core technical components, scientific and infrastructure software, and libraries or package managers), with observation periods ranging from 6 to 15 years. We focus on the top 1% of contributors in each project, identified through a composite index combining code contributions, community engagement, and temporal involvement.

This study makes three main contributions.

First, it provides empirical evidence on how emotional patterns cluster by project function rather than technical domain, with repositories converging toward stable emotional profiles over time.

Second, it introduces an Emotional Index that enables systematic comparison of emotional communication at project, temporal, and contributor levels using a consistent and reproducible methodology, building on a previously validated Contribution Index (CI).

Third, it demonstrates that high-impact contributors exhibit distinct emotional expression patterns that reflect their role and stage of engagement, with implications for understanding community formation and project sustainability. Together, these contributions offer a methodology for understanding emotional communication in OSS development under sustained external and organizational constraints.

To ensure transparency and reproducibility, all datasets, analysis scripts, and results are provided in a replication package at this link.

2. Related work

Here we provide an overview of existing research on emotional dynamics in OSS development, tools for emotion analysis in software engineering, and contribution patterns in OSS projects, and clarify how we intend to advance the state of the art on these topics.

2.1. Emotional developers' communication

The study of emotions in software engineering has gained increasing attention since the mid-2010s, as researchers have examined the impact of emotional states on developers' productivity and well-being [12,20,21], as well as on team dynamics [22,23]. Early work [11] demonstrated that developers' happiness correlates with higher productivity and problem-solving abilities, while negative affects can impair cognitive processes necessary for programming tasks. Since then, the conceptualization of emotions in software engineering research has evolved from positive-negative sentiment analysis to more nuanced approaches that capture the multidimensional nature of emotional states, thus leading to more sophisticated approaches for studying emotional dynamics in development teams [13,24] viewing emotions as complex, dynamic events involving multiple aspects of human behavior [18,25–29].

The field has benefited from advances in affective computing, focusing on emotion detection in communication traces, understanding emotional impact on development processes [8,9], and providing emotion-aware recommendations [12,30]. Murgia et al. [10] conducted pioneering work analyzing emotions in issue tracking systems, finding that developers express a wide range of emotions during collaborative development, with emotional awareness proving crucial for team collaboration and project management. Building on this foundation, Ortu et al. [8] investigated how emotional expressions correlate with development metrics, finding that positive emotions are associated with faster issue resolution times.

While existing research has made progress in identifying emotional patterns in software development contexts, several important gaps remain. First, most studies examine emotions within a single project or limited time period, providing snapshots rather than longitudinal insights into emotional evolution [31,32]. Second, research has typically focused on either project-level emotional characteristics or individual emotional expressions, with limited integration across levels of analysis [31]. Third, there is a lack of comparative studies examining how emotional patterns vary across different types of OSS projects operating under diverse constraints [31]. Our work addresses these gaps by analyzing emotional patterns across multiple projects, time periods, and contributor roles, offering a structured view of how communication patterns vary in OSS development under sustained external constraints.

2.2. Emotion analysis in software engineering

Emotion analysis in software engineering has progressed from early lexicon-based methods using general-purpose dictionaries [33], which struggled with domain specific language [34,35], to specialized tools tailored for software contexts. Senti4SD [36], trained on Stack Overflow, improved accuracy over generic tools. SentiCR [37] and DEVA [38] were developed for code review and issue tracker comments, respectively. More recent approaches use deep learning and transformer models [18], including applications of large language models for sentiment classification [39]. Beyond sentiment polarity, Mäntylä et al. [9] introduced affective dimensions like arousal and dominance, linking emotional states to team productivity and burnout detection.

We extend this line of work by using a transformer-based model fine-tuned on the GoEmotions [40] dataset, supporting multi-label classification of 27 emotions. We apply our approach across a diverse set of repositories and integrate contributor metrics to relate emotion patterns to developer roles and project functions.

2.3. Contribution patterns in open-source software

Understanding who drives development in OSS projects is fundamental to analyzing collaborative dynamics.

In our paper [19], we introduced a contribution index that integrates three key dimensions of OSS participation: direct code contributions, community engagement, and temporal involvement. This index provides a balanced view of developer impact by considering not only technical metrics such as commits and pull requests (PR) but also social aspects like issue comments and the consistency of participation over time.

While prior work has established methods for identifying core contributors and understanding their technical impact [41–43], there has been limited investigation into how these influential developers shape the emotional climate of projects. Our current study builds upon this foundation by examining the relationship between contribution patterns and emotional expression. By analyzing the emotional profiles of top contributors identified through our validated contribution index, we provide novel insights into how those who drive technical development also influence communication norms and emotional dynamics within their communities. This approach bridges the gap between technical contribution analysis and the growing body of research on emotions in software development, offering a more holistic understanding of how OSS projects function under varying constraints.

2.4. Communication dynamics and emotions in teams

Team communication and emotional dynamics have been extensively investigated beyond software engineering in organizational psychology, social psychology, and communication studies. Research in organizational behavior has investigated the role and importance of emotional intelligence in team settings, finding that teams with high emotional intelligence experience increased trust, enhanced communication, and improved problem-solving capabilities [44,45]. Emotional contagion — the transfer of emotions between team members — has been demonstrated across various professional contexts, with studies showing that emotions can spread even through text-based communication in virtual environments [46,47]. Work engagement, defined as "a positive, fulfilling, work-related state of mind", was also shown to correlate positively with employee health outcomes [48] and innovativeness [49]. Recent research also investigated emotions in virtual teams, showing the importance of continuous support of team awareness, informal communication, and effective informal socio-emotional communication to enhance and facilitate virtual team collaboration [50].

3. Dataset and preprocessing

3.1. Dataset selection and adequacy

Our dataset comprises 14 open-source repositories chosen to capture emotional dynamics across collaborative development contexts. The selection followed a two-phase approach designed to balance ecosystem-specific insights with broader generalizability.

We distinguish between *technical function*, the main purpose a repository serves (e.g., developer tools, package management, user-facing applications), and *technical domain*, its underlying technological area (e.g., blockchain, cryptography, machine learning). Classification in this study is consistently function-based: repositories that belong to different domains but perform the same function (e.g., package management) are grouped together.

Domain entered our design only as a sampling dimension when selecting control projects outside the Ethereum ecosystem, in order to ensure variation across contrasting technological areas.

The initial ten repositories are drawn from the Ethereum blockchain ecosystem, as presented in a previous work of some of the authors [51]. We define this *single ecosystem* as a collection of interdependent projects that share (1) a common technological foundation (the Ethereum blockchain), (2) synchronized release cycles coordinated with network protocol upgrades, and (3) external pressures such as security incidents and market-driven deadlines. These repositories represent the ecosystem's backbone because they provide essential infrastructure on which the network depends. For example, Go-ethereum is the reference client used to validate and execute transactions on the Ethereum network, while Solidity is the language through which all smart contracts are written and deployed. Consensus-Specs documents protocol rules for Ethereum's consensus layer, MetaMask provides a widely adopted interface for end users to interact with decentralized applications, and OpenZeppelin maintains libraries that are the de facto standard for secure smart contract development. Developer-oriented tools such as Truffle, Hardhat, Web3.js, and Ethers.js enable application building and testing, while Chainlink contributes oracle functionality that connects smart contracts with external data sources. The complete dataset has been employed in previous research to examine how different types of external events impact developer activity patterns across individual repositories [51], while the Solidity repository data specifically has been analyzed to understand contribution patterns and emotional dynamics among top contributors in programming language development [19], and part of the Go-ethereum dataset was used to analyze sustainability topics in developer discussions [52]. In this study, we considered the complete dataset with the addition of emotions extracted from each comment (see 4.1.1) and contribution score for each developer as described in 4.1.3, extending the additions made for Solidity to the complete dataset.

To examine whether observed emotional patterns are specific to the Ethereum ecosystem or general across open-source collaboration, we extended the dataset with four projects from outside blockchain. These were sampled to provide contrasts in domain, governance, and community structure, while remaining classified by function. The inclusion criteria were: (1) sustained development activity over multiple years, (2) an active and diverse contributor base, and (3) visibility beyond a niche community, as indicated by widespread adoption or reliance by other projects.

Based on these criteria, we selected one project from four different domains: *scikit-learn* (scientific computing), *OpenSSL* (cryptographic library), *Node.js*(Java Script runtime), and *Brew* (package management). Each differs markedly from Ethereum repositories in purpose, governance, and user base. While Ethereum projects operate within a coordinated ecosystem with aligned development goals and interdependent release cycles, the non-Ethereum repositories follow independent governance structures, are maintained by distinct contributor communities, and serve broader or domain-specific audiences. For

instance, Brew and OpenSSL support general-purpose infrastructure, while scikit-learn targets scientific computing, and Node.js powers server-side JavaScript applications. These differences provide a clear contrast to the tightly coupled and domain-specific Ethereum ecosystem. They are also large and active enough to support meaningful comparison: for example, Node.js contains more than 18,000 issues, while scikit-learn and OpenSSL each exceed 10,000. This scale ensures that the control group provides sufficient statistical power for our analyses while remaining focused enough to allow project-level interpretation. The number of control projects corresponds to the four domains we aimed to represent, giving a total of four repositories.

This dataset is well-suited for studying emotional dynamics: (I) It provides **scale**, with more than 1.3 million comments across 85,000+ issues, allowing robust statistical analysis even when stratified by repository, time period, or contributor role. (II) It offers **temporal depth** spanning more than a decade (2010–2025), covering multiple phases of project evolution, including inception, growth, and maintenance. (III) It captures **contextual diversity**, with variation in project maturity, contributor base size, development velocity, and governance models. (IV) Some repositories operate under **external constraints** comparable to those in industrial software development, including urgent security vulnerabilities (e.g., OpenSSL's Heartbleed[1]), disruptive API changes, and time-sensitive protocol upgrades. These conditions provide authentic emotional responses that are not confined to experimental settings.

3.2. Technical function categorization

Repositories were assigned to functional categories using three criteria: (1) the project's stated purpose in its documentation, (2) its role in dependency chains (whether other projects build on or rely on it), and (3) its target audience (end users, developers, or infrastructure maintainers). Each repository was placed in the category that best reflects its main purpose, even if it also fulfills secondary roles:

- **Developer tools**: Truffle, Ethers.js, Hardhat, Web3.js. Frameworks and libraries that developers use to build and test applications.
- **User-facing applications**: Chainlink, MetaMask. Software that interacts directly with end users or external data sources, bridging technical systems with usability.
- **Core technical projects**: Solidity, Go-ethereum, Consensus-Specs. Components that define or implement the fundamental rules of the Ethereum network. Within this category, a *core client* refers specifically to a software implementation (e.g., Go-ethereum) that can independently run the Ethereum protocol, connect a node to the network, and validate transactions and blocks. Core clients are critical because the entire ecosystem depends on their correctness and stability.
- **Scientific and infrastructure projects**: OpenSSL, scikit-learn, Node.js. Large-scale projects that provide runtime environments or computational infrastructure.
- **Libraries and package managers**: OpenZeppelin, Brew. Systems for distributing, versioning, and managing reusable components.

Table 1 summarizes the repositories in our dataset. For each project we report its technical domain, the technical functional category, popularity as measured by GitHub stars, observation period (the first and last year of data available), and number of core contributors (developers accounting for 80% of activity). This information complements the functional categorization by situating each project within its broader technological context and highlighting variation in community size, adoption, and temporal coverage.

[1] https://www.heartbleed.com

3.3. Indicators for project maturity

The dataset includes repositories at different development stages, allowing us to identify whether emotional patterns are temporary features or persistent characteristics. We characterize projects using observable metrics derived from prior empirical work [42,53].

Development Longevity. Our observation periods span 6 to 15 years of project activity (Table 1). Projects like scikit-learn (observed since 2010) and OpenSSL (observed since 2013) have demonstrated sustained development well beyond the 7-year threshold where only 10% of projects typically survive [53].

Contributor Base Stability. We assess this through:

- *Core team size*: Number of developers accounting for 80% of contributions (ranging from 21 to 901)
- *Gini coefficient*: A standard metric used to assess contribution inequality, capturing how evenly work is distributed among contributors (ranging from 0.724 to 0.945). A higher Gini value (closer to 1) indicates strong concentration of activity in a small group, while a lower value reflects more balanced participation.

Issue Management. Projects show varying closure ratios (80.6% to 98.5%) and resolution times (378 to 9810 h), reflecting different maintenance practices and project contexts.

Based on these characteristics, we observe three groupings in Table 1:

- **Mature projects** (Node.js, scikit-learn, OpenSSL): Observed since 2010–2014, with 11–15 years of data showing high Gini coefficients (0.935–0.945) and concentrated core teams (25–100 developers)
- **Maturing projects** (Go-ethereum, MetaMask, Solidity, Brew): Observed since 2014–2016, with 8–10 years of data, Gini coefficients 0.808–0.921, varied core team sizes (33–901)
- **Growing projects**: Observed since 2014–2018, with 6–10 years of data, Gini coefficients 0.724–0.903, often requiring larger core teams (e.g., Web3.js: 862, Truffle: 510)

The key distinction appears in contribution patterns: mature projects show highly concentrated contributions despite thousands of total contributors, while growing projects often require hundreds of contributors to account for 80% of activity.

3.4. Rationale for analyzing issue comments

We focus specifically on issue comments as our primary data source for analyzing emotional dynamics. Issue comments provide a rich record of developer interaction, documenting the full cycle of collaborative problem-solving from problem reporting through discussion to resolution. They capture a wide range of emotional expressions, including frustration with bugs, gratitude for support, confusion about implementations, and approval of solutions.

Issue discussions involve a broader set of participants, including end users, newcomers, and maintainers. This inclusiveness allows us to observe emotional dynamics across different levels of engagement. Prior work has also shown that issue tracking systems are particularly expressive sites of emotion in software projects [8,10], supporting our choice of artifact.

3.5. Preprocessing

For all 14 repositories, we collected the GitHub issues and their associated comments, covering more than a decade of development history. We applied the following preprocessing steps:

Table 1
Repositories grouped by maturity, with functional classification and activity metrics. For each project we report its technical domain, main technical function category, popularity (GitHub stars), observation period (first and last year of data available), comments contributor counts, number of core contributors (developers accounting for 80% of activity), and along with, issue statistics, and Gini coefficients.

Repository	Tech. Domain	Tech. Function	Stars k	Obs. period	Contrib.	Core (80%)	Issues	Closed %	Res. hrs	Gini
MATURE PROJECTS										
Node.js	JavaScript runtime	Scientific/Infra.	113.0	2014 to 2025	16 826	100	18 864	91.3	4018	0.945
scikit-learn	Machine learning	Scientific/Infra.	63.3	2010 to 2025	9780	76	11 409	86.2	9810	0.935
OpenSSL	Cryptography TLS	Scientific/Infra.	28.5	2013 to 2025	4247	25	10 356	84.3	7231	0.939
MATURING PROJECTS										
Go-ethereum	Blockchain client	Core Technical	49.7	2014 to 2024	7321	901	8071	97.1	3319	0.808
MetaMask	Blockchain wallet	User App	12.7	2016 to 2024	6817	129	11 248	84.1	5741	0.883
Solidity	Smart contract lang	Core Technical	25.1	2015 to 2024	2150	33	5984	92.8	7037	0.918
Brew	Package manager	Libs Packages	44.8	2016 to 2025	4029	43	4655	98.5	378	0.921
GROWING PROJECTS										
Web3.js	Blockchain JS API	Dev Tool	19.8	2014 to 2024	3636	862	3888	95.4	3346	0.724
Truffle	Smart contract fwk	Dev Tool	14.0	2015 to 2024	3042	510	2926	82.6	7502	0.772
Ethers.js	Blockchain JS lib	Dev Tool	8.4	2016 to 2024	2292	447	2634	83.9	3666	0.762
Hardhat	Smart contract env	Dev Tool	8.1	2018 to 2024	2032	239	2548	80.6	5034	0.809
OpenZeppelin	Smart contract libs	Libs Packages	26.3	2016 to 2024	1444	128	1901	89.4	3854	0.836
Consensus-Specs	Blockchain cons.	Core Technical	3.8	2018 to 2024	359	23	920	84.0	5986	0.879
Chainlink	Oracle tools	User App	7.5	2017 to 2024	452	21	430	81.4	4716	0.903

Table 2
Overview of total and human comments and % of neutral human comments in the selected repositories.

Repository	Total comments	Human comments	Neutral (%)
Node.js	541,753	538,994	56.38
scikit-learn	304,828	301,408	46.64
MetaMask	81,957	77,910	65.96
OpenSSL	136,853	136,328	52.16
Go-ethereum	56,109	54,654	49.26
Solidity	47,560	46,213	48.67
Brew	85,570	83,864	44.17
Web3.js	19,521	17,303	48.81
Truffle	18,546	17,472	37.99
Ethers.js	13,961	13,901	35.16
Hardhat	14,487	14,274	49.35
OpenZeppelin	14,359	14,087	41.73
Consensus-Specs	8164	8139	47.79
Chainlink	12,968	12,051	72.31
Total	1,356,636	1,336,598	–

- **Bot filtering:** Contributions by automated accounts were excluded using an approach that extends the model introduced by Golzadeh et al. [54]. Our detection combines five signals: (1) explicit bot flags in metadata fields, (2) user type designations from GitHub's API, (3) keyword patterns in usernames (e.g., "-bot", "dependabot", "renovate"), (4) URL patterns indicating GitHub Apps, and (5) content-based detection in comment text. To minimize false positives, we implemented a whitelist mechanism for known human contributors whose names might trigger bot patterns, and required multiple check for borderline cases. Across repositories, automated comments ranged from 0.4% in Ethers.js to 11.1% in Web3.js, with the majority in the 2 to 5% band. After filtering, we retained over 1.2 million human-authored comments for analysis. In Table 2 we summarize the total comments and the number of human comments and % of neutral across the selected repositories.
- **Overlapping contributors:** To study cross–project participation, we considered issues and comments of the 14 projects. A developer was counted as participating in a repository if they either authored an issue or posted at least one comment. To avoid false merges across projects, identities were matched only when both the numeric GitHub `author_id` and the lowercased `author`/login string coincided. Before computing overlaps, we removed bot accounts using the procedure described in the point before. On the resulting set of unique human developers, we computed the distribution of participation across repositories. The distribution is highly skewed: **91.54%** of developers appear in only 1 repository; **5.85%** in 2; **1.48%** in 3; and **0.59%**, **0.29%**, **0.13%**, **0.07%**, and **0.04%** in 4, 5, 6, 7, and 8 repositories, respectively. In addition, we observe **10** developers active in 9 repositories and **3** in 10 repositories. Manual checks of the identities by the authors (which also helped verify bot exclusions) indicate that most multi-repository developers concentrate on the Ethereum-related projects, with occasional participation in one of the 4 added repositories. Overall, the prevailing pattern in our dataset is specialization: most developers focus their activity on a single project.
- **Timestamp normalization:** GitHub API provides timestamps in ISO 8601 format with varying timezone information. We standardized all timestamps to UTC using pandas' timezone-aware datetime parsing. For timestamps lacking explicit timezone information, we interpreted them as UTC to maintain consistency. Normalization was needed for calculating accurate issue resolution times across repositories with global contributors. A cutoff date of August 28, 2024, was used for open issues to compute their current duration.
- **Text preservation:** We retained all original comment content including code snippets, URLs, and markdown formatting to preserve context for emotion detection. Code elements appear in 0.4% to 4.3% of comments (mean: 1.8%), while URLs are present in 8.2% to 57% of comments. The preservation approach aligns with the RoBERTa-GoEmotions model's training on mixed technical and natural language content.

Each comment was further annotated with emotion labels and confidence scores using RoBERTa-GoEmotions, described in Section 4.1.1.

4. Methodology

Fig. 1 illustrates the empirical protocol followed in this study. The workflow begins with data collection from GitHub repositories, followed by emotion detection on issue comments, and the computation of a Contribution Index to quantify developer activity. These components form the basis for the subsequent analyses addressing our three research questions. Each step is described in detail in the following subsections.

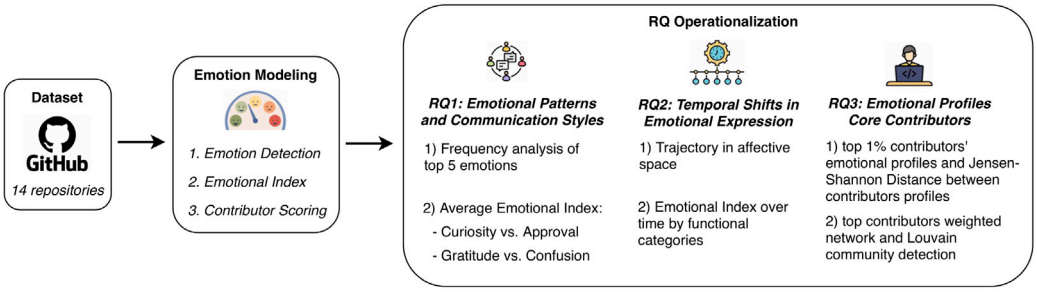


Fig. 1. Description of the empirical protocol.

4.1. Analysis methods

4.1.1. Emotion detection

To analyze emotional content in developer communications, we used the *RoBERTa-base-GoEmotions*[2] model, a transformer-based classifier fine-tuned on the GoEmotions dataset [40]. The model supports multi-label classification and identifies 27 distinct emotions, covering a broad range of affective states relevant to developer interactions, such as technical confusion, collaborative approval, or expressions of gratitude.

The GoEmotions dataset consists of 58k English-language Reddit comments collected across a wide variety of subreddits. While the dataset was filtered to include only English text, it does not distinguish between comments written by native and non-native English speakers. As a result, the training data reflects the linguistic diversity of Reddit's user base, with both fluent native usage and contributions from second-language English speakers.

Input/Output Process: The emotion detection takes as input the raw text of each comment and outputs a probability distribution over 27 emotion categories, with confidence scores ranging from 0 to 1. We retain all emotion scores regardless of their magnitude, preserving the full probability distribution for each comment. This allows capturing in detail the emotional expressions and maintains the model's complete output.

Interpretation: The emotion scores represent the model's confidence that a particular emotion is present in the text. A score of 0.7 for gratitude indicates 70% confidence that gratitude is expressed. When aggregated at the repository level, these scores are summed to create emotion frequencies, which represent the cumulative emotional expression across all comments. For example, a repository-level gratitude frequency of 31,101 indicates the sum of all gratitude confidence scores across the repository's comments. The aggregation approach preserves weak emotional signals that might be meaningful when accumulated across many comments.

This model was chosen based on prior validation in software development settings [19,55], where it showed strong performance. Its multi-label capability is important for capturing overlapping emotions often found in developer comments (e.g., simultaneous appreciation and concern). These annotations underpin all downstream analyses (repository trends, temporal shifts, and contributor profiles). While the model was originally pretrained and evaluated on a 27-class emotion detection task, prior studies have mainly considered coarse-grained categories such as joy, anger, sadness, and fear. Performance for some fine-grained categories remains limited due to class imbalance in the training data. To mitigate this, we aggregate the model's emotion predictions (i.e., the probability scores it assigns to each category) at the comment and repository level, emphasizing the most frequent and reliable emotions. This reduces noise from individual misclassifications.

[2] https://huggingface.co/SamLowe/roberta-base-go_emotions

4.1.2. Emotional index

To quantify emotional dynamics in software development, we introduce a composite Emotional Index (EI) that integrates emotion frequency, contextual intensity, and interaction-level adjustments. The formulation draws on prior work in sentiment and emotion analysis within software engineering [56–58], and is designed to reflect both the direct expression of emotions and the situational conditions under which they occur.

The EI positions entities (repositories or contributors) in a two-dimensional affective space. It is constructed from two components: (1) context-specific emotion weights, and (2) pairwise emotion comparisons along opposing affective axes.

Context weights. For each emotion e, we compute a context-aware weight combining issue resolution time and emotion activity level:

$$w_e = \text{resolution\_factor}_e \times \text{activity\_factor}_e \tag{1}$$

Resolution time for an issue i is computed as:

$$\text{resol\_time}_i = \begin{cases} \text{closed\_at} - \text{created\_at} & \text{if closed} \\ \text{cutoff\_date} - \text{created\_at} & \text{if open} \end{cases} \tag{2}$$

where *cutoff_date* is defined as the last observed activity in the repository.

The resolution factor for each emotion e is defined as:

$$\text{resolution\_factor}_e = 1 + \log\left(\frac{\text{median\_resolution\_time}_e}{\text{issue\_resolution\_time}_e}\right) \tag{3}$$

where:

- $\textit{median\_resolution\_time}_e$ is the median resolution time across issues containing emotion e
- $\textit{issue\_resolution\_time}_e$ is defined per issue as above[3]

The activity factor normalizes emotion frequency:

$$\text{activity\_factor}_e = \frac{\text{emotion\_frequency}_e}{\text{max\_emotion\_frequency}_e} \tag{4}$$

where $\text{emotion\_frequency}_e$ represents the sum of emotion e scores across all comments within a repository, and $\text{max\_emotion\_frequency}_e$ is the highest such sum for emotion e across all repositories in our dataset. For example, if gratitude has frequencies of 31,101 in Node.js, 28,774 in scikit-learn, and lower values in other repositories, then $\text{max\_emotion\_frequency}_{\text{gratitude}} = 31{,}101$. This normalization ensures comparability across repositories of different sizes and activity levels, with activity_factor_e ranging from 0 to 1.

This construction ensures that the EI reflects not only how often an emotion is expressed but also the resolution context in which it occurs.

[3] The resolution factor employs a logarithmic transformation to compress the wide range of time-based values and reduce sensitivity to outliers, as issue resolution times can vary from hours to months.

Emotional index calculation. For any opposing emotion pair (e_1, e_2), we compute the relative EI as:

$$\mathrm{EI}(e_1, e_2) = \frac{(f_{e_1} \cdot w_{e_1}) - (f_{e_2} \cdot w_{e_2})}{(f_{e_1} \cdot w_{e_1}) + (f_{e_2} \cdot w_{e_2})} \tag{5}$$

where:

- f_{e_1}, f_{e_2} are the observed frequencies (repository-level sums) of emotions e_1 and e_2
- w_{e_1}, w_{e_2} are the context weights computed as above

EI yields a relative score between −1 and +1, indicating the dominance of one emotion over its counterpart. The normalization to [−1, 1] ensures comparability across repositories regardless of their absolute comment volumes or emotion frequencies. All emotional indices are computed within each repository first, then compared across repositories to identify patterns.

Worked example. Assume that for one repository in a specific month, the emotion pair is (gratitude, anger). Let $f_{\text{gratitude}} = 200$ and $f_{\text{anger}} = 100$, and suppose $w_{\text{gratitude}} = 1.2$ and $w_{\text{anger}} = 1.0$. Then:

$$\mathrm{EI}(\text{gratitude}, \text{anger}) = \frac{(200 \cdot 1.2) - (100 \cdot 1.0)}{(200 \cdot 1.2) + (100 \cdot 1.0)} = \frac{240 - 100}{240 + 100} = \frac{140}{340} \approx 0.41$$

The resulting EI value indicates a strong net balance in favor of gratitude in that time window.

Affective coordinates. Each entity is placed in a two-dimensional space:

$$\begin{aligned} x &= \mathrm{EI}(e_1, e_2) \\ y &= \mathrm{EI}(e_3, e_4) \end{aligned} \tag{6}$$

The resulting coordinates reflect the balance between opposing affective poles, with all values normalized across projects and time windows.

Emotion selection and pairing. The choice of the four focal emotions, gratitude, approval, confusion, and curiosity, was driven by empirical prominence and relevance to collaborative development rather than strict alignment with canonical emotion taxonomies. These emotions consistently appeared with high frequency across all repositories, accounting for a substantial portion of the detected emotional content. The pairing strategy, gratitude vs. confusion and approval vs. curiosity, reflects contrasting communicative functions that are recurrent in developer interactions.

Gratitude and approval represent positive evaluative signals commonly expressed in response to contributions, issue resolutions, or design discussions. Curiosity and confusion capture cognitive or exploratory states typically associated with knowledge gaps, uncertainty, or requests for clarification. While we acknowledge that these categories do not correspond to classical 'basic' emotions as defined in psychological theories such as Ekman's or Plutchik's models, our intent is not to map directly onto affective universals. Instead, we adopt a functional lens grounded in the communicative context of OSS development.

These pairings are not meant to exhaust the emotional space, but to serve as interpretable axes for observing how constructive engagement (approval, gratitude) and epistemic tension (curiosity, confusion) vary across roles, time, and project structure. Their consistent detection and polarity across repositories make them suitable candidates for comparative modeling. Further exploration of other emotion pairs remains possible using the same methodological framework.

Application scope. In our study, we applied the EI at three levels. At the **repository level**, we aggregated all comments within a project to examine overall emotional characteristics (RQ1). At the **temporal level**, we computed the index over yearly time windows to capture changes in emotional expression over time (RQ2). At the **contributor level**, we calculated the index using comments authored by individual developers, focusing on the top 1% of contributors in each project (RQ3).

Generalizability. The EI is not intended as a universal ground-truth model of emotion, but as a comparative metric tailored to this dataset. While the design is guided by established principles and prior studies, the weight formulations (e.g., logarithmic compression, frequency normalization) are optimized for the type of data available in GitHub issues. Applying this metric to other settings (e.g., chat logs, mailing lists) may require adapting the weighting scheme or redefining the context variables.

4.1.3. Contributor scoring

To identify and compare developer involvement across projects, we applied the contribution index proposed in our previous work (Vaccargiu et al. [19]). This index integrates three dimensions of open-source participation: direct code contributions (commits and pull requests), community engagement (comments and issues), and temporal involvement (activity frequency and duration). Each dimension is weighted to reflect its relative importance in sustaining project development. Code-related activities receive the highest weight (45%), followed by community engagement (30%) and temporal metrics (25%). This weighting scheme captures the technical and social aspects of contribution within open-source ecosystems.

The three dimensions of the contribution index are designed to capture complementary aspects of developer activity. Direct code contributions (e.g., commits, pull requests) and community engagement (e.g., issue comments, issue openings) represent distinct modes of participation, with the former reflecting technical authorship and the latter capturing collaborative interaction. These two categories are treated as non-overlapping in the computation. Temporal involvement, by contrast, is orthogonal to the other two dimensions: it reflects the consistency and duration of participation rather than its type. This dimension captures sustained engagement over time, regardless of whether a contributor is more active in code development or community discussion. The inclusion of temporal involvement ensures that short bursts of intense activity are weighted differently from long-term participation. All three components are normalized and then combined using a weighted aggregation: 45% for code contributions, 30% for community engagement, and 25% for temporal involvement, following the procedure established in prior work [19].

Scores are normalized to a 0–100 scale within each repository, enabling identification of the top 1% of contributors based on relative contribution levels within each project, regardless of project size.

Table 3 summarizes the normalization approaches used across all analyses to ensure comparability within and across repositories.

This normalization process ensures fair comparisons: (1) within-repository normalization (contributor scores, Jensen–Shannon Distance (JSD)) identifies relative patterns regardless of repository size, (2) cross-repository normalization (activity factor) enables comparison across projects with different activity levels, and (3) the [−1, 1] EI scale provides a standardized metric for all emotional comparisons. When comparing across repositories, we first compute metrics within each repository, then compare the normalized values, preventing larger repositories from dominating the analysis.

4.2. RQ operationalization

4.2.1. RQ1: Emotion distribution

To analyze predominant emotional patterns, we first identified the five most frequent emotions in each repository. We then focused on the four most common emotions across all projects, *approval*, *confusion*, *curiosity*, and *gratitude*, to enable consistent cross-repository comparison.

Although the emotion detection model supports 27 categories, our empirical analysis showed that four emotions — gratitude, curiosity, confusion, and approval — consistently appeared among the top five most frequent in all repositories. The focus on these four emotions was not a selection imposed a priori but rather an empirical outcome

Table 3
Normalization methods for cross-repository comparison.

Metric	Normalization	Scope
Activity Factor	$\frac{f_c}{max(f_c)}$ across repos	Cross-repository comparison
EI	[−1, 1] via $(f_1 w_1 - f_2 w_2)/(f_1 w_1 + f_2 w_2)$	Computed per repository, then compared
Contributor Score	0-100 percentile ranking	Within repository (top 1% = 99-100)
JSD	0-1 distance metric	Within repository top contributors
Temporal EI	Same as EI, per year	Within repository by year

that emerged during repository-level aggregation. The remaining 23 emotions were not uniformly distributed across projects; many appeared only sporadically, with low frequency and without consistent presence across repositories or functional categories. This concentration around four emotions suggests that these categories represent stable and recurring aspects of developer communication, while the others may reflect more isolated or context-specific expressions. We therefore retained the four dominant emotions as the basis for cross-project comparison to ensure consistency, comparability, and analytical focus. The fact that such a small subset captures the majority of emotional expression across diverse projects is itself a significant observation and is discussed further in Section 4.

Emotions were quantified using the EI defined in Section 4.1.2, which maps each repository to a two-dimensional affective space. The EI accounts for both emotion frequency and contextual weights based on resolution time and activity. We conducted comparative analysis of repositories' positions in this space, identifying five categories based on functional roles rather than technical domains.

To strengthen the interpretation of the visual patterns in the affective space, we applied statistical tests with two explicit objectives. First, we examined whether repositories within the same functional category show comparable variance in their EI coordinates. This evaluates whether projects that serve the same function display consistent spread in their positions in the affective space. Second, we tested whether the mean positions of the functional categories, represented by their centroids in the affective space, differ from one another. This evaluates whether the apparent separation between categories reflects systematic differences in emotional expression rather than random variation.

For our inferential statistical analysis, we examined the distribution of EI values across functional categories by testing the following hypotheses. For within-cluster homogeneity, we tested: $\boldsymbol{H_0}$: *Projects belonging to the same functional category have equal variance in their EI values (across EI_x, EI_y, and distance to the category centroid*[4]*)*, with $\boldsymbol{H_A}$ stating that at least one project departs from this homogeneity. For between-cluster separation, we tested: $\boldsymbol{H_0}$: *Repositories are exchangeable across functional categories in the affective space*, with $\boldsymbol{H_A}$ stating that functional categories differ in their centroids.

We first checked normality of EI_x and EI_y using Shapiro–Wilk tests [59], as well as skewness and kurtosis, which indicated no strong deviations from normality. However, given the small number of projects per category (2–4) and the risk of low statistical power, we employed nonparametric and robust alternatives. For within-cluster homogeneity, we used the Brown–Forsythe test [60], a median-based variant of Levene's test that is robust to deviations from normality and focuses on equality of dispersion. For between-cluster separation, we applied permutation-based MANOVA[5] (PERMANOVA) [61] on (EI_x, EI_y), which does not assume multivariate normality and is suitable for small samples. Since the Brown–Forsythe analyses involved three related hypotheses (on EI_x, EI_y, and distance to centroid), we controlled the False Discovery Rate (FDR) using the Benjamini–Hochberg procedure [62] at α=.05. The global PERMANOVA constitutes a single test and is reported without multiplicity adjustment. For exploratory pairwise PERMANOVA contrasts between categories, we applied FDR across the set of pairwise p-values, consistent with recommendations for balancing Type I and II errors [63].

4.2.2. RQ2: Temporal variation

To analyze how emotions change over time within each repository, we tracked the variation in emotional profiles across multiple years of development. We focused on the four predominant emotions identified in RQ1 and applied the EI to plot the temporal progression of each repository's emotional profile. For each repository, we segmented comments by year based on their creation timestamp, calculated the EI for each year, plotted the repositories in the two-dimensional affective space defined by approval–curiosity (x-axis) and gratitude–confusion (y-axis), and connected points chronologically with directional arrows to visualize the path of change. Temporal analysis employs annual aggregation to smooth short-term fluctuations while preserving long-term trends. The EI values, already normalized to [−1, 1] through the relative difference formula, enable direct comparison of emotional positions across years and repositories. To improve interpretability, we organized repositories into functional categories as established in RQ1, visualizing emotional trajectories independently for each category and highlighting the first and last years to emphasize the overall direction of change. The temporal aggregation approach allowed us to identify patterns specific to different repository types, including convergence toward characteristic emotional profiles, cyclical variations, and transition points.

4.2.3. RQ3: Contributor profiles

To examine emotional variation among key contributors, we conducted a three-phase analysis. First, we identified the top 1% of contributors in each repository using a composite score that combines code contributions, community engagement, and temporal activity [19]. Second, we assessed emotional consistency by computing a probability distribution over the 27 detected emotions per contributor, then measuring pairwise JSD and summarizing the distribution within each repository. Lower JSD values indicate greater similarity. Third, we built weighted networks where nodes represent contributors and edges reflect emotional similarity. Community detection was performed on these networks, where nodes represent individual developers and edges represent emotional similarity (inverse of JSD). Communities were identified using the Louvain algorithm, which clusters developers based on their emotional expression patterns. Community size refers to the number of developers within each detected cluster, not to be confused with repository-level metrics such as issue count or total contributors. We calculated cohesion ratios between and within communities, and analyzed each group's emotional profile using the EI from RQ1.

4.2.4. Ethical note

This study analyzes only publicly available data from GitHub repositories. All emotional analysis is conducted at the level of issues or repositories, and no attempt is made to profile or interpret individual developer behavior. The study reports aggregate trends and avoids identifying or targeting contributors. No sensitive or private data is involved.

[4] A centroid is the arithmetic mean position of all points in a set. In this context, it represents the average EI (EI_x, EI_y) coordinates of the repositories within a functional cluster, serving as its central reference point in the affective space.

[5] Multivariate Analysis of Variance (MANOVA) is a statistical method used to test whether the mean vectors of two or more groups differ on a set of outcomes simultaneously.

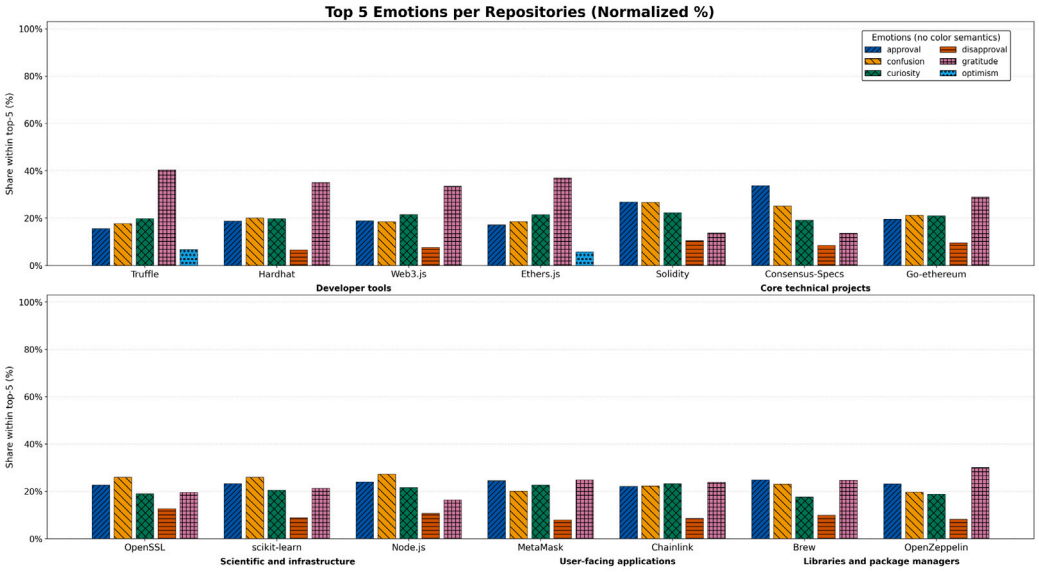


Fig. 2. Top-5 emotions per repository organized according to the functional categories shown in Table 4 (normalized shares, 0%–100%).

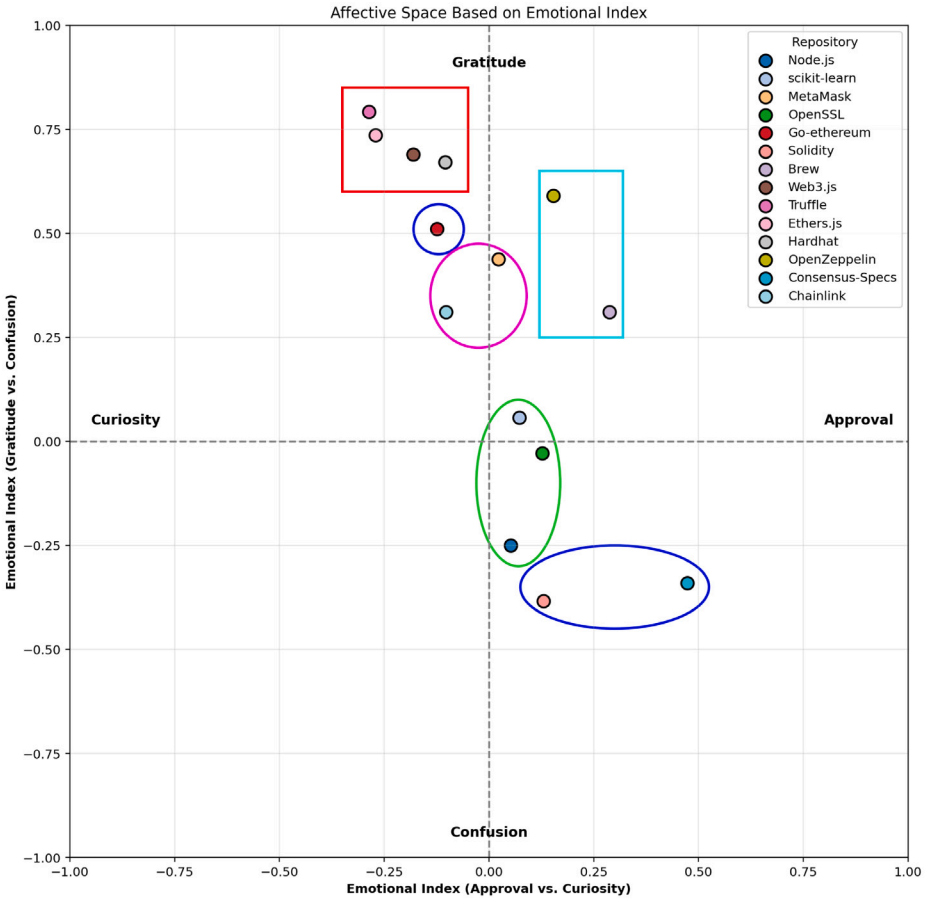


Fig. 3. Average EI across the lifecycle of each repository.

5. Emotional patterns and communication styles across open-source projects

The analysis of emotional content across the 14 repositories reveals four emotions that consistently appear among the most frequent: **gratitude (pink), curiosity (green), confusion (yellow)**, and **approval (blue)** (see Fig. 2). These emotions recur across projects regardless of technical domain, while the fifth most common emotion varies between optimism and disapproval.

To better understand how these emotions manifest in different project contexts, we classified the repositories based on their primary function (Table 4).

Table 4
Classification of repositories by functional category.

Functional category	Repositories
Developer tools	Truffle, Hardhat, Web3.js, Ethers.js
Core technical projects	Solidity, Consensus-Specs, Go-ethereum
Scientific and infrastructure	OpenSSL, scikit-learn, Node.js
User-facing applications	MetaMask, Chainlink
Libraries and package managers	Brew, OpenZeppelin

The functional categorization was introduced in Section 3.2. Here, we use it to assess whether projects with similar roles show comparable emotional patterns. The consistency observed in the EI analysis supports this grouping as a useful way of linking project function to communication style.

Emotional frequency varies by function. Core technical projects show distinct patterns (Fig. 2). Go-ethereum displays a balanced mix of gratitude, confusion, approval, and curiosity, unlike Node.js or scikit-learn, where specific emotions dominate. This balance sets Go-ethereum apart from peers like Solidity and Consensus-Specs, indicating that emotional expression can vary even within the same functional category. Developer tools (Truffle, Hardhat, Web3.js, Ethers.js) show a stronger presence of gratitude compared to other categories, with discussions in these repositories more often including appreciation among contributors. The fifth most frequent emotion varies across repositories, with optimism appearing only in Truffle and Ethers.js, whilst disapproval is more common elsewhere. Mature projects with large user bases (Node.js, scikit-learn, OpenSSL) exhibit distinctive emotional patterns, with confusion emerging as the predominant emotion by a substantial margin, particularly in Node.js and scikit-learn. In our analysis, we applied the EI defined in Section 4.1.2, pairing emotions as curiosity vs. approval and gratitude vs. confusion, representing two contrasting forces in the development process. The first captures the balance between exploration (asking questions, proposing new solutions) and validation (confirming correct implementations, building consensus), while the second reflects knowledge transfer effectiveness, with gratitude signaling successful resolution of challenges and confusion indicating unresolved obstacles.

The EI visualization in Fig. 3 positions each repository in a two-dimensional space. Several distinct groupings emerge:

Developer tools cluster (upper left quadrant - red): Truffle, Hardhat, Web3.js, and Ethers.js show high gratitude and curiosity values, indicating exploratory discussions combined with frequent expressions of appreciation.

Core technical projects cluster (lower right and center - blue): Consensus-Specs and Solidity display higher approval values with moderate confusion, while Go-ethereum shows higher gratitude alongside confusion. These projects are positioned toward the approval–confusion axis.

Scientific and infrastructure projects (near origin - green): OpenSSL, scikit-learn, and Node.js exhibit balanced emotional profiles with a slight tendency toward approval-confusion.

User-facing applications (center-left - magenta): MetaMask and Chainlink balance curiosity with moderate gratitude, showing a mixture of exploratory and appreciative interactions.

Package managers (right-center - cyan): Brew and OpenZeppelin occupy positions with moderate approval and medium-to-high gratitude values.

The data demonstrates that in general repositories with similar functional roles occupy similar regions in the affective space, even when they serve different technological domains. For example, Brew and OpenZeppelin occupy comparable positions despite serving entirely different domains (traditional package management versus blockchain smart contracts), indicating similar communication patterns in package management contexts. Go-ethereum is the only project that differs from its functional cluster (Solidity and Consensus), showing a moderate tendency toward gratitude and curiosity unlike the other projects that tend toward approval and confusion.

We statistically assessed the visual grouping in the Affective Space with two nonparametric tests. First, Brown–Forsythe tests across functional clusters on EI_x, EI_y, and distance to each cluster centroid indicated comparable within-cluster dispersion (after FDR correction, $p = .3732$, $p = .6796$, $p = .3732$). Second, a permutation MANOVA (PERMANOVA) on (EI_x, EI_y) showed significant overall separation among clusters ($F = 5.314$, $p = .0134$). While pairwise contrasts did not remain significant after FDR adjustment, descriptive centroid distances (ranging from 0.0760 to 0.8750) highlight a spread between clusters. Taken together, these results provide statistical support for the five functional groupings shown in Fig. 3, though the small sample size within each category (2–4 projects) limits pairwise resolution.

The recurrence of four core emotions (gratitude, curiosity, confusion, and approval) across all repositories suggests they serve key communicative roles. Gratitude supports community cohesion, curiosity fosters problem-solving, confusion signals gaps in understanding, and approval helps build consensus. These functions appear consistent across domains.

Differences in emotional frequency reflect each category's communication needs: Developer tools encourage exploration, core infrastructure focuses on technical validation, and user-facing projects blend technical detail with user support.

Answer to RQ1: Four emotions (gratitude, curiosity, confusion, and approval) predominate across all repositories, with the EI revealing clustering by functional role rather than technological domain. Developer tools show high gratitude and curiosity; Core technical projects favor approval and confusion (with Go-ethereum as an exception); Scientific and infrastructure projects display confusion as dominant; and User-facing applications blend curiosity with gratitude. Statistical analysis (Brown–Forsythe and PERMANOVA) supports the presence of these functional groupings, although the small number of projects per cluster limits pairwise resolution. This demonstrates that project function shapes emotional communication patterns more strongly than technical content.

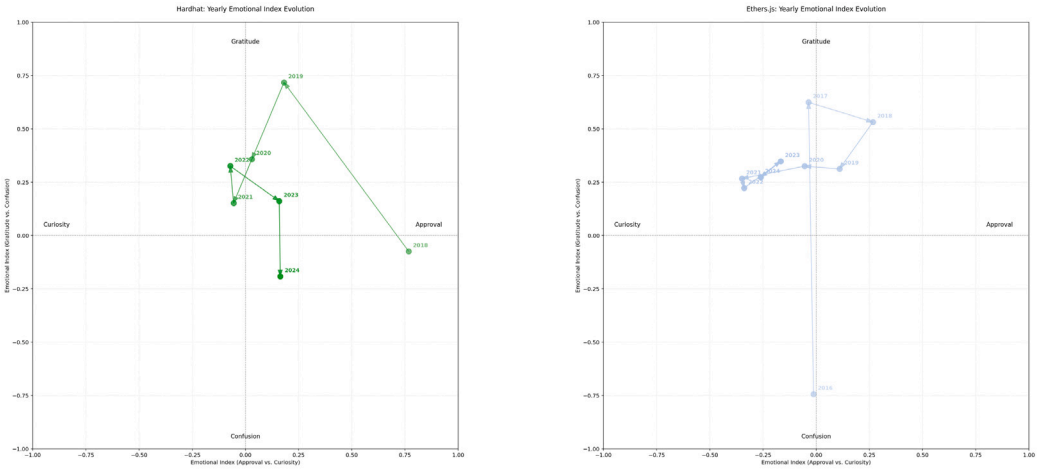
6. Temporal shifts in emotional expression during project development

Our temporal analysis of emotional indices highlights how emotional profiles change both within and across the repository categories defined in Table 4. Figs. 4–7 show how repositories move in affective space over time, with some converging toward category norms and others displaying volatility during development transitions.

Developer tools (Truffle, Hardhat, Web3.js, and Ethers.js), shown in Fig. 4, display early volatility in their emotional profiles before stabilizing over time. Ethers.js and Web3.js start at emotional extremes of confusion and curiosity, but gradually converge toward the gratitude–curiosity quadrant noted in RQ1. Hardhat begins near approval and shifts toward the center. Truffle starts centrally, moves toward gratitude–curiosity, and more recently leans toward approval.

User-facing applications (MetaMask and Chainlink), shown in Fig. 5, display cyclical emotional trajectories with noticeable shifts between affective states. MetaMask moves from confusion–approval toward gratitude. Chainlink shows a more volatile path, shifting between confusion–approval and gratitude before stabilizing near confusion.

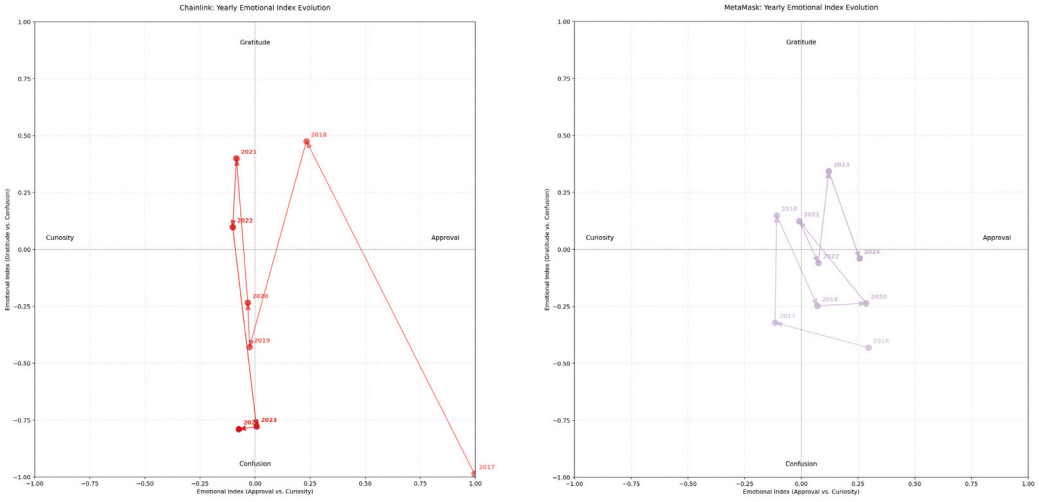
Core technical projects (Consensus-Specs, Solidity, and Go-ethereum), shown in Fig. 6, follow relatively stable emotional paths over time. Consensus-Specs moves between confusion–approval and gratitude before settling near the center of the affective space. Solidity shifts from gratitude to confusion and back. Go-ethereum transitions from approval to confusion.



(a) Hardhat **(b)** Ethers.js

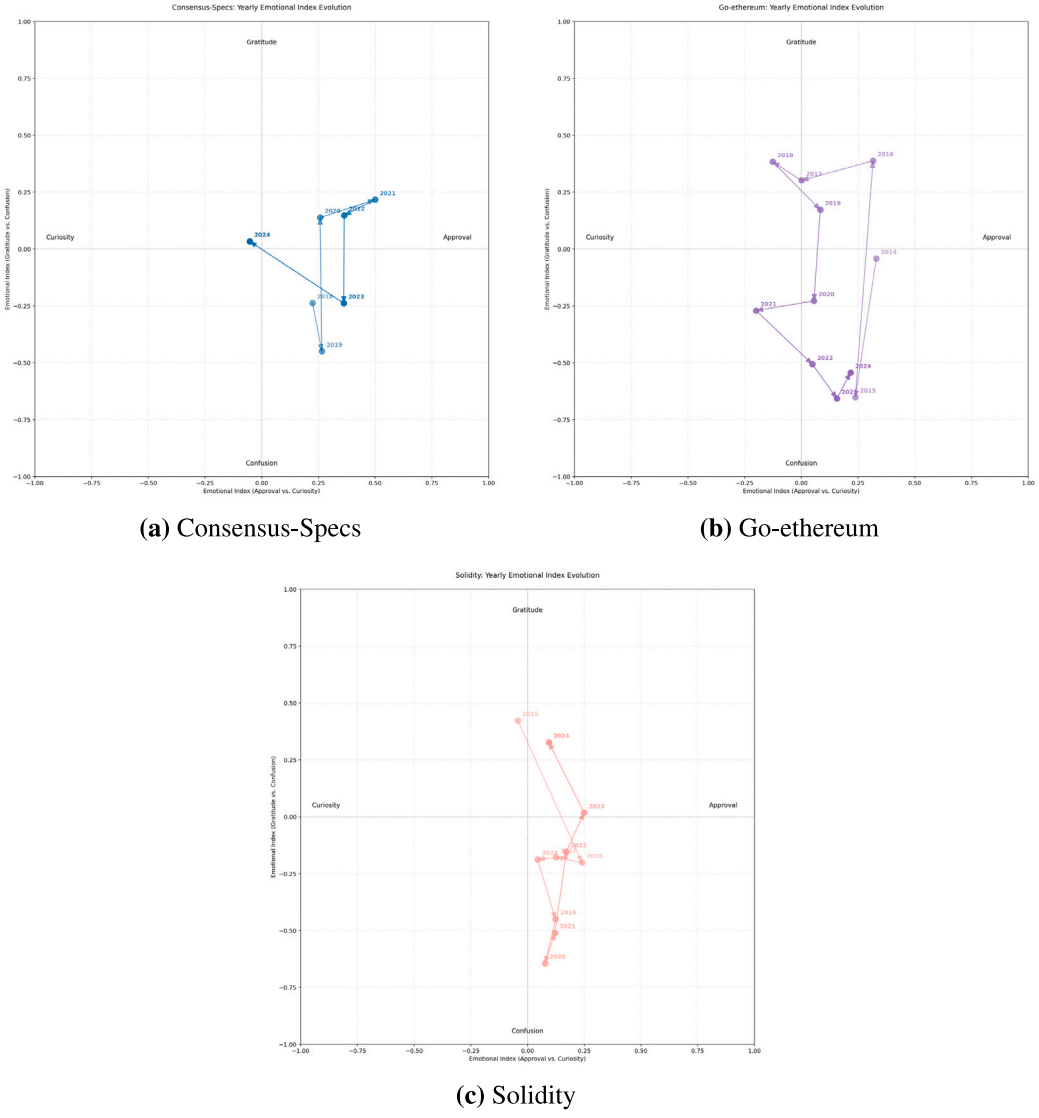
(c) Truffle **(d)** Web3.js

Fig. 4. Yearly EI evolution for **Developer tools**. Points are annual EI values (Approval vs. Curiosity on x; Gratitude vs. Confusion on y); arrows link consecutive years. The coloring follows a light-to-dark progression over time.



(a) Chainlink **(b)** MetaMask

Fig. 5. Yearly EI evolution for **User-facing applications**. Points are annual EI values (Approval vs. Curiosity on x; Gratitude vs. Confusion on y); arrows link consecutive years. The coloring follows a light-to-dark progression over time.



(a) Consensus-Specs **(b)** Go-ethereum

(c) Solidity

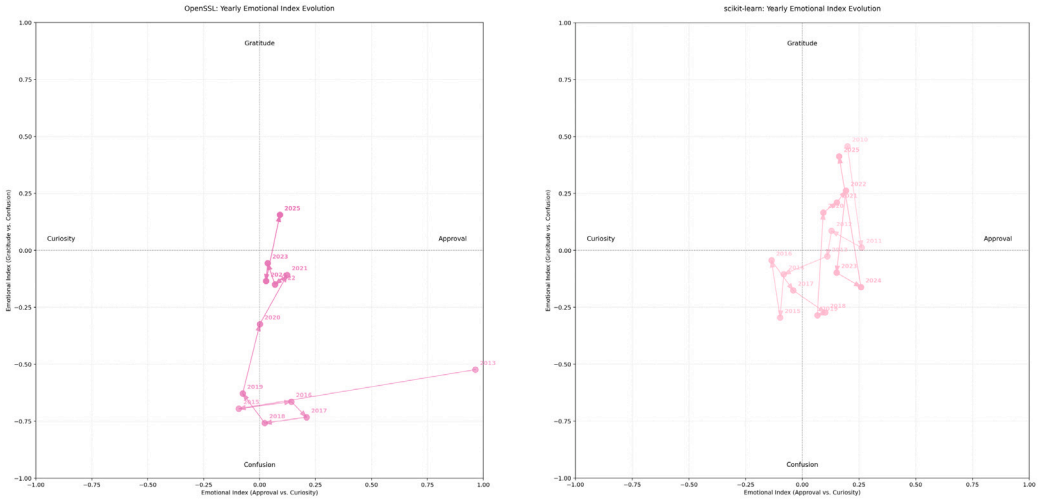
Fig. 6. Yearly EI evolution for **Core technical projects**. Points are annual EI values (Approval vs. Curiosity on x; Gratitude vs. Confusion on y); arrows link consecutive years. The coloring follows a light-to-dark progression over time.

Scientific and infrastructure projects (OpenSSL, scikit-learn, and Node.js), shown in Fig. 7, display varied but generally stable emotional trajectories. OpenSSL starts with dominant approval and confusion, gradually shifts toward curiosity, and later shows increased gratitude, eventually stabilizing near the category average. Scikit-learn begins near gratitude, moves toward confusion, and returns. Node.js shows little emotional variation.

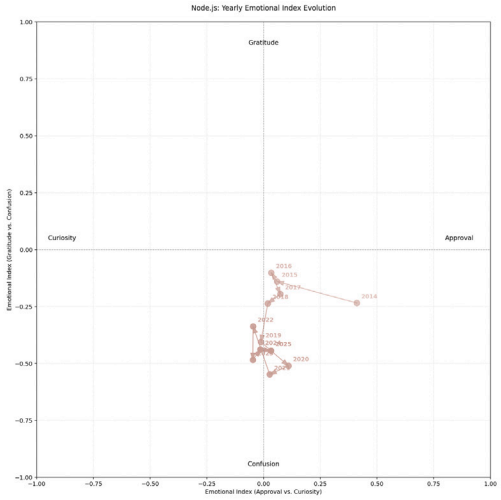
Dependency and library managers (OpenZeppelin and Brew), shown in Fig. 8, display stable profiles with occasional fluctuations. OpenZeppelin stays near its average EI for several years before shifting toward confusion and later approval. Brew begins with modest approval, passes through curiosity, gratitude, and confusion, before returning to its initial profile.

These temporal patterns are illustrated by specific trajectories. **Chainlink**, for example, moved from EI coordinates of (0.996, −0.985) in 2017 to (−0.073, −0.789) in 2024, representing a transition from strong approval and confusion to a more neutral but still confusion-leaning emotional tone. This path includes a swing toward gratitude in 2018, followed by sharp volatility and reversion. In contrast, **scikit-learn** exhibited cyclical fluctuation: it began with moderate gratitude and approval in 2010 (0.199, 0.457), shifted through confusion in 2015 and back toward positivity, reaching (0.162, 0.413) in 2025. These oscillations suggest a project responsive to both internal dynamics and external demand. Meanwhile, **Node.js** demonstrated overall emotional stability despite minor fluctuations: its EI values remained within the range of −0.05 to 0.41 on the x-axis and −0.55 to −0.1 on the y-axis between 2014 and 2025. This bounded variation is consistent with a mature, well-established infrastructure project maintaining a consistent tone in community interactions. Similarly, **Web3.js** showed pronounced emotional fluctuation, starting at (−0.699, −0.699) in 2014 — strong confusion and curiosity — then moving across the space to end at (0.073, 0.829) in 2024, indicating a significant shift toward gratitude and emotional engagement.

Across all categories, many repositories converge toward stable emotional profiles over time, while others show cyclical changes that gradually stabilize or sharp shifts at particular stages. Older projects, especially those launched before 2015, display greater emotional stability, indicating reduced volatility with maturity.



(a) OpenSSL **(b)** scikit-learn



(c) Node.js

Fig. 7. Yearly EI evolution for **Scientific and infrastructure projects** . Points are annual EI values (Approval vs. Curiosity on x; Gratitude vs. Confusion on y); arrows link consecutive years. The coloring follows a light-to-dark progression over time.

Answer to RQ2: Emotional expression changes over time in ways that reflect repository function and maturity. Developer tools shift from confusion to more stable profiles; User-facing applications show cyclical variation with higher volatility; Core technical projects remain stable; Scientific and infrastructure repositories shift gradually across states; and dependency managers tend to return to earlier emotional patterns. Overall, repositories converge toward their category's characteristic emotional profile as they mature.

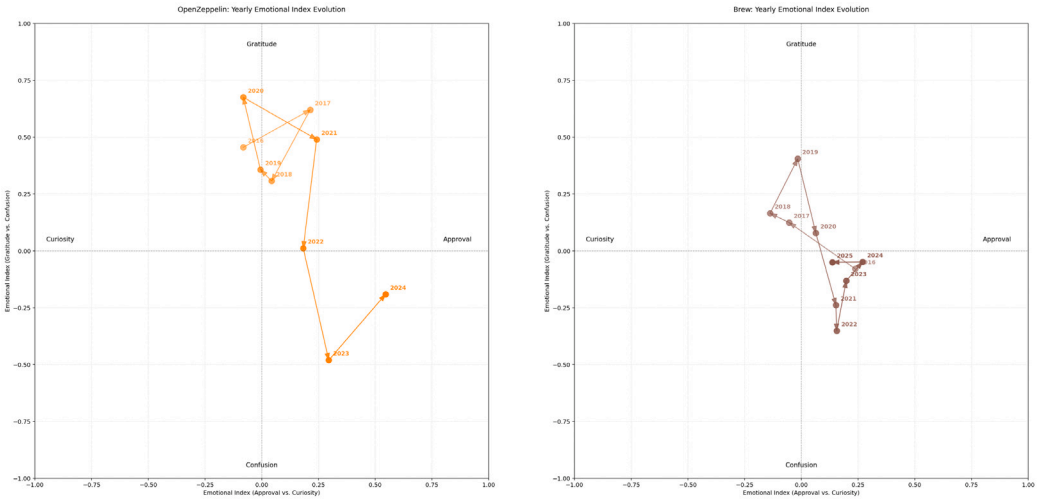
7. Consistency and variation in the emotional profiles of core contributors

To examine emotional variation among top contributors, we analyzed the emotional profiles of the top 1% contributors in each repository, identified using the contribution index. We focused on the top 1% of contributors for three reasons. First, this threshold captures developers who account for 79.6% of total contribution across all 14 repositories, indicating sustained high-impact participation. Second, the emotional profiles of the top 1% are nearly identical to those of the broader top 20%, with a global JSD of 0.015 and repository-level values ranging from 0.01 to 0.045 (median: 0.024). This similarity demonstrates that the top 1% captures the dominant emotional patterns without requiring analysis of larger contributor groups. Third, the 1% threshold provides sufficient sample sizes for network community detection while maintaining a consistent percentile-based criterion across all projects for direct comparison.

For each developer, we constructed a probability distribution over the 27 detected emotions and computed the JSD between each pair of contributors within the same repository. This metric quantifies the similarity of emotional expression, with lower values indicating more similar profiles. The results show clear differences in emotional consistency across repositories. Table 5 reports the minimum, maximum, mean, median, and standard deviation of JSD values between the top 1% in each repository.

Larger, more mature repositories such as scikit-learn, Node.js, Brew, and Ope- nSSL show very low JSD values among their top contributors, indicating high emotional consistency. This aligns with the stable emotional trajectories observed in Figs. 7, and 8, where these repositories remain close to a fixed position in the affective space.

In contrast, Developer tools such as Hardhat, Ethers.js, Web3.js, and Truffle exhibit the highest average JSD values, suggesting greater diversity in contributor emotional profiles. This pattern is consistent



(a) OpenZeppelin **(b)** Brew

Fig. 8. Yearly EI evolution for **Dependency and Library Managers**. Points are annual EI values (Approval vs. Curiosity on x; Gratitude vs. Confusion on y); arrows link consecutive years. The coloring follows a light-to-dark progression over time.

Table 5
JSD statistics and top 1% contributors per project.

Project	Top 1%	Min	Max	Mean	Median	Std
Hardhat	26	.0822	.5645	.2929	.2906	.1146
Truffle	36	.0781	.7549	.2690	.2375	.1217
Web3.js	45	.0174	.5944	.2672	.2455	.1074
Go-ethereum	94	.0294	.7752	.2576	.2260	.1308
Ethers.js	29	.1082	.4488	.2451	.2403	.0599
OpenZeppelin	19	.0735	.5402	.2357	.2032	.1052
MetaMask	82	.0282	.6471	.2215	.1987	.0965
Chainlink	6	.0717	.3743	.2190	.2130	.1032
Solidity	30	.0477	.5069	.1897	.1580	.0939
Consensus-Specs	5	.0746	.1843	.1146	.1080	.0343
scikit-learn	123	<.0001	.0005	.0001	<.0001	.0001
Node.js	213	<.0001	<.0001	<.0001	<.0001	<.0001
Brew	57	<.0001	<.0001	<.0001	<.0001	<.0001
OpenSSL	77	<.0001	<.0001	<.0001	<.0001	<.0001

Note: Values rounded to four decimals, sorted in descending order by mean.

with the emotional volatility shown in Fig. 4. Go-ethereum, although a Core technical project, shows the highest overall JSD and notable standard deviation, possibly due to its architectural complexity and long-term evolution, which attract contributors with differing emotional styles. Smaller repositories like Chainlink and Consensus-Specs also show relatively high standard deviations, likely due to their limited contributor base, where even minor shifts in participation can affect overall emotional consistency. To explore whether developers within projects express a common emotion or whether they can be identified in communities that display the same emotional tone, we constructed networks among top contributors using JSD-weighted edges. Table 6 reports the structural characteristics of these networks.

In Hardhat, Ethers.js, Truffle, Web3.js, Go-ethereum, and MetaMask, cohesion ratios range from 1.0878 to 1.1418, meaning contributors within each group are only 8%–14% more similar to each other than to those in other groups, indicating modest separation. Modularity values remain low (around 10^{-2}), suggesting these divisions are only slightly more structured than random splits. By contrast, Solidity, OpenZeppelin, Brew, Node.js, OpenSSL, and scikit-learn form single communities, with no meaningful internal division. Chainlink and Consensus-Specs also appear as single communities; however, with only 5 nodes, these networks are too small for reliable community detection, and their classification should be interpreted with caution as the limited number of nodes inherently constrains potential community structures. We identify the two communities using the labels "Community 0/1", which are arbitrary and only indicate distinct clusters.

Among multi-community projects, Go-ethereum shows the clearest split: Community 0 is more active across all metrics and began contributing 483 days earlier, indicating a distinction between early and later contributors. In MetaMask, differences are smaller; Community 0 is slightly more active (by overall contribution score) while Community 1 leads in several activity counts, and Community 1 started 294 days earlier, but both groups remain engaged. Hardhat shows a larger disparity: Community 1 contributes far more (often over 150%) and Community 0 started 374 days earlier, with Community 1 dominating development. Truffle shows the opposite: Community 0 is more active across most metrics, while Community 1 started 257 days earlier, suggesting an earlier cohort that is now less central. In Web3.js, Community 1 overtakes Community 0 in all key activity metrics and joined 686 days later, marking a clear generational change. Ethers.js shows less contrast: Community 1 is more active overall and started 211 days earlier, while Community 0 leads (slightly) only in merged pull requests. Building on this structural analysis, we calculated the EI for each community, as shown in Fig. 9.

In Go-ethereum, both communities express high gratitude, but differ in secondary emotions: Community 0 leans toward curiosity, while Community 1 is more approval-oriented. A similar pattern appears in MetaMask, where Community 0 leans toward curiosity and Community 1 is positioned between confusion and approval. In Hardhat, both communities gravitate toward gratitude, though Community 1 also shows curiosity, and Community 0 leans slightly toward approval.

Table 6
Network analysis summary.

Project	Nodes	Edges	Comm.	Modul.	Avg Within	Avg Between	Cohes. Ratio
Hardhat	24	276	2	.0103	.7513	.6580	1.1418
Ethers.js	29	406	2	.0031	.7911	.7192	1.1000
Truffle	35	595	2	.0104	.7701	.6942	1.1093
Web3.js	44	946	2	.0179	.7776	.6899	1.1271
Go-ethereum	94	4371	2	.0151	.7721	.7097	1.0878
MetaMask	80	3160	2	.0123	.8129	.7401	1.0984
Solidity	27	351	1	<.0001	.8103	.0000	–
OpenZeppelin	19	171	1	<.0001	.7643	.0000	–
Chainlink	5	10	1	<.0001	.7810	.0000	–
Consensus-Specs	5	10	1	<.0001	.8854	.0000	–
Brew	55	1485	1	<.0001	.9999	.0000	–
Node.js	211	22 155	1	<.0001	.9999	.0000	–
OpenSSL	75	2775	1	<.0001	.9999	.0000	–
scikit-learn	122	7381	1	<.0001	.9999	.0000	–

Note: Projects with 2 communities sorted by descending cohesion ratio, followed by projects with 1 community.

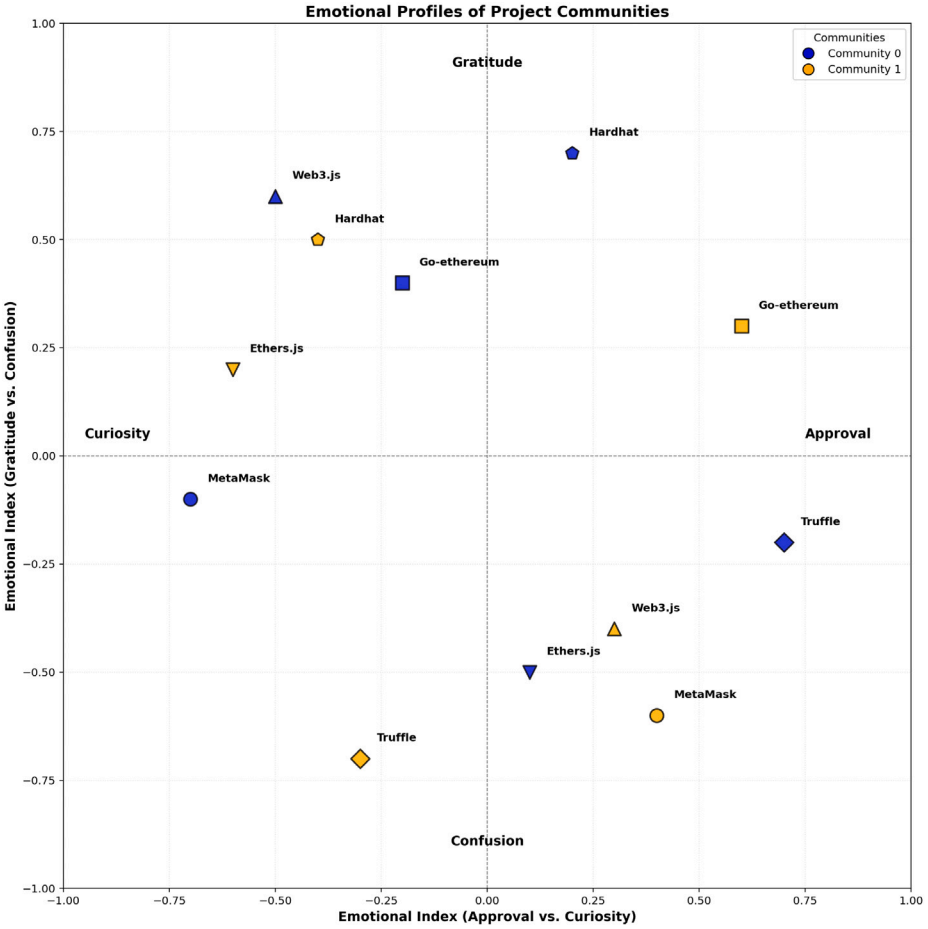


Fig. 9. Emotional profiles of project communities.

Truffle displays stronger divergence: Community 0 favors approval, while Community 1 lies between confusion and curiosity, indicating active engagement but also some uncertainty. The most pronounced contrasts appear in Web3.js and Ethers.js. In both, one community (Web3.js's Community 0 and Ethers.js's Community 1) is located in the curiosity–gratitude quadrant, while the other falls closer to approval and confusion. These patterns are reflected in the radar plots. For example, Fig. 10(a) shows that Go-ethereum's larger community exhibits strong gratitude, whereas the smaller group displays a more balanced profile leaning toward confusion and approval.

The emotional profiles of the two communities in Hardhat are nearly inverse: Community 0 shows dominant approval with higher confusion than curiosity, while Community 1 exhibits peak expression in gratitude with balanced curiosity and confusion, creating complementary communication patterns within the project.

Table 7 summarizes the emotional profiles and activity distributions for projects with two communities. Several patterns emerge. First, in most projects, one community (typically C1) shows higher gratitude whilst the other leans toward approval or confusion. Second, activity patterns differ between communities: in MetaMask, Web3.js, and

Table 7
Community emotional profiles and activity distribution in projects with two communities.

Project	Comm.	Dominant emotion	Commits (%)	PRs (%)	Comments (%)	Issues (%)
MetaMask	C0	Approval	5.3	4.2	87.3	3.2
	C1	Gratitude	29.7	16.3	44.6	9.4
Go-ethereum	C0	Approval & Confusion	25.7	10.1	61.8	2.3
	C1	Gratitude	9.8	10.2	72.8	7.1
Web3.js	C0	Approval	15.3	2.5	75.6	6.7
	C1	Gratitude	22.0	11.7	57.7	8.6
Truffle	C0	Gratitude & Curiosity	48.9	5.8	43.0	2.4
	C1	Approval & Confusion	49.0	10.4	36.2	4.4
Ethers.js	C0	Confusion	0.0	0.4	86.0	13.6
	C1	Gratitude	8.4	0.0	88.1	3.5
Hardhat	C0	Approval & Confusion	15.4	3.4	80.3	0.9
	C1	Gratitude	50.5	9.9	35.1	4.5

Hardhat, the gratitude-oriented community allocates more effort to commits and PRs, whilst the counterpart focuses on comments. Go-ethereum shows the opposite pattern, with both communities comment-heavy but C0 contributing more commits. Truffle presents balanced activity between communities, with both showing roughly equal commit shares but different emotional orientations. Ethers.js is exceptional, with both communities overwhelmingly focused on comments (86%–88%) and minimal PR activity, suggesting a discussion-centric development model.

Overall, the emotional profiles suggest distinct patterns of involvement across communities. In most repositories, the larger community tends to express more gratitude, reflecting mutual appreciation among contributors. Smaller communities are often more emotionally balanced or inclined toward approval or confusion, which may indicate uncertainty or a stronger need for validation. Within projects that split into two affective communities, one community tends to be discussion-centric (comments-heavy), while the other is code-centric (higher internal share of commits/PRs). This pattern appears clearly in MetaMask, Web3.js, and Hardhat, and is weaker or mixed in Go-ethereum, Truffle, and Ethers.js.

Answer to RQ3: Emotional profiles among top contributors vary across repositories, shaped by project type and maturity. Established projects such as *scikit-learn*, *Node.js*, *Brew*, and *OpenSSL* show high emotional consistency (low JSD), while Developer tools (*Hardhat*, *Ethers.js*, *Web3.js*, *Truffle*) exhibit greater divergence. Six repositories feature two moderately distinct contributor communities (cohesion ratios 1.0878–1.1418); the rest form a single community. Larger communities tend to express more *gratitude* and *curiosity*, while smaller ones lean toward *approval* or *confusion*, reflecting different modes of engagement. In several projects, the gratitude-leaning community allocates a larger internal share to commits and merged PRs, while the counterpart is comments-heavy; issues remain a minor share.

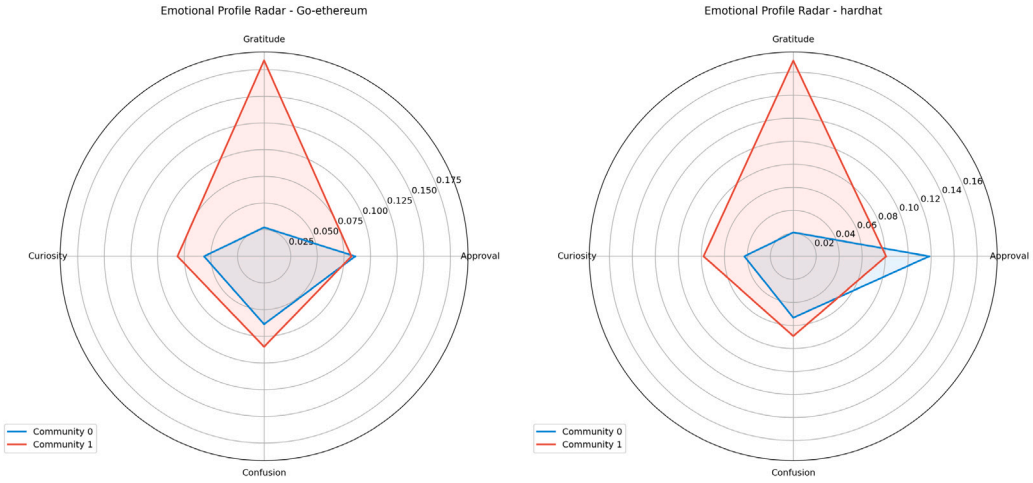
8. Discussion

The systematic emotional patterns identified across 14 repositories can inform project management, adoption decisions, and individual contribution strategies. Our temporal analysis shows that Developer tools shift from confusion-dominated states (EI values between −0.5 and −0.75) toward gratitude–curiosity equilibria (EI values between 0.5 and 0.75). This 1.0–1.5 point change on our EI scale typically occurs over 3–5 years. This pattern echoes findings by Murgia et al. [10], who showed that emotions are dynamically expressed throughout software collaboration. However, while much prior work has focused on static or short-term emotional snapshots [31], our results highlight longitudinal trends in how communication tone evolves as projects mature. During early confusion phases, targeted support strategies — such as enhancing onboarding or clarifying documentation — may be especially impactful, consistent with prior observations that positive affect improves problem-solving and engagement [11].

Functional clustering suggests possible benchmarks for monitoring project health. Developer tools in our dataset generally moved from confusion-dominated communication toward more gratitude–curiosity oriented profiles as they matured. Projects that remain in strongly confusion-oriented states after several years may therefore benefit from interventions such as simplifying APIs, strengthening documentation, or improving community support. Contributor community analysis provides additional perspective. In several repositories, including Go-ethereum and MetaMask, we observed two distinct contributor groups with partially different emotional tendencies. For example, in Go-ethereum, an earlier group expressed more gratitude and curiosity, while a later group leaned toward approval and confusion. Both groups remain active, indicating that different communication styles can coexist within the same project rather than converging to a single mode. These patterns support prior studies emphasizing the social diversity of contributors' communication styles [8], and add a new perspective by showing how emotional variance can persist even within tightly integrated ecosystems. Multi-level emotional analysis of contributors — from top developers to emerging participants — remains an underexplored area in prior work.

Our findings also have implications for project evaluation and adoption. Differences in JSD values across repositories illustrate how emotional expression can vary with project maturity. For instance, projects such as Node.js, Brew, and OpenSSL show highly consistent emotional profiles among their top contributors, while projects like Hardhat and Ethers.js exhibit greater variation. This contrast reflects earlier work by Mäntylä et al. [9], which linked emotional consistency to team maturity and productivity. Our cross-project comparison provides a complementary view by connecting these traits to project age and community development stage. Higher variation suggests that contributor communication styles are still in flux, whereas lower variation points to more consistent patterns of interaction among core members. Temporal trajectories provide a complementary signal: Chainlink, for example, covers a longer path through emotional space than Consensus-Specs despite similar ages, suggesting greater volatility in communication. Functional categorization also shapes these patterns. OpenZeppelin and Brew, although serving different technical purposes, occupy nearby positions in emotional space. This proximity indicates that adoption decisions may be informed not only by technical ecosystem but also by functional role, since projects with similar functions can share comparable communication dynamics.

Implications extend to individual developers as well. Temporal analysis shows that contributors who join during early confusion phases face steeper learning curves but can establish themselves as core members, as seen in the 226–815 day gaps between communities in multi-community projects. Later joiners encounter more stable environments



(a) Go-ethereum **(b)** Hardhat

Fig. 10. Emotional Profile Radar charts comparison.

but may face established hierarchies. Community-specific profiles also matter: larger groups, which account for about two-thirds of top contributors, consistently show higher gratitude scores (EI differences of 0.3–0.5), while smaller groups lean toward approval–confusion. New contributors may therefore find more receptive environments in larger, gratitude-oriented communities, whereas those with specialized expertise may contribute more effectively in groups where approval and confusion dominate communication. The relationship between contribution activity and emotional expression reinforces that performance is not measured solely by code volume. In Go-ethereum, one community contributed 150% more commits than the other, yet their emotional profiles were comparable. This suggests that alignment in communication style is as important as productivity for integration into core teams. Contributors can assess their fit by comparing their emotional expression to established norms in the project.

Our findings also align with broader research on team communication and emotional dynamics in organizational psychology and studies of hybrid collaboration, which emphasize the roles of emotional intelligence and emotional contagion in shaping group effectiveness [44–46]. Prior studies have shown that emotions can propagate even in asynchronous text-based environments [46], reinforcing the relevance of emotion-sensitive analysis for OSS communication, as also explored in [12,27]. This parallel suggests that OSS development does not exhibit unique patterns of emotional behavior but reflects more general characteristics of human collaboration. Such convergence indicates that established theories of team emotions and work engagement can be applied in OSS contexts, opening the possibility of adapting intervention strategies developed in organizational and hybrid settings. In particular, approaches that take into account emotional expression and emotional intelligence in intra-team communication may also be relevant in supporting open-source collaboration.

There are also broader implications for open-source sustainability. Gratitude emerged as one of the most frequent emotions among top contributors across all repositories, indicating that expressions of appreciation are a recurring feature in sustaining voluntary participation. Projects with consistently lower levels of gratitude may find it more difficult to retain contributors, as indicated by the comparison between longer-lived projects and the overall distribution of emotions. Functional clustering also challenges the assumption that technical ecosystems alone determine community similarity. For example, Go-ethereum appears more distant from other Ethereum projects in emotional space than Brew is from OpenZeppelin, despite the latter pair operating in very different domains. This suggests that governance practices and support structures may shape communication patterns more strongly than technical proximity. Temporal trajectories provide another perspective. When projects deviate sharply from the patterns observed in their functional category, or undergo sudden directional shifts from one year to the next, these changes can serve as early warning signals. In Web3.js, for instance, a marked move toward confusion coincided with major architectural changes. Such patterns indicate that monitoring emotional trajectories could help identify periods when community support and communication may require closer attention. This aligns with findings from Ortu et al. [8], who observed that positive emotions were associated with more effective collaboration and shorter issue resolution times. Consistent gratitude expression may thus signal not just politeness, but a behavioral norm that supports sustainable engagement — a hypothesis also supported in organizational literature on emotional intelligence and trust [44].

Finally, methodological considerations affect how these findings can be applied. The weighting scheme used in the EI enables comparison across projects with very different response speeds, such as Brew (median resolution of 378 h) and scikit-learn (9810 h). While this normalization supports comparison, it may obscure absolute differences in responsiveness that remain relevant for user experience. Moreover, the 27-emotion model reduces effectively to four dominant categories that capture 65%–80% of expression. This suggests that practical monitoring can focus on these four categories without significant loss of signal, simplifying application in real-world settings.

9. Threats to validity

Construct Validity: Primary threats involve our operationalization of emotions and contributor importance. The emotion detection is based on the *roberta-base-go_emotions* model, which may not fully capture the nuanced emotional content typical of technical discussions. The selection of paired emotions (curiosity vs. approval and gratitude vs. confusion) for the EI introduces a conceptual abstraction that, while grounded in prior work, may not reflect all relevant emotional dimensions in developer communication. Similarly, the contribution index aggregates direct code contributions, community engagement, and temporal involvement to identify influential developers. However, it may underrepresent other forms of influence, such as architectural decision-making or off-platform coordination. To address this, we employed multiple metrics and validated patterns across repositories to reduce over-reliance on any single measure.

Internal Validity: The relationship between emotional patterns and project characteristics may be influenced by unobserved external events, such as market shifts, security incidents, or leadership changes, that

occur independently of the internal dynamics we analyzed. To mitigate this, we adopted a longitudinal approach covering a ten-year period, allowing us to distinguish persistent trends from short-term fluctuations. Differences in repository size could also affect results. While our dataset includes both large and small projects, key findings are based on repositories with substantial activity. Additionally, although we used a validated model for bot detection, some misclassifications of automated accounts remains possible and may slightly distort emotional distributions.

External Validity: Our sample includes 14 repositories selected for their diversity in function and domain. While most originate from a blockchain ecosystem, we included four unrelated projects (scikit-learn, OpenSSL, Node.js, and Brew) to extend generalizability. Still, emotional patterns are shaped by each project's community structure, development context, and governance model. A further limitation stems from the training data of the RoBERTa-base-GoEmotions model. The GoEmotions dataset consists of English-language Reddit comments, but does not provide demographic or proficiency metadata about authors. As a result, the model's predictions may be influenced by informal or non-standard English usage, which is especially relevant when applied to developer communication in international, technical settings.

Conclusion Validity: We employed statistical techniques suited to the structure and limitations of our data, while acknowledging the constraints posed by small sample sizes in certain analyses. For RQ1, nonparametric and robust methods (Brown–Forsythe, PERMANOVA) were selected to reduce sensitivity to distributional assumptions, but limited numbers of repositories per cluster (2–4) reduce statistical power and increase the risk of inconclusive results. Accordingly, we interpret inferential outcomes as complementary to descriptive and visual analysis rather than as standalone evidence. For RQ3, The JSD provides a robust basis for comparing emotional profiles, but determining thresholds for "similarity" remains somewhat interpretive. To reduce this subjectivity, we report a range of descriptive statistics rather than relying on a single summary metric. An additional limitation concerns varying sample sizes when computing probability distributions for JSD analysis. Contributors with different activity levels yield emotional profiles derived from varying numbers of data points, potentially inflating JSD measurements for less active contributors. Despite this limitation, the substantial differences observed between mature projects (with near-zero JSD values) and developer tools (with consistently higher values) support the validity of our primary findings. Temporal analysis was conducted at annual resolution. While this smooths short-term fluctuations, it provides more stable emotional trajectories less affected by transient events. A complete replication package with datasets, scripts, and models is provided to support transparency and reproducibility (link).

Emotion detection reliability. The emotion classifier used in this study is based on a pretrained RoBERTa model evaluated in prior work, which reported moderate precision across coarse-grained emotions. While the model is capable of predicting 27 emotion categories, its performance is uneven across classes, partly due to training data imbalance. As a result, predictions for rare or ambiguous emotions may be less reliable. To mitigate this, we report aggregated distributions and focus on dominant emotional signals that persist across repositories. We acknowledge that the reliability of fine-grained emotion classification remains a potential source of measurement error.

Emotional Index design. The EI aggregates frequency, confidence, and discursive position into a single score. While the components are theoretically grounded, the weighting is heuristic and optimized for our dataset. Applying the index to different settings may require re-weighting or recalibration based on context-specific factors.

Scope of influence metrics. Our definition of contributor influence is based on observable on-platform activities: comments, issues, pull requests, and duration of engagement. This operationalization captures active and sustained participation but does not account for off-platform influence, such as architectural leadership or informal decision-making roles communicated via mailing lists, meetings, or other channels. While this is a recognized limitation, our focus is on measurable engagement patterns in publicly available repositories. Future studies could expand this analysis by integrating additional data sources to capture broader dimensions of influence.

10. Conclusion

This study contributes to our understanding of emotional dynamics in OSS development by analyzing 14 repositories over a decade. Our findings show that emotional patterns are not random or purely individual but shaped by project function, maturity, and contributor role. We observed that emotional communication aligns more with functional purpose than with technical domain or governance structure. This shift in perspective highlights how emotional norms reflect a project's role within the ecosystem and influence collaborative behavior. For maintainers and contributors, these patterns offer practical value. Emotional trajectories tend to follow predictable phases, such as the shift from confusion to gratitude-curiosity in Developer tools, indicating that communication challenges during transitions can be anticipated and addressed. We also found that core teams are not emotionally uniform. Distinct contributor communities often coexist, each with its own style. Larger groups tend to express gratitude and curiosity, while smaller ones lean toward approval and confusion, suggesting complementary forms of engagement. Overall, emotional expression is a structured and recurring feature of collaboration, not background noise. As OSS becomes increasingly central to software development, recognizing these emotional dynamics is central to supporting sustainable, resilient projects.

We plan to extend this study beyond within-repository patterns to trace cross-project contributor trajectories and test whether individuals adapt their emotional profiles to project-specific norms; examine whether cross-project activity reflects upstream/downstream coordination or deliberate alignment of development efforts; and relate affect to roles and work types (e.g., employment status, company affiliation, GUI vs. backend), using purpose-collected, consented metadata and ethically grounded methods that go beyond the aggregate, community-level analyses presented here.

CRediT authorship contribution statement

Matteo Vaccargiu: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Formal analysis, Data curation, Conceptualization. **Silvia Bartolucci:** Writing – review & editing, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Nicole Novielli:** Writing – review & editing, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Marco Ortu:** Writing – review & editing, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Roberto Tonelli:** Writing – review & editing, Validation, Formal analysis. **Giuseppe Destefanis:** Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data are shared.

References

[1] K. Nakakoji, Y. Yamamoto, Y. Nishinaka, K. Kishida, Y. Ye, Evolution patterns of open-source software systems and communities, in: Proceedings of the International Workshop on Principles of Software Evolution, 2002, pp. 76–85.

[2] J. Ayala, Y.-J. Tung, J. Garcia, Investigating vulnerability disclosures in open-source software using Bug Bounty reports and security advisories, 2025, arXiv preprint arXiv:2501.17748.

[3] X. Tan, Y. Zhang, J. Cao, K. Sun, M. Zhang, M. Yang, Understanding the practice of security patch management across multiple branches in oss projects, in: Proceedings of the ACM Web Conference 2022, 2022, pp. 767–777.

[4] F.R. Cogo, G.A. Oliva, A.E. Hassan, Deprecation of packages and releases in software ecosystems: A case study on npm, IEEE Trans. Softw. Eng. 48 (7) (2021) 2208–2223.

[5] A. Decan, T. Mens, A. Zerouali, C. De Roover, Back to the past–analysing backporting practices in package dependency networks, IEEE Trans. Softw. Eng. 48 (10) (2021) 4087–4099.

[6] A.A. Sawant, G. Huang, G. Vilen, S. Stojkovski, A. Bacchelli, Why are features deprecated? an investigation into the motivation behind deprecation, in: 2018 IEEE International Conference on Software Maintenance and Evolution, ICSME, IEEE, 2018, pp. 13–24.

[7] B. Chinthanet, R.G. Kula, S. McIntosh, T. Ishio, A. Ihara, K. Matsumoto, Lags in the release, adoption, and propagation of npm vulnerability fixes, Empir. Softw. Eng. 26 (3) (2021) 47.

[8] M. Ortu, B. Adams, G. Destefanis, P. Tourani, M. Marchesi, R. Tonelli, Are bullies more productive? Empirical study of affectiveness vs. issue fixing time, in: 2015 IEEE/ACM 12th Working Conference on Mining Software Repositories, IEEE, 2015, pp. 303–313.

[9] M. Mäntylä, B. Adams, G. Destefanis, D. Graziotin, M. Ortu, Mining valence, arousal, and dominance: possibilities for detecting burnout and productivity? in: Proceedings of the 13th International Conference on Mining Software Repositories, 2016, pp. 247–258.

[10] A. Murgia, P. Tourani, B. Adams, M. Ortu, Do developers feel emotions? an exploratory analysis of emotions in software artifacts, in: Proceedings of the 11th Working Conference on Mining Software Repositories, 2014, pp. 262–271.

[11] D. Graziotin, X. Wang, P. Abrahamsson, Software developers, moods, emotions, and performance, IEEE Softw. 31 (4) (2014) 24–27, http://dx.doi.org/10.1109/MS.2014.94.

[12] D. Graziotin, F. Fagerholm, X. Wang, P. Abrahamsson, What happens when software developers are (un) happy, J. Syst. Softw. 140 (2018) 32–47.

[13] N. Novielli, A. Serebrenik, Sentiment and emotion in software engineering, IEEE Softw. 36 (5) (2019) 6–23.

[14] G. Destefanis, M. Ortu, S. Counsell, S. Swift, M. Marchesi, R. Tonelli, Software development: do good manners matter? PeerJ Comput. Sci. 2 (2016) e73.

[15] D. Graziotin, F. Fagerholm, Happiness and the productivity of software engineers, in: Rethinking Productivity in Software Engineering, Springer, 2019, pp. 109–124.

[16] S. Wagner, F. Deissenboeck, Defining productivity in software engineering, in: Rethinking Productivity in Software Engineering, Springer, 2019, pp. 29–38.

[17] A. Filippova, H. Cho, The effects and antecedents of conflict in free and open source software development, in: Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing, 2016, pp. 705–716.

[18] Z. Chen, Y. Cao, H. Yao, X. Lu, X. Peng, H. Mei, X. Liu, Emoji-powered sentiment and emotion detection from software developers' communication data, ACM Trans. Softw. Eng. Methodol. (TOSEM) 30 (2) (2021) 1–48.

[19] M. Vaccargiu, R. Neykova, N. Novielli, M. Ortu, G. Destefanis, More than code: Technical and emotional dynamics in solidity's development, in: Proceedings of the 2025 IEEE/ACM 18th International Conference on Cooperative and Human Aspects of Software Engineering, CHASE, IEEE/ACM, 2025.

[20] T. Fritz, A. Begel, S.C. Müller, S. Yigit-Elliott, M. Züger, Using psycho-physiological measures to assess task difficulty in software development, in: 36th Int'L Conf. on Software Engineering, ICSE '14, Hyderabad, India - May 31 - 7 June, 2014, 2014, pp. 402–413, http://dx.doi.org/10.1145/2568225.2568266.

[21] D. Girardi, F. Lanubile, N. Novielli, A. Serebrenik, Emotions and perceived productivity of software developers at the workplace, IEEE Trans. Softw. Eng. 48 (9) (2022) 3326–3341, http://dx.doi.org/10.1109/TSE.2021.3087906.

[22] E. Guzman, B. Bruegge, Towards emotional awareness in software development teams, in: Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering, in: ESEC/FSE 2013, Association for Computing Machinery, New York, NY, USA, 2013, pp. 671–674, http://dx.doi.org/10.1145/2491411.2494578.

[23] N. Novielli, A. Serebrenik, Sentiment and emotion in software engineering, IEEE Softw. 36 (5) (2019) 6–23, http://dx.doi.org/10.1109/MS.2019.2924013.

[24] B. Lin, N. Cassee, A. Serebrenik, G. Bavota, N. Novielli, M. Lanza, Opinion mining for software development: A systematic literature review, ACM Trans. Softw. Eng. Methodol. 31 (3) (2022) http://dx.doi.org/10.1145/3490388.

[25] R. Cowie, Emotional life, terminological and conceptual clarifications: Technical report, FP6 Proj. 507422 (2006).

[26] D. Girardi, F. Lanubile, N. Novielli, A. Serebrenik, Emotions and perceived productivity of software developers at the workplace, IEEE Trans. Softw. Eng. 48 (9) (2021) 3326–3341.

[27] S.F. Huq, A.Z. Sadiq, K. Sakib, Is developer sentiment related to software bugs: An exploratory study on github commits, in: 2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering, SANER, IEEE, 2020, pp. 527–531.

[28] M. Ortu, S. Vacca, G. Destefanis, C. Conversano, Cryptocurrency ecosystems and social media environments: An empirical analysis through Hawkes' models and natural language processing, Mach. Learn. Appl. 7 (2022) 100229.

[29] A.S.M. Venigalla, S. Chimalakonda, Understanding emotions of developer community towards software documentation, in: 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Society, ICSE-SEIS, IEEE, 2021, pp. 87–91.

[30] D. Graziotin, X. Wang, P. Abrahamsson, Happy software developers solve problems better: psychological measurements in empirical software engineering, PeerJ 2 (2014) e289.

[31] M. Sánchez-Gordón, R. Colomo-Palacios, Taking the emotional pulse of software engineering — A systematic literature review of empirical studies, Inf. Softw. Technol. 115 (2019) 23–43, http://dx.doi.org/10.1016/j.infsof.2019.08.002, URL https://www.sciencedirect.com/science/article/pii/S0950584919301661.

[32] T. Rahayu Tulili, A. Rastogi, A. Capiluppi, Investigating developer sentiments in software components: An exploratory case study of gentoo, Softw.: Pr. Exp. (2025).

[33] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, A. Kappas, Sentiment strength detection in short informal text, J. Am. Soc. Inf. Sci. Technol. 61 (12) (2010) 2544–2558.

[34] B. Lin, F. Zampetti, G. Bavota, M. Di Penta, M. Lanza, R. Oliveto, Sentiment analysis for software engineering: How far can we go? in: 2018 IEEE/ACM 40th International Conference on Software Engineering, ICSE, 2018, pp. 94–104, http://dx.doi.org/10.1145/3180155.3180195.

[35] N. Novielli, F. Calefato, F. Lanubile, A. Serebrenik, Assessment of off-the-shelf SE-specific sentiment analysis tools: An extended replication study, Empir. Softw. Engg. 26 (4) (2021) http://dx.doi.org/10.1007/s10664-021-09960-w.

[36] F. Calefato, F. Lanubile, F. Maiorano, N. Novielli, Sentiment polarity detection for software development, in: Proceedings of the 40th International Conference on Software Engineering, ICSE '18, Association for Computing Machinery, New York, NY, USA, 2018, p. 128, http://dx.doi.org/10.1145/3180155.3182519.

[37] T. Ahmed, A. Bosu, A. Iqbal, S. Rahimi, Senticr: A customized sentiment analysis tool for code review interactions, in: 2017 32nd IEEE/ACM International Conf. on Automated Software Engineering, ASE, IEEE Press, 2017, pp. 106–111, http://dx.doi.org/10.1109/ASE.2017.8115623.

[38] M.R. Islam, M.F. Zibran, DEVA: sensing emotions in the valence arousal space in software engineering text, in: Proc. of the 33rd Annual ACM Symposium on Applied Computing, SAC 2018, Pau, France, April 09-13, 2018, 2018, pp. 1536–1543, http://dx.doi.org/10.1145/3167132.3167296.

[39] T. Zhang, I.C. Irsan, F. Thung, D. Lo, Revisiting sentiment analysis for software engineering in the era of large language models, ACM Trans. Softw. Eng. Methodol. 34 (3) (2025) http://dx.doi.org/10.1145/3697009.

[40] D. Demszky, D. Movshovitz-Attias, J. Ko, A. Cowen, G. Nemade, S. Ravi, Goemotions: A dataset of fine-grained emotions, in: D. Jurafsky, J. Chai, N. Schluter, J. Tetreault (Eds.), Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Online, 2020, pp. 4040–4054, http://dx.doi.org/10.18653/v1/2020.acl-main.372, URL https://aclanthology.org/2020.acl-main.372/.

[41] T. Bock, N. Alznauer, M. Joblin, S. Apel, Automatic core-developer identification on GitHub: A validation study, ACM Trans. Softw. Eng. Methodol. 32 (2023) 1–29, http://dx.doi.org/10.1145/3593803.

[42] A. Mockus, R.T. Fielding, J.D. Herbsleb, Two case studies of open source software development: Apache and mozilla, ACM Trans. Softw. Eng. Methodol. (TOSEM) 11 (3) (2002) 309–346.

[43] G. Robles, J.M. Gonzalez-Barahona, I. Herraiz, Evolution of the core team of developers in libre software projects, in: 2009 6th IEEE International Working Conference on Mining Software Repositories, 2009, pp. 167–170, http://dx.doi.org/10.1109/MSR.2009.5069497.

[44] I. Coronado-Maldonado, M.-D. Benítez-Márquez, Emotional intelligence, leadership, and work teams: A hybrid literature review, Heliyon 9 (10) (2023) e20356, http://dx.doi.org/10.1016/j.heliyon.2023.e20356, URL https://www.sciencedirect.com/science/article/pii/S2405844023075643.

[45] S.G. Barsade, The Ripple Effect: Emotional contagion and its influence on group behavior, Adm. Sci. Q. 47 (4) (2002) 644–675, http://dx.doi.org/10.2307/3094912, arXiv:https://journals.sagepub.com/doi/pdf/10.2307/3094912. URL https://journals.sagepub.com/doi/abs/10.2307/3094912.

[46] A. Cheshin, A. Rafaeli, N. Bos, Anger and happiness in virtual teams: Emotional influences of text and behavior on others' affect in the absence of non-verbal cues, Organ. Behav. Hum. Decis. Process. 116 (1) (2011) 2–16, http://dx.doi.org/10.1016/j.obhdp.2011.06.002, URL https://www.sciencedirect.com/science/article/pii/S0749597811000756.

[47] S.G. Barsade, C.G. Coutifaris, J. Pillemer, Emotional contagion in organizational life, Res. Organ. Behav. 38 (2018) 137–151, http://dx.doi.org/10.1016/j.riob.2018.11.005, URL https://www.sciencedirect.com/science/article/pii/S0191308518300108.

[48] W.B. Schaufeli, M. Salanova, V. González-Romá, A.B. Bakker, The measurement of engagement and burnout: A two sample confirmatory factor analytic approach, J. Happiness Stud. 3 (2002) 71–92.

[49] J.J. Hakanen, R. Perhoniemi, S. Toppinen-Tanner, Positive gain spirals at work: From job resources to work engagement, personal initiative and work-unit innovativeness, J. Vocat. Behav. 73 (1) (2008) 78–91.

[50] G. Hertel, S. Geister, U. Konradt, Managing virtual teams: A review of current empirical research, Hum. Resour. Manag. Rev. 15 (1) (2005) 69–95, http://dx.doi.org/10.1016/j.hrmr.2005.01.002, URL https://www.sciencedirect.com/science/article/pii/S1053482205000033.

[51] M. Vaccargiu, S. Aufiero, C. Ba, S. Bartolucci, R. Clegg, D. Graziotin, R. Neykova, R. Tonelli, G. Destefanis, Mining a decade of event impacts on contributor dynamics in Ethereum: A longitudinal study, in: 2025 IEEE/ACM 22nd International Conference on Mining Software Repositories (MSR), IEEE, 2025, pp. 552–563.

[52] M. Vaccargiu, S. Aufiero, S. Bartolucci, R. Neykova, R. Tonelli, G. Destefanis, Sustainability in blockchain development: A BERT-based analysis of ethereum developer discussions, in: Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering, EASE '24, Association for Computing Machinery, New York, NY, USA, 2024, pp. 381–386, http://dx.doi.org/10.1145/3661167.3661194.

[53] J. Coelho, M.T. Valente, L. Milen, L.L. Silva, Is this GitHub project maintained? Measuring the level of maintenance activity of open-source projects, Inf. Softw. Technol. 122 (2020) 106274, http://dx.doi.org/10.1016/j.infsof.2020.106274, URL https://www.sciencedirect.com/science/article/pii/S0950584920300240.

[54] M. Golzadeh, A. Decan, D. Legay, T. Mens, A ground-truth dataset and classification model for detecting bots in GitHub issue and PR comments, J. Syst. Softw. 175 (2021) 110911.

[55] M.M. Imran, Emotion classification in software engineering texts: A comparative analysis of pre-trained transformers language models, in: Proceedings of the Third ACM/IEEE International Workshop on NL-Based Software Engineering, 2024, pp. 73–80.

[56] E. Guzman, D. Azócar, Y. Li, Sentiment analysis of commit comments in GitHub: an empirical study, in: Proceedings of the 11th Working Conference on Mining Software Repositories, in: MSR 2014, Association for Computing Machinery, New York, NY, USA, 2014, pp. 352–355, http://dx.doi.org/10.1145/2597073.2597118.

[57] M. Ortu, B. Adams, G. Destefanis, P. Tourani, M. Marchesi, R. Tonelli, Are bullies more productive? Empirical study of affectiveness vs. Issue fixing time, in: 2015 IEEE/ACM 12th Working Conference on Mining Software Repositories, 2015, pp. 303–313, http://dx.doi.org/10.1109/MSR.2015.35.

[58] N. Novielli, D. Girardi, F. Lanubile, A benchmark study on sentiment analysis for software engineering research, in: Proceedings of the 15th International Conference on Mining Software Repositories, MSR '18, Association for Computing Machinery, New York, NY, USA, 2018, pp. 364–375, http://dx.doi.org/10.1145/3196398.3196403.

[59] S.S. Shapiro, M.B. Wilk, An analysis of variance test for normality (complete samples), Biometrika 52 (3–4) (1965) 591–611.

[60] M.B. Brown, A.B. Forsythe, Robust tests for the equality of variances, J. Amer. Statist. Assoc. 69 (346) (1974) 364–367.

[61] M. Anderson, PERMANOVA+ for PRIMER: Guide to Software and Statistical Methods, Primer-E Limited, 2008.

[62] D. Thissen, L. Steinberg, D.C. Kuang, Quick and easy implementation of the Benjamini-Hochberg procedure for controlling the false positive rate in multiple comparisons, J. Educ. Behav. Stat. 27 (2002) 77–83, http://dx.doi.org/10.3102/10769986027001077.

[63] H. Keselman, R.A. Cribbie, B. Holland, Controlling the rate of type I error over a large set of statistical tests., British J. Math. Statist. Psych. 55 Pt 1 (2002) 27–39, http://dx.doi.org/10.1348/000711002159680.