Enhancing Log Sentiments: An Exploratory Study of Sentiments and Emotions with Software Logs

XIAOHUI WANG, School of Computer Science, Wuhan University, China
YOUSHUAI TAN, School of Computer Science, Wuhan University, China
ZISHUO DING, The Hong Kong University of Science and Technology (Guangzhou), China
JINFU CHEN, School of Computer Science, Wuhan University, China
JIFENG XUAN, School of Computer Science, Wuhan University, China
WEIYI SHANG, University of Waterloo, Canada

Software logs serve as valuable resources for understanding system running and are extensively used in diverse software maintenance tasks. Logs are generated by logging statements in the code, which are written by developers. Therefore, logs may reflect developers' sentiments about the described situations. Consequently, when developers and system administrators read logs, the sentiments embedded in logs may influence their understanding. Although the sentiments associated with logs can convey valuable information, such information is not leveraged in research and practice. Previous research has primarily relied on verbosity levels of logs to gauge sentiments, which does not really capture the sentiments and emotions perceived by humans.

To bridge this gap, in this paper, we first conduct an exploratory study to investigate sentiments and emotions that are communicated within logs. Our study encompasses five anomaly log datasets from LogHub and a dataset involving eight open-source Apache Java projects. We find that 8% of the logs express sentiments and emotions though developers are suggested to write them in an objective way. While most log messages might not explicitly express sentiments and emotions, they can still implicitly evoke sentiments and emotions in those who read them. Therefore, we exploit issue reports referencing logs to capture such sentiments and emotions. In these issue reports, 47.5% exhibit emotions, with 54.7% of those emotions being related to logs and 8.1% directly addressing logs. Furthermore, we demonstrate the potential of leveraging sentiment analysis to complement verbosity levels in logs, showcasing how sentiment information can offer novel insights and enhance log analysis. Specifically, by applying automatic tools, we identify 41 issue reports (9.8% on average) with negative sentiment and 55 reports (13.2% on average) with negative emotions, all referencing INFO or DEBUG logs (i.e., low severity). After manually verifying and filtering exception logs, we uncover three main concerns from 22 critical instances.

1 INTRODUCTION

Software logs serve as an approach for understanding system behavior. Logs are widely used in software maintenance, e.g., debugging [12, 55, 57], anomaly detection [15, 16, 19], failure localization [8, 50, 60], performance

Jinfu Chen and Jifeng Xuan are the corresponding authors.

Authors' addresses: Xiaohui Wang, xiaohuiwangharriet@gmail.com, School of Computer Science, Wuhan University, Wuhan, Hubei, China; Youshuai Tan, tanyoushuai123@gmail.com, School of Computer Science, Wuhan University, Wuhan, Hubei, China; Zishuo Ding, zishuoding@hkust-gz.edu.cn, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, Guangdong, China; Jinfu Chen, jinfuchen@whu.edu.cn, School of Computer Science, Wuhan University, Wuhan, Hubei, China; Jifeng Xuan, jxuan@whu.edu.cn, School of Computer Science, Wuhan University, Wuhan, Hubei, China; Weiyi Shang, wshang@uwaterloo.ca, University of Waterloo, Waterloo, Ontario, Canada.

ACM 1557-7392/2025/3-ART
https://doi.org/10.1145/3723355

diagnosis and improvement [10, 29, 30]. While designed to objectively present system behavior, logs are essentially created by developers, potentially reflecting their sentiments towards the described situations. These embedded sentiments can influence how developers and system administrators interpret the logs. Additionally, the information contained within logs can also elicit their sentiments. Therefore, log sentiment analysis presents a valuable opportunity to extract implicit information that might be overlooked. For instance, the Reddit post[1] presents a log snippet characterized by subjective language with several sentiment words, i.e., "`strange`", "`dazed`", "`confused`". This elicits users' curiosity, prompting them to investigate the power-saving functionalities since the log snippet mentions "`a strange power saving mode enabled`".

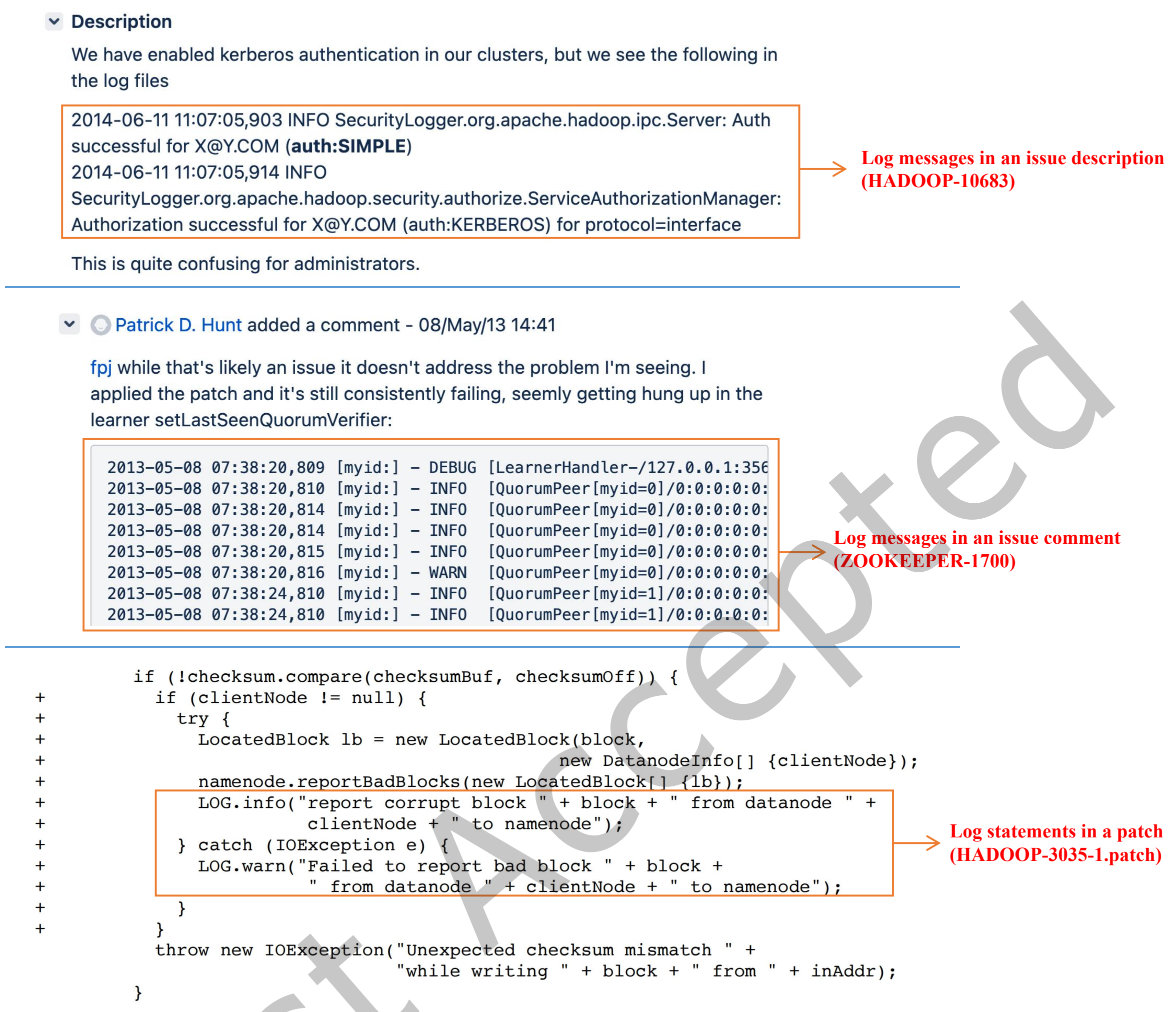
Despite the potential benefits of log sentiment analysis, existing research in this area is limited. Existing studies [4, 58] focusing on log sentiment analysis primarily rely on verbosity levels, e.g., ERROR and DEBUG, to differentiate between positive and negative logs. Another study employs a deep learning approach for log sentiment classification and automatically generates ground truth data through keyword matching. However, it lacks clear guidelines or justification for the selection of these keywords [49]. These approaches fail to capture the nuances of sentiments that humans can perceive. There is a lack of evidence demonstrating how these logs are construed as emotional.

To address this limitation, our study investigates the sentiments and emotions presented within log messages. For instance, from the ThunderBird log message "`lots of deferred mail, that is bad for performance`"[2], it is evident that the sentiment is negative due to the presence of the word "`bad`" which implies a negative emotion. However, it is a common and recommended practice to write log messages objectively [43]. This leads to the first challenge: ***Challenge 1*: It is not clear how practitioners express emotions in logs.** To address this challenge, we first explore the extent to which sentiments and emotions are present within log messages.

We conduct manual labeling of logs using Parrot's emotion framework [41], mapping these emotions to sentiments. We then compare the results with automated sentiment analysis tools like SentiStrength [51], SentiStrength-SE [23], and Senti4SD [6], all known for their accuracy in software engineering tasks [39]. Additionally, we employ the emotion analysis tool EmoTxt [7], capable of detecting six discrete primary emotions consistent with Parrot's emotion framework. Our initial analysis reveals that 8.1% of log messages convey emotions and sentiments, mostly negative. This aligns with the findings from specialized analysis tools (SentiStrength-SE, Senti4SD, EmoTxt). Moreover, our analysis highlights the potential to leverage the emotions and sentiments of log messages to help with log analysis and bug fixing.

While most log messages might not explicitly express emotions, they can still implicitly evoke emotions in those who read them. For example, HADOOP-13693[3] highlights that a warning message during a successful operation can cause confusion for administrators [25]. This introduces the second challenge: ***Challenge 2*: Capturing the implicit emotions and sentiments perceived by readers remains a complex task.** To address this challenge, we shift our focus to issue reports. Logs are generated from logging statements that are originally written by developers to monitor system status or facilitate troubleshooting. In common software engineering practices, issue reports are a primary channel through which developers describe observed issues and their code changes. Therefore, on the one hand, logs and issue reports often share overlapping information, such as emotions and problem descriptions. On the other hand, issue reports are an intuitive channel where developers are able to express their emotions and sentiments about logs. Approximately 20% of issue reports contain log messages [9]. As shown in Figure 1, log information can be found in various components of issue reports (i.e., descriptions, comments, patches). Furthermore, developers write issue reports in natural language, making the expressed emotions easier to understand for both humans and existing sentiment analysis tools.

[1]https://www.reddit.com/r/Proxmox/comments/xionvx/nmi_spurious_interrupts/
[2]https://github.com/logpai/loghub
[3]https://issues.apache.org/jira/browse/HADOOP-13693



Description

We have enabled kerberos authentication in our clusters, but we see the following in the log files

```
2014-06-11 11:07:05,903 INFO SecurityLogger.org.apache.hadoop.ipc.Server: Auth
successful for X@Y.COM (auth:SIMPLE)
2014-06-11 11:07:05,914 INFO
SecurityLogger.org.apache.hadoop.security.authorize.ServiceAuthorizationManager:
Authorization successful for X@Y.COM (auth:KERBEROS) for protocol=interface
```

This is quite confusing for administrators.

Patrick D. Hunt added a comment - 08/May/13 14:41

fpj while that's likely an issue it doesn't address the problem I'm seeing. I applied the patch and it's still consistently failing, seemly getting hung up in the learner setLastSeenQuorumVerifier:

```
2013-05-08 07:38:20,809 [myid:] - DEBUG [LearnerHandler-/127.0.0.1:356
2013-05-08 07:38:20,810 [myid:] - INFO  [QuorumPeer[myid=0]/0:0:0:0:0:
2013-05-08 07:38:20,814 [myid:] - INFO  [QuorumPeer[myid=0]/0:0:0:0:0:
2013-05-08 07:38:20,814 [myid:] - INFO  [QuorumPeer[myid=0]/0:0:0:0:0:
2013-05-08 07:38:20,815 [myid:] - INFO  [QuorumPeer[myid=0]/0:0:0:0:0:
2013-05-08 07:38:20,816 [myid:] - WARN  [QuorumPeer[myid=0]/0:0:0:0:0:
2013-05-08 07:38:24,810 [myid:] - INFO  [QuorumPeer[myid=1]/0:0:0:0:0:
2013-05-08 07:38:24,810 [myid:] - INFO  [QuorumPeer[myid=1]/0:0:0:0:0:
```

```
          if (!checksum.compare(checksumBuf, checksumOff)) {
+           if (clientNode != null) {
+             try {
+               LocatedBlock lb = new LocatedBlock(block,
+                                   new DatanodeInfo[] {clientNode});
+               namenode.reportBadBlocks(new LocatedBlock[] {lb});
+               LOG.info("report corrupt block " + block + " from datanode " +
+                         clientNode + " to namenode");
+             } catch (IOException e) {
+               LOG.warn("Failed to report bad block " + block +
+                         " from datanode " + clientNode + " to namenode");
+             }
+           }
            throw new IOException("Unexpected checksum mismatch " +
                                  "while writing " + block + " from " + inAddr);
          }
```

Log messages in an issue description (HADOOP-10683)
Log messages in an issue comment (ZOOKEEPER-1700)
Log statements in a patch (HADOOP-3035-1.patch)

Fig. 1. Log information in different components of JIRA issue reports.

Moreover, sentiment analysis and emotion analysis for issue reports have been extensively conducted and are relatively mature [24, 35, 37, 38, 40, 54]. Therefore, we consider that analyzing sentiments and emotions within issue reports can provide insights into the emotions associated with the referenced logs.

To study whether issue reports can assist in analyzing the sentiments and emotions associated with logs, we manually annotate the issue report descriptions referencing log messages from Chen et al.'s dataset [9]. Such annotation helps us understand whether these issue descriptions convey human-perceivable sentiments and emotions, and to what extent these sentiments and emotions are linked with the corresponding log messages. In addition, we apply automatic analysis tools to the natural language part of issue reports (excluding logs and code) and compare the results with manual annotations. Our study reveals that 47.5% of issue report descriptions containing logs exhibit emotions, with 54.7% of these emotions associated with the referenced logs.

Furthermore, we explore whether the log sentiments and emotions obtained from the issue reports referencing logs can provide novel insights into log analysis and assist in downstream log analysis tasks. By comparing the information provided by log verbosity levels, we recognize the potential of log-related sentiments and emotions conveyed in issue reports to aid tasks like improving logging and fixing bugs. In this context, automatic tools identify 41 issue reports (9.8% on average) with negative sentiments and 55 reports (13.2% on average) with negative emotions, all referencing INFO or DEBUG logs (i.e., low severity). After manually verifying and filtering exception logs, we uncover three main concerns from 22 critical instances.

The contributions of this paper include:

- We conduct an exploratory study of the challenges associated with log sentiment analysis using a large dataset.
- To the best of our knowledge, this is the first study in software engineering that leverages issue reports to complement log sentiment analysis.
- We demonstrate the potential of log sentiment analysis in aiding downstream tasks like improving logging and fixing bugs.

Paper Organization. Section 2 introduces our research questions. Section 3, 4, and 5 present the three parts of our exploratory study, with their corresponding results. Section 6 and 7 present the discussions and threats to the validity of our study. Section 8 discusses the related prior research to this paper. Finally, we conclude this study in Section 9. The replication package including the data and manual labeling results is publicly accessible[4].

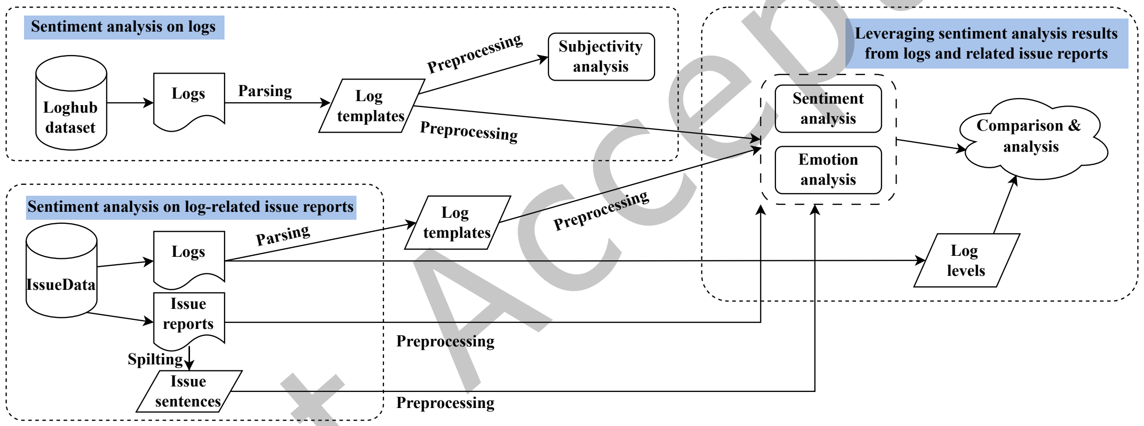


Fig. 2. An overview of our study.

2 RESEARCH QUESTIONS

Our empirical study investigates three research questions (RQs):

- **RQ1 (Sentiment Analysis on Logs):** To what extent are the sentiments and emotions expressed in the objective logs?
- **RQ2 (Sentiment Analysis on Log-Related Issue Reports):** Can issue reports aid in analyzing the sentiments and emotions of logs? To what extent are the sentiments and emotions expressed in the issue reports relevant to logs? What are the distributions of sentiments and emotions within these reports?
- **RQ3 (Leveraging Sentiment Analysis Results from Logs and Related Issue Reports)**: Can the log sentiments obtained from the issue reports bring new insights to log analysis, and can they contribute to downstream tasks?

[4]https://github.com/harrietwhh/Log-Sentiment-Empirical-Study-Data

These three RQs are progressively connected, as each research question builds upon the findings of the preceding one. In RQ1, we aim to explore whether sentiments and emotions are explicitly expressed in log text and evaluate the performance of existing sentiment analysis tools. We find that about 8.0% of log messages contain sentiments and emotions, mostly negative, providing a new perspective for log analysis and bug fixing. Nevertheless, it is worth noting that developers and system administrators may experience emotions when interacting with log text, suggesting that log text indirectly conveys emotions. Given the scarcity of emotions in log messages, capturing indirectly conveyed emotions and sentiments necessitates leveraging additional software artifacts. Therefore, we resort to issue reports for enhancement, which leads to RQ2.

In RQ2, we investigate whether issue reports aid in analyzing the sentiments and emotions of logs. Our study reveals that 47.5% of issue report descriptions containing logs exhibit emotions, with 54.7% of these emotions associated with the referenced logs. Compared to directly analyzing log messages, deriving sentiments from issue reports referencing logs yields triple as many instances, while deriving emotions yields nine times as many. With these derived sentiments and emotions, a natural research question arises: how can we utilize them? Therefore, in RQ3, we investigate whether the captured sentiments and emotions can facilitate downstream tasks. The overview of our study is presented in Figure 2.

3 SENTIMENT ANALYSIS ON LOGS

In this section, we start our exploratory study by applying sentiment analysis on software log messages directly. We answer: **RQ1 (Sentiment Analysis on Logs):** To what extent are the sentiments and emotions expressed in the objective logs? Specifically, we demonstrate the motivation, studied datasets, approach, and results of RQ1.

3.1 Motivation

Log messages are critical for software developers to understand system behavior and troubleshoot issues. Developers are encouraged to write logs objectively (i.e., provide actionable factual information and avoid venting emotions). However, previous research finds that logging is often subjective and arbitrary in practice [56]. Since logs are written by developers, they inevitably reflect the developers' sentiments about the situations described. This part of our study explores the extent to which sentiments and emotions are expressed in the objective logs and whether logs containing sentiments provide additional insights into system behavior and developer experience.

3.2 Studied datasets and data preparation

We utilize a benchmark log dataset containing anomaly annotations from Loghub[5] [61]. Loghub is a widely-used resource for log analysis research, encompassing a total of 19 log datasets collected from various software systems [61]. Among these datasets, five of them (i.e., BGL, Hadoop, HDFS, OpenStack, and Thunderbird) are labeled for the task of log-based anomaly detection. For our study, we utilize these five labeled datasets from Loghub, as they have undergone a rigorous validation process, ensuring high data quality and reliability. In summary, the Loghub dataset provides logs from a diverse range of system types, including systems with different purposes (e.g., high-performance computing, distributed computing, cloud computing, and email), varying technical architectures (e.g., distributed, centralized, and client-based), and distinct user groups (e.g., enterprises/organizations and individual users).

3.2.1 Deriving log templates. Following previous work in log analysis tasks [34, 53, 59], we also adopt log templates for sentiment analysis, as raw log messages frequently contain extraneous information such as IP addresses and ports, which can hinder the sentiment analysis process. As shown in Figure 3, a typical

[5]https://github.com/logpai/loghub

log message [52, 62] is semi-structured, involving objective fields such as *timestamp* (e.g., ``2005-08-03-13.46.20.777338'') and *verbosity level* (e.g., ``ERROR''), along with *log text* describing the event that occurred (e.g., ``Bad cable going into LinkCard (20393750343838370000000000594C31314B34333237303248) Jtag (0) Port (C) - 1 bad wire''). The log text is pre-organized by developers in logging statements, incorporating both *static natural language text* (e.g., ``Bad cable going into LinkCard'') and *variables* (e.g., ``20393750343838370000000000594C31314B34333237303248'') [17, 26]. During log parsing, a log text is converted into a log template [52], with variables replaced by wildcard tokens (e.g., ``<*>'') to remove the redundant information. As a result, log templates primarily consist of the natural language texts that the developers provide.

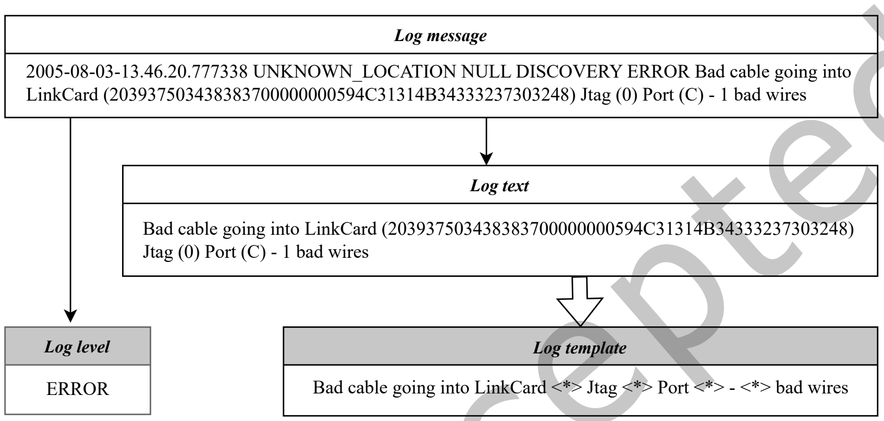

Log message
2005-08-03-13.46.20.777338 UNKNOWN_LOCATION NULL DISCOVERY ERROR Bad cable going into LinkCard (20393750343838370000000000594C31314B34333237303248) Jtag (0) Port (C) - 1 bad wires
Log text
Bad cable going into LinkCard (20393750343838370000000000594C31314B34333237303248) Jtag (0) Port (C) - 1 bad wires
Log level
ERROR
Log template
Bad cable going into LinkCard <*> Jtag <*> Port <*> - <*> bad wires

Fig. 3. An example for illustrating log-related terminology.

Log templates are readily available for the BGL and HDFS projects from the Loghub benchmark. However, for the remaining three projects, i.e., OpenStack, Thunderbird, and Hadoop, the templates are not readily available. We use Drain3[6] to parse the log messages and extract the corresponding log templates. We chose Drain3 for its improved log formatting performance and ease of use. As shown in Table 1, this process results in 2,320, 7,035, and 458 unique templates for OpenStack, Thunderbird, and Hadoop, respectively.

The automatically generated parsing results by Drain3 may still contain parsing mistakes, which could adversely affect the subsequent sentiment analysis. To address this, we manually examine the parsed log messages, using the sample log templates provided in Loghub as a reference, and also include these sample log templates in our analysis. This manual inspection ensures the accuracy of the parsed templates and reduces redundancy. Specifically, we identify 47 templates for OpenStack and 283 templates for Hadoop. Due to the complexity and abundance of log information in Thunderbird, we do not manually inspect its parsed results. Instead, we focus solely on the 149 unique log templates identified from the 2,000 log messages provided by Loghub. Finally, we obtain a total of 886 log templates.

3.2.2 Preprocessing data. Next, we preprocess log templates to ensure they can be interpreted by sentiment and emotion analysis tools. Specifically, we split camel-case words into separate words (e.g., ``DeclareServiceNetworkCharacteristics'' becomes ``Declare Service Network Characterist- ics''), insert spaces around periods (e.g., ``db.properties'' becomes ``db . properties''), and insert spaces around underscores (e.g., ``LOAD_MESSAGE'' becomes ``LOAD _ MESSAGE''). Additionally, we remove non-alphabet characters, convert text to lowercase, and eliminate extra spaces to reduce noise in the text.

[6]https://github.com/logpai/Drain3

Table 1. Number of log templates adopted from open-source projects. (#Templates by Drain3: number of unique log templates generated by Drain3; #Templates by Human: number of unique log templates after manual analysis)

Projects	**#Templates by Drain3**	**#Templates by Human**
BGL	–	377
Hadoop	458	283
HDFS	–	30
OpenStack	2,320	47
Thunderbird	7,035	149

3.3 Study approach

In this subsection, we present our analysis of the preprocessed software log messages. In total, we conduct three types of analysis on log messages including: 1) subjectivity analysis, 2) sentiment analysis, and 3) emotion analysis.

3.3.1 Subjectivity analysis. Subjectivity detection is to decide whether a given text contains subjective opinions or objective information [31]. For instance, ``` ``Machine A decomm request was sent to standby.'' ``` conveys a factual statement about software without expressing any sentiments or emotions, while ``` ``Thanks to [asreekumar] for uncovering this issue !'' ``` expresses gratitude. Some researchers argue that sentiment analysis' significance hinges on text subjectivity [24, 36]. However, most research treats sentiment and subjectivity analysis separately, noting factual texts contain indirectly expressed opinions, and opinionated texts may include factual expressions [36]. In particular, we perform a subjectivity analysis using TextBlob[7]. TextBlob is a Python library designed to facilitate various natural language processing (NLP) tasks, e.g., part-of-speech tagging, and sentiment analysis.

3.3.2 Sentiment and emotion analysis. Sentiment analysis evaluates text positivity or negativity through sentiment scores and polarities. Although sentiment analysis has been extensively explored in the software engineering community, a recent study by Lin et al. [31] suggested that evaluators find it easier to agree on discrete emotions rather than sentiment polarities for software artifacts. Therefore, we also conduct an emotion analysis, which offers a more detailed perspective by categorizing emotions into distinct, identifiable states such as joy, sadness, and surprise [42]. We perform both manual and automatic analyses to ensure more accurate and reliable results.
Manual analysis of log sentiments and emotions. To manually assess the emotions and sentiments conveyed in log templates, we first identify the six basic emotions outlined in Parrot's framework. During the manual labeling, two authors of the paper independently annotate the emotions for each log template. The annotation process follows the explanations and examples of each emotion provided in a previous study [37], adhering to the widely used emotion framework [7, 37, 38, 40] established by Parrot [41].

Afterwards, following the work of Islam et al. [23], we map these emotions to sentiment polarities, as shown in Figure 4. Specifically, *joy* and *love* are mapped to positive sentiment polarity, while *anger*, *sadness*, and *fear* are mapped to negative polarity. Since *surprise* can be either positive or negative, another discussion is used to further determine its emotional polarity. For the Loghub dataset, two annotators have achieved substantial agreement [46] (Cohen's Kappa of 0.83).
Automatic analysis of log sentiments and emotions. Based on the findings of Obaidi et al.'s systematic mapping study [39], we choose the most frequently used general sentiment analysis tool, SentiStrength [51], along with the two leading sentiment analysis tools tailored for SE, SentiStrength-SE [23], Senti4SD [6] and EmoTxt [7], the predominant emotion analysis tool in software engineering related studies.

[7]https://textblob.readthedocs.io/en/dev/

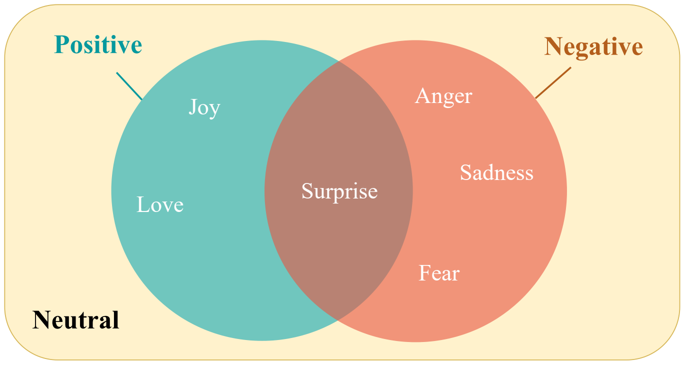


Fig. 4. Mapping between Parrot's six basic emotions and three sentiment polarities.

SentiStrength [51]: Designed for short informal texts, this lexicon-based classifier assigns sentiment scores (-5 to 5) to words based on a sentiment lexicon. The highest positive and negative scores are used to determine the overall sentiment of the text.
SentiStrength-SE [23]: This tool addresses the limitations of SentiStrength in the software engineering domain by leveraging domain-specific dictionaries and heuristics to classify sentiments in SE texts.
Senti4SD [6]: This tool uses machine learning (SVM) trained on Stack Overflow posts, to classify text polarity (positive, negative, or neutral) based on features like Bag of Words (BoW) and sentiment lexicons.
EmoTxt [7]: Developed for identifying emotions in text, EmoTxt uses tf-idf weighting, n-grams, and SVM techniques, along with WordNet Affect [48] for emotion lexical feature capture. EmoTxt identifies six basic emotions (joy, sadness, anger, fear, love, and surprise) in text. We do not opt to use ESEM-E [37] despite its superior performance in emotion analysis because its classification is limited to a few emotion categories, i.e., love, joy, and sadness, which are less prevalent in software engineering data.

These tools have been employed in over fifteen SE-related studies and have an accuracy exceeding 70% [39].

3.4 Study results

Over 20% of the log messages have subjective scores over 0.3. Figure 5 presents the results of subjectivity analysis on log templates. About 21.5% (186 out of 886) of the log templates are assigned subjective scores over 0.3 by Textblob. Notably, 4.7% (41 out of 886) of these templates have subjectivity scores surpassing 0.7, indicating a high level of subjectivity. This finding is beyond our expectations, as log messages are exploited to document data, activities, and other events in specific environments, avoiding conveying subjective opinions [43].

Therefore, for each log template, we manually examine the subjectivity results using our annotated log templates (annotated with Parrot's emotion framework). Among 72 log templates manually labeled as expressing emotions, 39 (54.2%) have subjectivity scores greater than 0.3. The subjectivity often relies on the sentiment words such as "`bad`", "`successful`", and " `abnormal`". We examine the remaining 33 (45.8%) log templates with subjectivity scores of 0.3 or less and find that 15 are incorrectly assigned scores by TextBlob. These 15 log templates include the words "success" or "succeed", which are given a score of 0, while "successful" and "successfully" are assigned a score of 0.95. The remaining 18 log templates show no evidence to classify them as subjective. For example, the log template "`Error receiving packet on tree network, expecting type <*> instead of type <*>`" meets the annotation guideline for "`unexpected, generally negative, behavior of the software`" [37], so it is considered to express *surprise*. However, it exhibits no subjectivity. This observation aligns with the contention that factual texts contain indirectly expressed opinions [36].

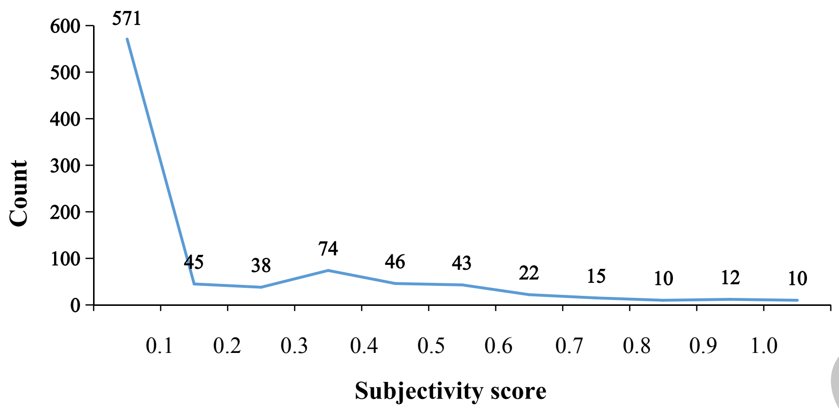


Fig. 5. Results of subjectivity analysis of log messages with TextBlob.

Table 2. Log emotion analysis results from manual annotation and EmoTxt.

Emotions	Manual Annotation	EmoTxt
Anger	0	28
Fear	0	28
Love	0	3
Joy	22	6
Sadness	21	30
Surprise	36	26
No emotions	814	782

Table 3. Log sentiment results from manual annotation and existing tools.

Tools	#Positive	#Neutral	#Negative
Manual	22	814	50
SentiStrength	49	523	314
SentiStrength-SE	10	843	33
Senti4SD	48	752	86

On the other hand, among 814 log messages manually labeled as expressing no explicit emotions, 667 (81.9%) log messages have subjectivity scores of 0.3 or less, while 147 (18.1%) log messages have subjectivity scores greater than 0.3. After manually examining these 147 log templates, we attribute the inconsistency to the presence of certain key adjectives e.g., "`critical`", "`active`", "`free`". However, these adjectives merely convey running information rather than emotions. Besides, some class names and method names may contain subjective words, e.g., "`super`" in "`JobHistoryEventHandler.super.stop()`".

Over 8.0% of the log messages contain emotions and sentiments, most of which are negative. Table 2 presents the emotion analysis results of studied log messages. Out of the 886 log templates analyzed, 72 are found to convey emotions, despite the expectation for log messages to maintain a neutral tone in documenting factual events. Among these, 29 express *surprise*, 14 convey *sadness*, 22 evoke *joy*, and seven express both *sadness* and *surprise*. Based on the manually annotated emotions, we derive the sentiment results, including 814 *neutral* log templates, 50 *negative* log templates, and 22 *positive* log templates, as shown in Table 3. As illustrated in Figure 4, all 22 templates expressing *joy* are categorized as positive while 50 templates expressing *sadness* or *surprise* are categorized as negative. Based on Z-test for population proportion [1, 63], the proportion of negative emotions is significantly higher than that of positive emotions (i.e., p-value < 0.01).

For the 22 log templates labeled as *joy*, annotators identify keywords like "`successfully`" and "`successful`". In these log templates, joy is typically linked to positive achievements. For the 21 log templates labeled as *sadness*, judgments are based on keywords such as "`bad`", "`unusable`", and "`not fully functional`". In these cases, sadness is generally associated with unsatisfactory situations. For the 36 log templates labeled as *surprise*, annotators note the occurrence of phrases like "`expect ... instead of ...`", "`abnormally`", "`greater than`", "`exceed`", "`unexpected`", "`has been ... but ...`", and "`wanted ... got ...`". In log templates, surprise is usually related to unexpected and typically negative software behavior.

Despite being annotated as containing emotions, these log templates maintain an objective writing style. For instance, in the log template ``` ``ciod: Unexpected eof at line <*> of node map file <*>'' ```, the determination of *surprise* is based on the presence of the word ``` ``unexpected'' ```. This log signifies the occurrence of unexpected events, which may pose challenges to the system's operation.

The more negative the emotions, the higher the severity of the verbosity levels of the log messages. Figure 6 uses the bar chart to show the distribution of various verbosity levels in log messages with different emotions. The verbosity levels taken into account include FAILURE, FATAL, ERROR, SEVERE, WARN, and INFO, listed in the decreasing order of severity [27, 28]. We find the emotions are generally consistent with verbosity levels, i.e., the more negative the emotions, the higher the severity of the verbosity level of the log messages. Based on contingency table analysis [2] and χ^2 test of independence [3], the proportions of log levels with the highest severity (FAILURE and FATAL) differ significantly across the three emotions (i.e., p-value < 0.05). Thus, we suggest that emotion can be used as a new indicator of log priority. To be specific, log messages containing *joy* have the highest proportion of INFO verbosity level (100%) and the lowest proportion of FAILURE and FATAL levels (0%). In contrast, log messages containing *sadness* have the lowest proportion of INFO (23.8%) and the highest proportion of FAILURE and FATAL levels (47.6%). Log messages expressing *surprise* span all verbosity levels, from FAILURE to INFO. Apart from FATAL (25.0%) and INFO (33.3%), the verbosity levels for surprise messages are nearly evenly distributed, ranging from 5.6% (ERROR) to 16.7% (WARN).

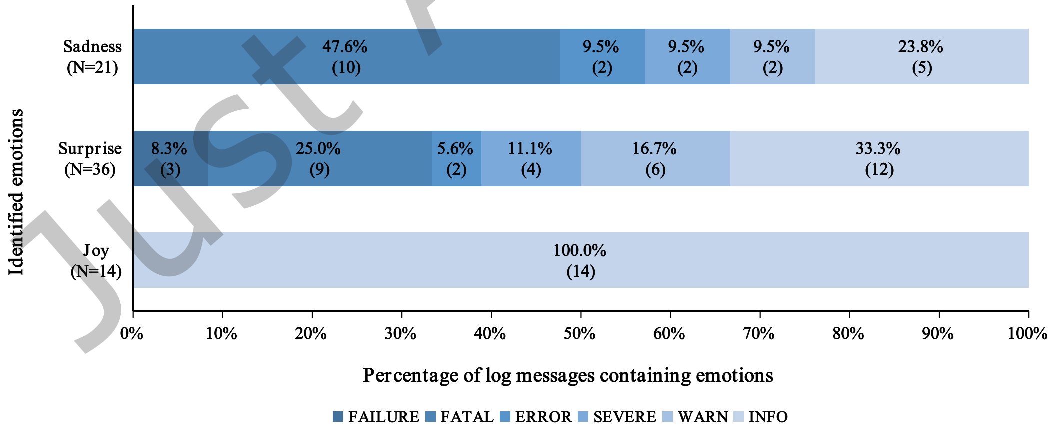


Fig. 6. Distribution of verbosity levels among identified emotions.

Emotions detected from log messages help refine the assigned verbosity levels. Developers often do not assign the right verbosity level at the first time [26, 56]. Thus, verbosity levels may not accurately reflect the severity, and emotions can serve as corrective signals in this context. For example, the log message below is set at the INFO level. However, this is misleading because INFO messages typically do not indicate any immediate problems but keep the user informed about the system's current operations and health [27]. The log text states that the system encountered an unexpected end of file (EOF) in the specified node map file, with the word "unexpected" expressing surprise. Therefore, this message should be set at least to the WARN level, as it reveals a potential issue about incomplete or corrupted data that needs attention but might not be critical [27]. Setting it as INFO can cause it to be overlooked as developers tend to identify system failures from tremendous log messages through specified log levels, e.g., error [33].

```
// Example 1:
   "2005-06-20-19.04.02.242115 R10-M0-N0-I:J18-U11 RAS KERNEL INFO ciod: Unexpected eof at line 8193 of node
       map file /p/gb2/cabot/miranda/newmaps/8k_128x64x1_8x4x4.map"
```

Sentiment analysis tools exhibit extreme performance variations, performing well for neutral sentiments but poorly for negative and positive sentiments. Table 4 shows that overall, tools tailored for software engineering, SentiStrength-SE and Senti4SD (F-Measures of 89% and 87%), outperform the general tool SentiStrength (F-Measure of 71%). The two tools (SentiStrength-SE and Senti4SD) perform well on neutral log messages, with F-Measure of 95% and 92%, respectively. However, their performance significantly declines for negative and positive log messages.

Table 4. Performance of sentiment tools on log messages.

Category	Precision			Recall			F-Measure		
	SentiStrength	SentiStrength-SE	Senti4SD	SentiStrength	SentiStrength-SE	Senti4SD	SentiStrength	SentiStrength-SE	Senti4SD
Negative	9%	36%	16%	54%	24%	28%	15%	29%	21%
Neutral	96%	93%	96%	62%	97%	88%	75%	95%	92%
Positive	43%	0%	46%	95%	0%	100%	59%	0%	63%
Overall	90%	88%	90%	62%	90%	85%	71%	89%	87%

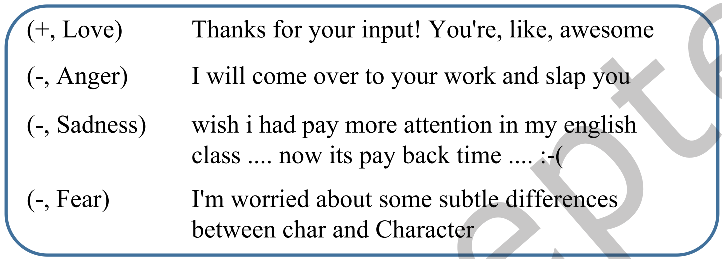
For negative sentiments, all tools show lower performance compared to neutral sentiments. SentiStrength-SE again stands out with the highest Precision (36%) and F-Measure (29%). We further analyze its false negatives and false positives and identify several issues. False negatives occur because the sentiment dictionary lacks negative emotion adjectives and adverbs like "`abnormally`", as well as other words that reflect emotions, such as "`already`", "`expected`", "`unexpected`", and "`but`", which are important for inferring sentiments and emotions from context. For instance, in the sentence "`Expected 10 active FanModules, but found 9`", although there are no explicit negative adjectives or adverbs, the phrase "`expected ... but`" conveys a sense of surprise. On the other hand, false positives arise from the presence of the term "`mess*`", which should be removed because it causes sentences containing "`message`" to be incorrectly classified as negative. SentiStrength shows the highest Recall (54%) for negative sentiments, yet this comes at the cost of lower Precision (9%). Compared to SentiStrength-SE, the sentiment dictionary of SentiStrength includes a broader range of words, resulting in higher recall but lower precision.

For positive sentiments, SentiStrength and Senti4SD both achieve perfect Recall (95% and 100%) with lower Precision (43% and 46%). By contrast, SentiStrength-SE fails entirely in this category with a Precision, Recall, and F-Measure of 0%. The poor performance of SentiStrength-SE in classifying positive sentiments is due to the absence of positive sentiment words in its sentiment dictionary, such as "`successful`", "`graceful`", and "`correctly`". To illustrate the limitations of the three tools, we present typical examples in Table 5.

SentiStrength: Although SentiStrength is the most commonly utilized sentiment analysis tool in SE-related research [39], it was designed to identify sentiments in short, informal texts, such as content found on social networking sites and discussion forums. Log templates typically contain a limited vocabulary dominated by

Table 5. Examples for three sentiment tools.

Id	Log template	Keyword	GT	Predicted sentiment		
				SentiStrength	SentiStrength-SE	Senti4SD
BGL-E7	<*>exited abnormally due to signal: Aborted	abnormally	Surprise	Negative	Neutral	Neutral
BGL-E28	Bad cable going into LinkCard <*>Jtag <*>Port <*>-<*>bad wires	bad	Sadness	Negative	Negative	Negative
BGL-E80	ciod: Unexpected eof at line <*>of node map file <*>	unexpected	Surprise	Neutral	Neutral	Neutral
BGL-E148	Expected 10 active FanModules, but found 9 (Found <*><*><*><*><*><*><*><*><*>).	expected...but...	Surprise	Neutral	Neutral	Neutral
BGL-E354	total of <*>ddr error(s) detected and corrected	-	-	Negative	Neutral	Neutral
Hadoop-E32	DefaultSpeculator.addSpeculativeAttempt – we are speculating task_<*>	speculating	Surprise	Negative	Negative	Neutral
HDFS-E20	<*>Unexpected error trying to delete block<*>BlockInfo not found in volumeMap<*>	unexpected	Surprise	Negative	Neutral	Neutral
OpenStack-E27	[instance: <*>] Claim successful	successful	Joy	Positive	Neutral	Positive



(+, Love)	Thanks for your input! You're, like, awesome
(-, Anger)	I will come over to your work and slap you
(-, Sadness)	wish i had pay more attention in my english class .... now its pay back time .... :-(
(-, Fear)	I'm worried about some subtle differences between char and Character

Fig. 7. Examples of text that explicitly express sentiments and emotions and can be perceived by humans.

domain-specific terminology, some of which are commonly perceived as negative in general sentiment analysis scenarios, such as "`kill`", "`error`", and "`die`". For example, in Example *HDFS-E20*, while SentiStrength successfully predicts the template as negative, it does so based on the presence of the word "`error`" rather than "`unexpected`". Consequently, in Example *BGL-E354*, where the log template is neutral, SentiStrength classifies it as negative based on the word "`error`", which leads to many false positives.

SentiStrength-SE: SentiStrength-SE, a version of SentiStrength customized for SE domain, was trained on Jira issue comments and leverages domain-specific dictionaries and heuristics. It mitigates the impact of SE-specific terms in SentiStrength. However, it also omits some common sentiment terms that are included in the original SentiStrength lexicon, such as "`abnormally`", "`success`", and "`succeed`". As a result, it fails to classify Example *BGL-E7* and *OpenStack-E27*.

Senti4SD: Senti4SD is an SVM-based polarity classifier trained on Stack Overflow posts with Bag of Words (BoW). However, this tool is limited by its training corpus, which reduces its generalizability. Consequently, it fails to classify Example *BGL-E7* and *Hadoop-E32*. It is advisable to consider the training corpus when selecting a sentiment analysis tool, as applying the tool to a similar corpus can help maintain the tool's effectiveness [31, 39].

All three tools successfully identify Example *BGL-E28* but fail to identify Examples *BGL-E80* and *BGL-E148*. Interestingly, while SentiStrength's lexicon includes the word "`unexpectedly`", it does not contain "`unexpected`". For Example *BGL-E148*, accurate sentiment detection requires understanding the logic of "`expected...but`". However, none of the three tools can handle such complex cases.

We employ EmoTxt to perform emotion analysis on log templates, and the result shows that among 886 templates, 104 are detected to contain emotions, achieving an overall accuracy of 83.9%. However, as shown in Table 2, 28 log templates are detected as expressing *anger*, 28 as expressing *fear*, and three as expressing *love*, which contradicts the manual annotations (i.e., 0 for each category). Besides, only one of the six detected as *joy* is manually annotated as joy. For *sadness*, only 13 out of 30 are annotated, and just one of the 26 detected as *surprise* is annotated as surprise. We attribute the poor performance to EmoTxt being trained on Stack Overflow posts and JIRA issue comments, while log templates represent a significantly different corpus.

***Summary*:** We find that about 8.0% of log messages convey emotions and sentiments, mostly negative. Generally, the more negative the emotions, the higher the severity of the verbosity levels in the log messages. Sentiment analysis tools show extreme performance variations, performing well with neutral sentiments but poorly with negative and positive ones.
***Implications*:** Our findings suggest a new perspective (i.e., sentiments and emotions of logs) for log analysis and bug fixing. Besides, emotions detected in log messages can help refine the assigned verbosity levels as the initial ones may be inaccurate. Moreover, adapting sentiment analysis to different software artifacts requires further exploration.

4 SENTIMENT ANALYSIS ON LOG-RELATED ISSUE REPORTS

In this section, we answer: **RQ2 (Sentiment Analysis on Log-Related Issue Reports):** Can issue reports aid in analyzing the sentiments and emotions of logs? In particular, we would like to examine to what extent the sentiments and emotions expressed in the issue reports are relevant to logs and whether we can leverage such information when using logs.

4.1 Motivation

In the previous section, we observe that logs typically lack explicit emotional undertones, as they are primarily intended for monitoring software status. However, logs are often referenced by issue reports when submitted by developers [9, 13]. These issue reports, written in natural language by users, can express a range of emotions. Figure 7 showcases examples of JIRA issue comments identified by Murgia et al. [38], which clearly convey feelings and emotions and are easily understood by humans. For instance, a warning log during a successful operation might be confusing (HADOOP-136937[8]), potentially leading to frustration [25]. Therefore, we hypothesize that issue reports may offer insights into the sentiments associated with logs. We focus on issue descriptions to understand how their sentiments and emotions relate to the referenced logs. This is because descriptions typically provide details related to issues (with logs often used as evidence for issues), whereas comments focus on solutions (e.g., test cases) [5, 13].

4.2 Studied dataset and data preparation

To investigate the sentiments associated with logs within issue reports, we select the dataset curated by Chen et al. [9] from issue reports of ten widely-used Java open-source systems (e.g., HDFS and YARN). This dataset comprises user-provided log messages extracted from 591 issue reports. To the best of our knowledge, it is the only available dataset that includes logs embedded in issue reports. For the remainder of this paper, we refer to this dataset as *IssueData*. Additionally, note that the term "*issue report*" refers to issue reports that reference logs (i.e., issue reports embedding log messages in their descriptions).

In the *IssueData* dataset, the logs are often embedded within the descriptions in issue reports. To ensure accurate sentiment analysis while minimizing the influence of other contents, we separately extract the logs and natural language descriptions presented in the "description" field (see Figure 1) within each issue report. We note that code snippets and logs are typically enclosed within *code* and *noformat* tags in issue descriptions. After extraction, we collect a total of 417 log snippets and accompanying natural language descriptions belonging

[8]https://issues.apache.org/jira/browse/HADOOP-13693

Table 6. Issue description emotion analysis results from manual annotation and EmoTxt.

Emotions	Manual Annotation	EmoTxt
Anger	4 (1.0%)	132 (31.7%)
Fear	32 (7.7%)	108 (25.9%)
Joy	11 (2.6%)	21 (5.0%)
Love	6 (1.4%)	14 (3.4%)
Sadness	81 (19.4%)	22 (5.3%)
Surprise	152 (36.5%)	99 (23.7%)
No emotions	219 (52.5%)	173 (41.5%)

to eight projects. To conduct a more fine-grained and accurate analysis of the descriptions, we also performed sentence tokenization on them, resulting in a total of 1,799 sentences.

Similar to the preparation process for the Loghub dataset (cf. Section 3.2.2), we apply Drain3 to the logs extracted from *IssueData*, followed by a manual examination to correct the derived log templates, leading to the templating of 2,017 log messages. Specifically, there are 54 log messages for ActiveMQ, 143 for Hadoop, 206 for HDFS, 557 for Hive, 218 for MapReduce, 282 for Storm, 486 for YARN, and 71 for ZooKeeper. Additionally, we perform the same preprocessing step for both the log templates and the extracted natural language descriptions.

4.3 Study approach

Similar to the approach to analyzing the logs (cf. Section 3.3), we also perform both automatic and manual analyses on issue descriptions at the sentence level and report level. Since the issue description comprises multiple sentences, each possibly expressing different sentiment polarities. Additionally, we observe emotions such as *confusion* and *curiosity* when annotating issue reports, which aligns with the findings in the work of Imran et al. [20]. Therefore, we follow their practice to categorize these two emotions into *surprise* because they are not included in Parrot's framework. For the *IssueData* dataset, two annotators have achieved moderate agreement [46] (Cohen's Kappa of 0.66). Moreover, we compare the sentiment and emotion analysis results on the logs that are referenced by the issue reports and on the issue reports themselves, in order to understand whether issue reports can provide more sentiment and emotion information.

4.4 Study results

Although merely 8.0% (72 out of 886) of the log messages contain emotions, 29.7% (124 out of 417) of the issue reports convey emotions relevant to logs. Based on Z-test for population proportion [1, 63], the proportion of emotional issue reports is significantly higher than that of log messages (i.e., $p\text{-value} < 0.01$). Through our manual analysis, we detect 198 issue report descriptions conveying emotions, accounting for 47.5%, as shown in Table 6. Among these, 31 issue report descriptions convey multiple emotions or express emotions toward multiple subjects, yielding a total of 234 instances. As illustrated in Table 7, issue reports conveying emotions are categorized into six distinct categories according to the emotion subjects, as detailed below:

- Interpretation of Logs: This includes running status that reporters find directly in the logs.
- Running Information: This refers to details about the running status reported by reporters that cannot be inferred from the logs. This information may indicate either issues or successful operations.
- Comments on Logs: This encompasses comments from reporters about the logs, including their feelings, complaints, or suggestions.
- Comments on Code: This is noted when reporters provide source code along with comments on it, which may include their feelings, complaints, or suggestions.

Table 7. Emotion subjects and corresponding example sentences and counts.

Emotion subjects	Example sentences	Counts
Interpretation of logs	I dont see any stacktraces in logs. But the debug logs show negative vcores-	109
Running information	The error is strange. socket.getRemoteSocketAddress() returned null implying this socket is not connected yet. But we have already read a few bytes from it!	78
Comments on logs	In debugging a NM retry connection to RM (non-HA), the NM log during RM down time is very misleading:	19
Comments on code	The problem is that we expect the index to be positive, but since it is a mod of hashcode it can be negative.	16
Solutions for issues	Possible resolution:... Cons: Someone can argue that it is hacky!	8
Colleagues	Thanks to [asreekumar] for uncovering this issue !	4
Total		234

Table 8. Issue description sentence sentiment results from manual annotation and SentiStrength-SE.

Tools	#Positive	#Neutral	#Negative
Manual	15	1533	251
SentiStrength-SE	24	1677	98

- Solutions for Issues: This pertains to potential solutions proposed by reporters for the issues described.
- Colleagues: This generally refers to instances where reporters express gratitude towards their colleagues within the issue description.

Out of these 234 instances, 128 are associated with logs, accounting for 54.7%, including interpretation of logs and comments directly on logs. For example, in the issue YARN-476[9], the reporter expresses confusion emotion and reports misleading logs to developers.

Although almost half of the issue reports convey emotions, each individual issue report contains fewer emotions at the sentence level. Among the 198 issue reports expressing emotions (out of a total of 417), 266 sentences convey emotions, as shown in Table 6 and Table 8. Interestingly, the average number of sentences per issue report is 3.8. However, within these issue reports, an average of 1.3 sentences convey emotions. This translates to roughly one-third (34.2%) of sentences in an issue report containing emotions on average. This highlights the potential value of analyzing issue reports to understand the emotional context surrounding logs.

Most of the emotions in issue descriptions are related to surprise. As shown in Table 6, of the six emotions examined, surprise, sadness, and fear are the most prevalent emotions found in issue descriptions. Specifically, 47.5% of the issue reports do express emotions (198 issue descriptions). Surprise, sadness, and fear represent 36.5%, 19.4%, and 7.7%, respectively. Conversely, joy, love, and anger are the least common, making up 2.6%, 1.4%, and 1.0%, respectively.

Besides, when conducting analysis at the issue sentence level, we have similar observations: as shown in Table 8, for the sentences with sentiments, most are labeled as negative (94.4%). This is reasonable, as issue reports are typically submitted when unexpected behaviors occur. For example, in issue HADOOP-2486[10], the reporter is surprised about the inconsistency observed during the execution of a MapReduce job and feels "weird". On

[9]https://issues.apache.org/jira/browse/YARN-476
[10]https://issues.apache.org/jira/browse/HADOOP-2486

Table 9. Performance of sentiment tools on issue reports referencing logs at the sentence level.

Category	Precision			Recall			F-Measure		
	SentiStrength	SentiStrength-SE	Senti4SD	SentiStrength	SentiStrength-SE	Senti4SD	SentiStrength	SentiStrength-SE	Senti4SD
Negative	22%	42%	27%	44%	16%	18%	30%	23%	22%
Neutral	90%	87%	88%	69%	95%	88%	78%	91%	88%
Positive	9%	17%	15%	80%	27%	87%	16%	21%	25%
Overall	80%	80%	79%	65%	84%	78%	71%	81%	78%

the other hand, 85.2% of the manually annotated sentences in issue descriptions display no obvious sentiments, which aligns with previous findings on issue comments [38].

Sentiment analysis tools exhibit extreme performance variations, performing well for neutral sentiments but poorly for negative and positive sentiments. We apply SentiStrength, SentiStrength-SE, and Senti4SD to 1,799 sentences, categorizing them into three sentiment polarities. Table 9 presents similar findings for issue reports in terms of overall results and the neutral class results, compared to the Loghub dataset analysis (cf. Section 3.4). Specifically, tools tailored for software engineering, SentiStrength-SE and Senti4SD, outperform the general tool SentiStrength.

For negative sentiments, all tools exhibit lower performance compared to neutral sentiments. SentiStrength stands out with the highest F-Measure (30%) and Recall (44%). This is because SentiStrength includes a broader range of sentiment words compared to SentiStrength-SE. By contrast, SentiStrength-SE achieves the highest Precision (42%) but the lowest Recall (16%). The poor performance of SentiStrength-SE in classifying negative sentiments can be attributed to the similar issues identified in the Loghub dataset analysis (cf. Section 3.4), i.e., lacking negative emotion words and other words that help infer emotions from context. Furthermore, we observe that, compared to log templates, issue descriptions contain a broader vocabulary and longer sentences with more complex grammar, along with typos (e.g., “I thank we should” instead of “I think we should”), which presents additional challenges for the application of lexicon-based approaches. As for positive sentiments, Table 9 shows similar results for issue reports compared to the Loghub dataset analysis (cf. Section 3.4). Besides, we attribute the poor performance to a similar issue.

We employ EmoTxt at the issue description level due to its capability to detect multiple emotions simultaneously. The analysis reveals that among the 417 issue descriptions, EmoTxt detects 132 issue descriptions expressing anger, 108 expressing fear, 14 expressing love, 21 expressing joy, 22 expressing sadness, 99 expressing surprise, and 173 descriptions exhibiting no discernible emotions, as shown in Table 6. We manually compare the emotions detected by EmoTxt with human annotations. For the 198 issue descriptions annotated as expressing emotions, EmoTxt matches the same emotions in 66 cases (33.3%). However, when these cases are mapped to sentiment polarities, the results are more consistent, with 149 cases (68.0%) aligning with the expected sentiment polarities. For the 219 issue descriptions with no explicitly annotated emotions, EmoTxt correctly identifies 129 cases (58.9%) as having no emotion (i.e., neutral sentiment).

When describing the same issue, the natural language descriptions in the issue reports contain more sentiments and emotions than user-provided logs. In this part, we apply the automatic tools to the IssueData dataset, focusing on two types of text: 1) issue reports referencing logs and 2) the corresponding referenced logs. It should be noted that issue reports are prepared as described in Section 4.2, with logs and code removed from the issue descriptions.

Due to the unsatisfying performance of the widely used SE-specific sentiment tools, we adjust the most effective tool, SentiStrength-SE, using our annotation experience and its customizable sentiment dictionary [23]. We refine the tool by incorporating additional sentiment adjectives and adverbs, such as “`successful`”, “`graceful`”, “`correctly`”, “`abnormally`”, “`undesired`”, “`confusing`”, and “`hacky`”. We also add negation terms like “`no`”, “`doesn't`”, and “`fail`”, as well as other words that reflect emotions, e.g., “`already`”, “`even`”, “`should`”, “`likely`”, and “`but`”.

Table 10. Performance of modified SentiStrength-SE on Loghub dataset and IssueData dataset.

Category	Precision		Recall		F-Measure	
	Loghub	IssueData	Loghub	IssueData	Loghub	IssueData
Negative	69%	60%	86%	64%	77%	62%
Neutral	99%	94%	97%	92%	98%	93%
Positive	76%	41%	100%	93%	86%	57%
Overall	97%	89%	96%	88%	96%	89%

Besides, we remove some words that cause false positives, such as "`mess*`", which leads to sentences containing "`message`" being classified as negative.

Table 10 presents the performance of modified SentiStrength-SE on Loghub dataset and IssueData dataset. After adjustments, the overall F-Measures improve from 93% to 96% for the Loghub dataset and from 83% to 89% for the IssueData dataset. Regarding the categories we are most concerned with, *Negative* and *Positive*, the modified SentiStrength-SE shows more improvement on the Loghub dataset, with recalls of 86% and 100% respectively, and F-Measures of 77% and 86%. The effectiveness is attributed to the limited vocabulary in the log messages. In contrast, the IssueData dataset shows F-Measures of 62% and 57%, with a high Recall of 93% for *Positive* but a much lower Recall (64%) for *Negative*. This discrepancy is due to the broader and more varied expressions of negative sentiment by humans [44].

Table 11. Sentiments detected from logs in issue reports and issue report descriptions.

Sentiments	Artifacts	Total	Projects							
			ActiveMQ	Hadoop	HDFS	Hive	MapReduce	Storm	YARN	Zookeeper
Positive	Logs in Issue Reports	18 (4.3%)	0 (0.0%)	1 (3.7%)	2 (3.3%)	3 (3.7%)	2 (3.8%)	3 (7.1%)	7 (5.6%)	0 (0.0%)
	Issue Descriptions	48 (11.5%)	4 (19.0%)	5 (18.5%)	9 (14.8%)	6 (7.3%)	9 (17.0%)	4 (9.5%)	9 (7.3%)	2 (28.6%)
Neutral	Logs in Issue Reports	332 (79.6%)	16 (76.2%)	19 (70.4%)	49 (80.3%)	68 (82.9%)	40 (75.5%)	32 (76.2%)	104 (83.9%)	4 (57.1%)
	Issue Descriptions	223 (53.5%)	9 (42.9%)	10 (37.0%)	23 (37.7%)	56 (68.3%)	27 (50.9%)	22 (52.4%)	74 (59.7%)	2 (28.6%)
Negative	Logs in Issue Reports	71 (17.0%)	5 (23.8%)	7 (25.9%)	10 (16.4%)	12 (14.6%)	11 (20.8%)	9 (21.4%)	14 (11.3%)	3 (42.9%)
	Issue Descriptions	182 (43.6%)	12 (57.1%)	16 (59.3%)	36 (59.0%)	25 (30.5%)	20 (37.7%)	20 (47.6%)	49 (39.5%)	4 (57.1%)
#Issues		417	21	27	61	82	53	42	124	7

As shown in Table 11, SentiStrength-SE categorizes the sentiments of text into three main categories: *positive*, *neutral*, and *negative*. Overall, we observe that the number of sentiments discovered through issue reports is over triple that of directly analyzing log texts, accounting for 46.1% compared to 14.7%. On average, across all projects, 3.9% of the log texts is classified as *positive*, while 24.3% is deemed *negative*. Conversely, the results obtained from analyzing issue reports reveal that 16.7% are *positive* and 53.9% are *negative*. Notably, Zookeeper exhibits the highest proportion of sentiments, with 42.9% identified in the log texts and 71.4% in the issue report descriptions.

As shown in Table 12, EmoTxt is employed to classify the emotions expressed in the text into six categories: *anger*, *fear*, *joy*, *love*, *sadness*, and *surprise*. We observe that overall, the total number of emotions detected using issue reports is nine times that obtained from direct analysis of log texts, accounting for 94.0% compared to 10.8%. Specifically, *anger*, *fear*, and *surprise* account for the highest proportions, averaging at 34.5%, 31.9%, and 29.0%, respectively.

Table 12. Emotions detected from logs in issue reports and issue report descriptions.

Emotions	Artifacts	Total	Projects							
			ActiveMQ	Hadoop	HDFS	Hive	MapReduce	Storm	YARN	Zookeeper
Anger	Logs in Issue Reports	12 (2.9%)	0 (0.0%)	1 (3.7%)	0 (0.0%)	0 (0.0%)	3 (5.7%)	2 (4.8%)	2 (1.6%)	4 (57.1%)
	Issue Descriptions	128 (30.7%)	10 (47.6%)	14 (51.9%)	20 (32.8%)	17 (20.7%)	12 (22.6%)	16 (38.1%)	39 (31.5%)	0 (0.0%)
Fear	Logs in Issue Reports	3 (0.7%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	2 (3.8%)	1 (2.4%)	0 (0.0%)	0 (0.0%)
	Issue Descriptions	108 (25.9%)	6 (28.6%)	14 (51.9%)	18 (29.5%)	16 (19.5%)	14 (26.4%)	9 (21.4%)	29 (23.4%)	2 (28.6%)
Joy	Logs in Issue Reports	2 (0.5%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	2 (1.6%)	0 (0.0%)
	Issue Descriptions	21 (5.0%)	1 (4.8%)	1 (3.7%)	2 (3.3%)	3 (3.7%)	3 (5.7%)	3 (7.1%)	7 (5.6%)	1 (14.3%)
Love	Logs in Issue Reports	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)
	Issue Descriptions	14 (3.4%)	1 (4.8%)	2 (7.4%)	3 (4.9%)	2 (2.4%)	3 (5.7%)	2 (4.8%)	1 (0.8%)	0 (0.0%)
Sadness	Logs in Issue Reports	8 (1.9%)	1 (4.8%)	1 (3.7%)	0 (0.0%)	0 (0.0%)	1 (1.9%)	3 (7.1%)	2 (1.6%)	0 (0.0%)
	Issue Descriptions	22 (5.3%)	0 (0.0%)	1 (3.7%)	3 (4.9%)	5 (6.1%)	0 (0.0%)	3 (7.1%)	9 (7.3%)	1 (14.3%)
Surprise	Logs in Issue Reports	20 (4.8%)	0 (0.0%)	1 (3.7%)	1 (1.6%)	7 (8.5%)	4 (7.5%)	1 (2.4%)	6 (4.8%)	0 (0.0%)
	Issue Descriptions	99 (23.7%)	5 (23.8%)	10 (37.0%)	17 (27.9%)	21 (25.6%)	9 (17.0%)	13 (31.0%)	22 (17.7%)	2 (28.6%)
#Issues		417	21	27	61	82	53	42	124	7

Summary: We find that 29.7% of the issue reports convey emotions relevant to logs, compared to only 8.0% of the log messages. When describing the same issues, compared to directly analyzing log messages, deriving sentiments from issue reports referencing these logs yields triple as many instances, while deriving emotions yields nine times as many.
Implications: The findings support the feasibility of using issue reports to analyze log sentiments. Not only do logs and issue reports often share overlapping information, but issue reports also contain a meaningful amount of emotions related to logs, making them a practical resource for such analysis.

5 LEVERAGING SENTIMENT ANALYSIS RESULTS FROM LOGS AND RELATED ISSUE REPORTS

In this section, we answer **RQ3 (Leveraging Sentiment Analysis Results from Logs and Related Issue Reports)**: Can the log sentiments obtained from the issue reports bring new insights to log analysis, and can they contribute to downstream tasks?

5.1 Motivation

According to previous research [26, 56], developers often implement logging subjectively and arbitrarily at first and make improvements afterward. For example, the extensively used verbosity levels may be inaccurate or even misleading, which hinders understanding and filtering the massive information in logs [25, 33]. Therefore, to explore the potential of leveraging sentiment and emotion information in issue reports when using logs, we study whether such sentiment information can be used to improve logging. We use critical logs as our research subject. According to Schmidt et al.'s definition [43], critical logs reflect severe and urgent events, such as successful or likely successful attacks, system changes that can lead to security issues, system crashes, systems reaching maximum capacity, and hardware errors. Systems facing these severe and urgent issues can result in significant losses[11], making critical logs a key focus.

Specifically, we focus on potentially critical logs with verbosity levels of *INFO* or *DEBUG*, as higher verbosity levels (such as FATAL level) inherently attract more attention from developers; while practitioners may not pay as

[11]Report: Software failure caused $1.7 trillion in financial losses in 2017

much attention to logs with INFO level. This way, we would like to examine whether the sentiment and emotion information generated by the corresponding issue reports referencing logs can better express the information in logs to complement verbosity levels.

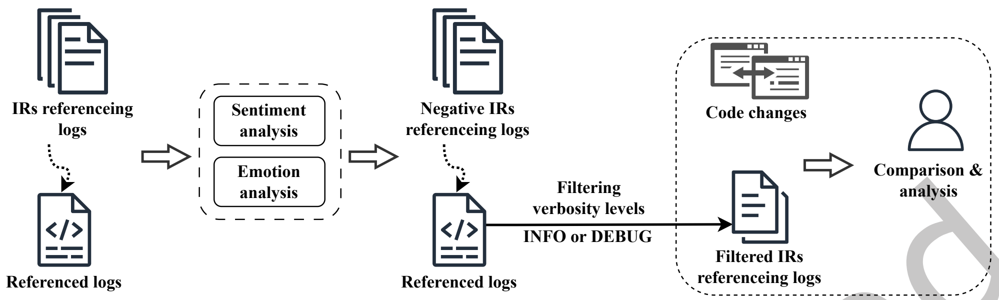


Fig. 8. Process of collecting potentially critical logs and conducting analysis.

5.2 Study approach

As shown in Figure 8, to collect potentially critical logs from the perspective of sentiment analysis, we select issue reports expressing negative emotions. Next, we filter verbosity levels to uncover logs lacking severe or urgent information based on their verbosity levels. Finally, we interpret the negative sentiments and emotions through the investigation of code changes provided by developer teams.

In particular, we first apply automatic sentiment and emotion analysis tools to issue reports from IssueData. Similar to the prior steps in our exploratory study (cf. Section 3 and 4), we use our modified SentiStrength-SE (cf. Section 4.4) for sentiment analysis and EmoTxt for emotion analysis. We then extract log messages from issue reports that satisfy two conditions: 1) the issue reports exhibit negative sentiment or emotions, and 2) the embedded logs have verbosity levels of INFO or DEBUG. Since these cases are detected by automated tools, which may not be fully accurate, we manually verify each instance. We retain only those instances where the sentiment and emotion analysis results are consistent between manual review and automated tools. Subsequently, we filter out instances with thrown exceptions in logs, resulting in the final critical instances. Based on these critical instances, we conduct manual analysis of their issue descriptions and code changes provided by developer teams (e.g., patches, source code files, commits).

5.3 Study results

As shown in Table 13, we find 41 issue reports (9.8% on average) attached with negative sentiment and 55 issue reports (13.2% on average) attached with negative emotions have referenced log messages with verbosity levels set to INFO or DEBUG. Among them, 33 instances are overlapping ones with both negative sentiments and emotions, resulting in 62 unique instances of logs. After manual verification, we obtain 39 instances. After excluding instances with thrown exceptions in logs, we are left with 22 critical instances. Based on them, we identify three types of concerns that are made according to the issue reports. We discuss each type of concern in detail with examples in the rest of this part of our study.

Concerning logging statements (6 instances). In this context, sentiments and emotions in the issue reports concern changing the log statements. In this case, reporters express negative emotions like surprise, confusion, and dissatisfaction with logs, concerning modification of log levels or descriptive text. For example, in the issue report HIVE-15309[12], the reporter complains about unnecessary logs causing the logging flood problem. We

[12]https://issues.apache.org/jira/browse/HIVE-15309

Table 13. Numbers of logs detected by comparing the sentiments and emotions in issue report descriptions with corresponding log verbosity levels.

Type	Total	AMQ	Hadoop	HDFS	Hive	MapReduce	Storm	YARN	Zookeeper
Sentiment Analysis	41 (9.8%)	4 (19.0%)	5 (18.5%)	7 (11.5%)	6 (7.3%)	2 (3.8%)	2 (4.8%)	15 (12.1%)	0 (0.0%)
Emotion Analysis	55 (13.2%)	6 (28.6%)	8 (29.6%)	10 (16.4%)	4 (4.9%)	4 (7.5%)	2 (4.8%)	19 (15.3%)	2 (28.6%)
Total #Issue Reports	417	21	27	61	82	53	42	124	7

conclude that either the log levels needed to be lowered or the number of log messages reduced. Actually, based on the patch submitted by the developer team for this issue, the log levels are adjusted from *INFO* to *DEBUG*. In the issue report HDFS-6797[13], the reporter complains that a log message confuses operators because the layout version of the data node is not updated to the "new LV" value after an upgrade. We determine that the description text of this log should be modified to address this confusion. Actually, based on the patch provided by the developer team for this issue, a variable in the log statement is modified to use the new data node layout version instead of the name node layout version. Moreover, the log statements can be removed if unnecessary. For example, in the issue report YARN-476[14], the reporter complains about the not-so-helpful, uninteresting, and misleading log messages. Actually, based on the patch provided by the developer team for this issue, a log statement is removed entirely.

Concerning runtime behaviour of software with logs as evidence (7 instances). This concern involves changing the source code. In this case, the emotions expressed by reporters are not aimed solely at log messages but at both logs and other sources of information. Typically, reporters present logs as evidence to compare with the actual software behaviors or to troubleshoot. During this process, negative emotions such as surprise, confusion, and dismay are aroused. For example, in the issue report AMQ-6343[15], the reporter intends to publish a message indicating that a device is disconnected from an MQTT broker. While the log messages indicate the disconnection, the reporter does not receive the expected LWT message, leading to confusion and disappointment. However, these emotions are not directed solely at the log statements and do not necessitate their modification. Instead, they serve to identify bugs in the code. Consequently, the developer team fix this bug without modifying log statements.

Concerning both runtime behaviour of software and logs (9 instances). This concern primarily focuses on altering the source code, with some improvements made to log statements as well. Similar to the second type of concern, in this case, reporters' emotions target both logs and other sources of information. They often use log messages as evidence to compare with actual software behaviors, leading to the arousal of negative emotions like surprise, confusion, and dismay. This type of modification differs in that it also improves logging by providing additional information or addressing logging flood issues while fixing bugs. This includes augmenting log texts with more details and reorganizing logging within the source code to determine where and what to log. For example, in the issue report YARN-1661[16], the reporter is confused by inconsistent results from Application Master logs and container logs. Apart from fixing the bug, the developer team also improves the logging statement to report the number of allocated and requested containers. This improvement offers more insightful information for debugging purposes, surpassing the previous version's simple notification of the Application Master's failure and exit.

[13]https://issues.apache.org/jira/browse/HDFS-6797
[14]https://issues.apache.org/jira/browse/YARN-476
[15]https://issues.apache.org/jira/browse/AMQ-6343
[16]https://issues.apache.org/jira/browse/YARN-1661

***Summary and Implications*:** We demonstrate the potential of log-related sentiments and emotions conveyed in issue reports to aid downstream tasks, such as improving logging and fixing bugs. Future studies could explore more downstream tasks based on the perspectives discussed in Section 6.

6 DISCUSSION

Improving logging practices. Developers often implement logging in a subjective and arbitrary manner initially, making adjustments and improvements over time [26, 56]. Recording sentiment and emotion information when logging could help preserve the original intent of logging. For instance, in the issue report HIVE-11062[17], the reporter suggests separating exception information from log messages and lowering the verbosity level of exception details from INFO to DEBUG to prevent user frustration, as these exceptions are harmless. Recording sentiment and emotion information for the original log message could help avoid such frustration.

However, in the current practice of logging, there exist no explicit channels to express sentiments and emotions associated with logs. The commonly used logging libraries, such as Log4j[18] and Python's logging library[19], allow developers to set a verbosity level threshold to filter which logs should be output and which should be ignored. On the one hand, verbosity level, as named after its purpose, should be used to control the amount of information that is communicated during runtime, instead of the qualitative characteristics (like emotion and sentiment) of the information.

On the other hand, verbosity levels are often predefined by common practice which may be inaccurate or even misleading [25, 33] to describe emotions and sentiments. Therefore, supporting the incorporation of sentiments and emotions as additional explicit labels in logs would further support the expression of information by developers during the execution of software systems.

Integrating with existing log-based tools. In the logging stage, sentiments and emotions can either be directly recorded in logging statements or integrated through logging libraries. This enables developers to preserve the original emotional attitude towards the logged events. In the log parsing stage, sentiments and emotions can be included in log parsing tools as an additional field. By adding this field, developers gain a new perspective on the log data, helping them better understand the context of log messages and prioritize logged events based on sentiments and emotions. In the log mining and analysis stage, if sentiments or emotions are recorded when logging, sentiments or emotions can act as a dimension to analyze logs. For instance, we can analyze the distribution of specific sentiments over time, and a sudden increase may indicate potential issues during that period. If logs are not tagged with sentiments or emotions, it becomes challenging to infer implicit ones. To mitigate this problem, logs can be jointly mined with other software artifacts referencing these logs, e.g., issue reports as we demonstrated in this study. Besides, logs are often employed for anomaly detection, and sentiments and emotions can enhance this process by making anomalies easier to identify. For instance, logs attached with surprise are more likely to indicate abnormal events, while logs attached with sadness are more likely to suggest bugs.

Generalizability. We used only execution logs in our study, which are typically presented to end-users and tend to be more objective. In contrast, pre-release logs, such as testing logs intended for internal review, often contain more subjective content, potentially including more sentiments and emotions. Monitoring logs, commonly used for real-time monitoring, performance analysis, and capacity planning, are less likely to include emotional expressions. The characteristics of different types of logs may influence the findings.

[17]https://issues.apache.org/jira/browse/HIVE-11062
[18]https://logging.apache.org/log4j/
[19]https://docs.python.org/3/library/logging.html

Enterprise software may have logs focusing on technical and operational details, while consumer-facing applications might emphasize user-friendly messages. The different styles of log messages may affect the sentiments and emotions captured in logs. Additionally, systems designed for specific industries (e.g., healthcare, finance, or gaming) often use domain-specific terminologies and have unique logging practices, further affecting the sentiments and emotions conveyed in logs. Therefore, we suggest that future studies explore a broader range of software systems, including both industry-specific and consumer-facing applications, when analyzing sentiments and emotions in logs and issue reports. This would provide a more comprehensive understanding of how different system types impact the sentiments and emotions in logs.

This study does not account for potential cultural differences in analyzing sentiments and emotions in logs and issue reports. Prior studies have shown that app reviews exhibit significant differences in features such as sentiment, content, and length across different countries [18, 47]. For instance, reviews from South Africa often display more positive language, whereas those from Confucian-Asian countries typically adopt a more neutral tone. It is worth considering whether similar cultural biases exist in logging practices. Since logging is a global practice, it is highly likely that the cultural background of developers influences the way logs are written, thus influencing the proportion of sentiments and emotions in logs. Understanding these cultural nuances could help integrate sentiments and emotions in log-based tools and ensure their applicability across diverse contexts. Future work could explore the extent of cultural diversity in logs and how it impacts log interpretation and analysis.

7 THREATS TO VALIDITY

In this section, we discuss the threats to the validity of our research.

External Validity. In this paper, we conduct analysis on two datasets (i.e., Loghub dataset [61], and IssueData [9]). Both datasets focus on open-source systems that are mainly written in Java. Evaluating our analysis on other systems that are written in other languages, with closed-source code, may further demonstrate the implications and limitations of our analysis. In our analysis concerning issue reports, they are from IssueData [9]. All the issue reports are categorized as "Bug" and priority labeled as "Major" or higher. Therefore, our findings might not be generalizable to other types of issue reports. Further study can apply our analysis to other types of issue reports, e.g., the ones categorized as "Improvement", or priority labeled as "Minor". In RQ3, we identify three main concerns from a subset of 22 critical instances drawn from 417 issue reports. As the scope is limited to the specific dataset and context, future studies should explore larger datasets and diverse contexts to determine whether these concerns hold true in broader scenarios.

Internal Validity. We study the sentiments and emotions in log messages and issue reports. However, such emotions may not be directly and fully related to the software under study or the artifacts under study. There may be other factors that can impact the sentiments and emotions of developers when working with the studied software artifacts. More in-depth studies with developers may further help understand the emotions and sentiments in logs. For sentiment and emotion analysis, we adopt three SE-specific state-of-the-art automatic tools (i.e., SentiStrength-SE, Senti4SD, and EmoTxt) and one widely used general tool, SentiStrength. Employing other tools may lead to different results as it is widely admitted that there are discrepancies between the results of various tools [20, 32, 39].

The Loghub dataset used in our study is from Loghub 1.0, which does not include Hadoop, OpenStack, and Thunderbird log templates derived from full log datasets. For Hadoop and OpenStack, we parsed the logs from Loghub 1.0 using Drain 3 and made manual modifications to derive 47 and 283 log templates, respectively. With the release of Loghub 2.0, which provides log templates from full log datasets, we compared the two versions and found minimal differences. Specifically, for OpenStack, Loghub 2.0 includes two additional templates compared to ours. For Hadoop, Loghub 2.0 includes two additional templates compared to ours but misses 25 templates due to differences in the full log datasets. For instance, Loghub 2.0's dataset lacks stack traces, which are present in

Loghub 1.0. This demonstrates that the log templates we derived manually might actually be a better option for this study. For Thunderbird, we used 149 log templates derived from 2K logs in LogHub 1.0. The χ^2 test comparing these templates with a 95% confidence level sample from LogHub 2.0 found no significant differences (p-value is 0.16). We provide the differing log templates for Hadoop and OpenStack, along with the 95% confidence level sample for Thunderbird, in our replication package.

Construct Validity. Human annotation of sentiments and emotions in log messages and issue reports is inherently subjective as humans perceive emotions differently. To alleviate this issue, we followed Parrot's emotion framework [41] and the annotation guideline from prior work [37]. To address this issue, future research can involve more human annotators. Nonetheless, it is important to note that inter-annotator agreement among humans is often low [32, 39], and even for humans, distinguishing between different emotions can be challenging [42].

8 RELATED WORK

Sentiment analysis assesses text positivity or negativity through sentiment scores and polarities, while emotion analysis offers a more fine-grained perspective and generally employs two main approaches: discrete and dimensional methods [42]. The former categorizes emotions into distinct, identifiable states like joy, sadness, and surprise. The latter employs various dimensions to depict diverse aspects of emotions, e.g., intensity, and polarity, which is also known as the Valence-Arousal-Dominance (VAD) approach. We present related work from two aspects: sentiment and emotion analysis for logs, and sentiment and emotion analysis for issue reports.

8.1 Sentiment and emotion analysis for logs

By far, though sentiment analysis is prevalent in the SE field, it is rarely applied to logs. To the best of our knowledge, only three works [4, 49, 58] focus on log sentiments. However, while they claimed to analyze or utilize log sentiments, two of them relied on the verbosity level of logs as a criterion to differentiate between positive and negative logs. Zhang et al. [58] designated logs with a verbosity level of *ERROR* as negative and those with a verbosity level of *DEBUG* as neutral, using these labels for supervised learning (i.e., BiLSTM model). Similarly, Alharthi et al. [4] employed these labels as the explained variable in a regression model to construct a sentiment dictionary, implying that words in logs with verbosity levels of *ERROR* or *FATAL* were more likely to be considered negative. The derived sentiment dictionary is then utilized to assign sentiment scores to log messages. It appears that these two studies have not truly delved into sentiment analysis because this diverges from our typical understanding of sentiment analysis or emotion analysis, which involves assessing whether the text expresses positive or negative emotions.

Studiawan et al. [49] employed a deep learning approach, i.e., Gated Recurrent Unit (GRU) with word embedding, for log sentiment classification. Although they did not assign sentiment labels based on verbosity levels, their model was trained on automatically generated ground truth data using keyword matching, as indicated in their replication package[20]. However, they failed to provide any guidelines or explanations for this process, nor did they demonstrate the relevance of these keywords to emotions. As a result, there is a lack of evidence demonstrating how these logs are construed as emotional.

Therefore, we aim to conduct sentiment and emotion analysis on logs in a more intuitive manner. We manually label logs using Parrot's emotion framework [41] and then employ automated tools to analyze log texts. Our goal is to explore the validity of directly applying sentiment analysis to log texts and evaluate the performance of state-of-the-art tools in this context. To address the objective nature of log texts, which poses a challenge for log sentiment and emotion analysis, we resort to issue reports to enhance the sentiments expressed in the logs.

[20]https://github.com/studiawan/pylogsentiment/tree/master/pylogsentiment

8.2 Sentiment and emotion analysis for issue reports

Various empirical studies have been done for issue report sentiment and emotion analysis. Ortu and his colleagues [37, 38, 40] investigated the emotions expressed in JIRA issue comments through manual analysis for six basic emotions from Parrot's framework. Their manual analysis involved examining whether issue comments conveyed emotions, measuring raters' agreement, assessing the influence of context on raters' agreement, investigating correlations between sentiments, emotions, and politeness, and exploring the relationship between sentiment metrics and problem resolution time. They also employed machine learning classifiers to identify expressions of love, joy, and sadness in issue comments. Mantyla et al. [35] calculated VAD values of different parts of JIRA issues (i.e., title, description, all comments, first comment, last comment). Besides, they explored VAD values' evolution during issue resolution and their potential correlation with defect resolution time. Jurado and Rodriguez [24] studied emotions in GitHub issues, and found that more than 80% of them do not contain obvious emotions.

Besides, various sentiment and emotion analysis tools have been developed or evaluated on issue report datasets. Sentiment polarity detection tools include SentiStrength-SE [23], SEntiMoji [11], and SentiSW [14]. SentiStrength-SE adapts the lexicon-based tool, SentiStrength, for SE field using JIRA issue comments. It is currently the most widely used sentient analysis tool for SE [39]. SentiSW is developed to detect the sentiments of issue comments at the entity level using supervised machine learning techniques, and it is reported that it outperforms SentiStrength-SE. SEntiMoji employs transfer learning for emotion detection in SE text based on emojis. Emotion Detection tools include DEVA [22], MarValous [21], and EmoTxT [7]. DEVA is a lexicon-based tool designed for analyzing JIRA issue comments. MarValous is tailored for Stack Overflow posts and issue reports, reportedly surpassing DEVA in performance. Both DEVA and MarValous employ the VAD theory to categorize emotional states into Excitation, Relaxation, Depression, and Stress. Specifically, DEVA calculates sentiment polarity and sentiment arousal based on dictionaries while MarValous exploits machine learning techniques. EmoTxt is also developed to identify emotions in both Stack Overflow posts and JIRA issue comments, leveraging Shaver et al.'s emotion framework [45] to identify six primary emotions.

Previous work mainly analyzed the sentiments and emotions of issue comments. However, we focus on issue descriptions because the logs we extracted come from issue descriptions. Providing context for logs, their sentiments and emotions can be associated with the extracted logs. More importantly, our analysis extends beyond just assessing the sentiments and emotions within issue reports. We associate the sentiments and emotions conveyed in issue descriptions with the logs and compare them with the results of directly analyzing logs. This approach significantly detects more sentiments and emotions. Furthermore, we investigate the potential utility of captured emotions for downstream tasks, such as critical log detection and anomaly detection within logs.

9 CONCLUSION

In this exploratory study, we find that over 8.0% of the log messages convey sentiments and emotions. Besides, we find that the issue reports referencing logs provide a deeper insight into the developers' sentiments and emotions. In particular, almost half of the issue reports convey emotions, with most of these emotions being related to *surprise*. We also find that such sentiments and emotions that are associated with logs could be potentially leveraged in subsequent analysis of logs.

Our findings highlight the challenges and importance of considering all relevant information that is communicated by developers through logs as well as other related artifacts during software engineering activities. Future work may explore the design of a more holistic approach to understanding and utilizing emotions and sentiments in logs to assist in software development and maintenance.

ACKNOWLEDGMENT

This work is partly supported by the National Natural Science Foundation of China (Grant Nos. 62302347 and 62141221).

REFERENCES

[1] Alan Agresti. 2012. *An introduction to categorical data analysis.* John Wiley & Sons. 26–27 pages.

[2] Alan Agresti. 2012. *An introduction to categorical data analysis.* John Wiley & Sons. 21–25 pages.

[3] Alan Agresti. 2012. *An introduction to categorical data analysis.* John Wiley & Sons. 34–38 pages.

[4] Khalid Ayedh Alharthi, Arshad Jhumka, Sheng Di, Franck Cappello, and Edward Chuah. 2021. Sentiment analysis based error detection for large-scale systems. In *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN).* 237–249.

[5] Olga Baysal, Michael W Godfrey, and Robin Cohen. 2009. A bug you like: A framework for automated assignment of bugs. In *2009 IEEE 17th International Conference on Program Comprehension.* IEEE, 297–298.

[6] Fabio Calefato, Filippo Lanubile, Federico Maiorano, and Nicole Novielli. 2018. Sentiment polarity detection for software development. In *Proceedings of the 40th International Conference on Software Engineering.* 128–128.

[7] Fabio Calefato, Filippo Lanubile, and Nicole Novielli. 2017. Emotxt: a toolkit for emotion recognition from text. In *Proceedings of the 2017 7th International Conference on Affective Computing and Intelligent Interaction Workshops and Demos.* 79–80.

[8] An Ran Chen, Tse-Hsun Chen, and Shaowei Wang. 2022. Pathidea: Improving information retrieval-based bug localization by re-constructing execution paths using logs. *IEEE Transactions on Software Engineering* 48, 8 (2022), 2905–2919.

[9] An Ran Chen, Tse-Hsun Chen, and Shaowei Wang. 2021. Demystifying the challenges and benefits of analyzing user-reported logs in bug reports. *Empirical Software Engineering* 26 (2021), 1–30.

[10] Jinfu Chen, Weiyi Shang, Ahmed E. Hassan, Yong Wang, and Jiangbin Lin. 2019. An experience report of generating load tests using log-recovered workloads at varying granularities of user behaviour. In *34th IEEE/ACM International Conference on Automated Software Engineering, ASE 2019, San Diego, CA, USA, November 11-15, 2019.* IEEE, 669–681.

[11] Zhenpeng Chen, Yanbin Cao, Xuan Lu, Qiaozhu Mei, and Xuanzhe Liu. 2019. SEntiMoji: an emoji-powered learning approach for sentiment analysis in software engineering. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering.* 841–852.

[12] Di Cui, Ting Liu, Yuanfang Cai, Qinghua Zheng, Qiong Feng, Wuxia Jin, Jiaqi Guo, and Yu Qu. 2019. Investigating the impact of multiple dependency structures on software defects. In *Proceedings of the 41st International Conference on Software Engineering, ICSE 2019, Montreal, QC, Canada, May 25-31, 2019.* IEEE / ACM, 584–595.

[13] Steven Davies and Marc Roper. 2014. What's in a bug report?. In *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement.* 1–10.

[14] Jin Ding, Hailong Sun, Xu Wang, and Xudong Liu. 2018. Entity-level sentiment analysis of issue comments. In *Proceedings of the 3rd International Workshop on Emotion Awareness in Software Engineering.* 7–13.

[15] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. 2017. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017.* ACM, 1285–1298.

[16] Qiang Fu, Jian-Guang Lou, Yi Wang, and Jiang Li. 2009. Execution anomaly detection in distributed systems through unstructured log analysis. In *ICDM 2009, The Ninth IEEE International Conference on Data Mining, Miami, Florida, USA, 6-9 December 2009.* 149–158.

[17] Qiang Fu, Jieming Zhu, Wenlu Hu, Jian-Guang Lou, Rui Ding, Qingwei Lin, Dongmei Zhang, and Tao Xie. 2014. Where do developers log? an empirical study on logging practices in industry. In *Companion Proceedings of the 36th International Conference on Software Engineering.* 24–33.

[18] Emitza Guzman, Luís Oliveira, Yves Steiner, Laura C. Wagner, and Martin Glinz. 2018. User Feedback in the App Store: A Cross-Cultural Study. In *IEEE/ACM International Conference on Software Engineering: Software Engineering in Society.* 13–22.

[19] Shilin He, Qingwei Lin, Jian-Guang Lou, Hongyu Zhang, Michael R. Lyu, and Dongmei Zhang. 2018. Identifying impactful service system problems via log analysis. In *Proceedings of the 2018 ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/SIGSOFT FSE 2018, Lake Buena Vista, FL, USA, November 04-09, 2018.* 60–70.

[20] Mia Mohammad Imran, Yashasvi Jain, Preetha Chatterjee, and Kostadin Damevski. 2022. Data augmentation for improving emotion recognition in software engineering communication. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering.* 1–13.

[21] Md Rakibul Islam, Md Kauser Ahmmed, and Minhaz F Zibran. 2019. MarValous: Machine learning based detection of emotions in the valence-arousal space in software engineering text. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing.* 1786–1793.

[22] Md Rakibul Islam and Minhaz F Zibran. 2018. DEVA: sensing emotions in the valence arousal space in software engineering text. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*. 1536–1543.

[23] Md Rakibul Islam and Minhaz F. Zibran. 2018. SentiStrength-SE: Exploiting domain specificity for improved sentiment analysis in software engineering text. *Journal of Systems and Software* 145 (2018), 125–146.

[24] Francisco Jurado and Pilar Rodriguez. 2015. Sentiment Analysis in monitoring software development processes: An exploratory case study on GitHub's project issues. *Journal of Systems and Software* 104 (2015), 82–89.

[25] Heng Li, Weiyi Shang, Bram Adams, Mohammed Sayagh, and Ahmed E Hassan. 2020. A qualitative study of the benefits and costs of logging from developers' perspectives. *IEEE Transactions on Software Engineering* 47, 12 (2020), 2858–2873.

[26] Heng Li, Weiyi Shang, and Ahmed E Hassan. 2017. Which log level should developers choose for a new logging statement? *Empirical Software Engineering* 22 (2017), 1684–1716.

[27] Yinglung Liang, Yanyong Zhang, Anand Sivasubramaniam, Ramendra K Sahoo, Jose Moreira, and Manish Gupta. 2005. Filtering failure logs for a bluegene/l prototype. In *2005 International Conference on Dependable Systems and Networks (DSN'05)*. IEEE, 476–485.

[28] Yinglung Liang, Yanyong Zhang, Hui Xiong, and Ramendra Sahoo. 2007. Failure prediction in ibm bluegene/l event logs. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*. IEEE, 583–588.

[29] Lizhi Liao, Jinfu Chen, Heng Li, Yi Zeng, Weiyi Shang, Jianmei Guo, Catalin Sporea, Andrei Toma, and Sarah Sajedi. 2020. Using black-box performance models to detect performance regressions under varying workloads: an empirical study. *Empirical Software Engineering* 25, 5 (2020), 4130–4160.

[30] Lizhi Liao, Jinfu Chen, Heng Li, Yi Zeng, Weiyi Shang, Catalin Sporea, Andrei Toma, and Sarah Sajedi. 2022. Locating performance regression root causes in the field operations of Web-Based Systems: An Experience Report. *IEEE Transactions on Software Engineering* 48, 12 (2022), 4986–5006.

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

[32] Bin Lin, Fiorella Zampetti, Gabriele Bavota, Massimiliano Di Penta, Michele Lanza, and Rocco Oliveto. 2018. Sentiment analysis for software engineering: How far can we go?. In *Proceedings of the 40th International Conference on Software Engineering*. 94–104.

[33] Jiahao Liu, Jun Zeng, Xiang Wang, Kaihang Ji, and Zhenkai Liang. 2022. Tell: log level suggestions via modeling multi-level code block information. In *Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*. 27–38.

[34] Jian-Guang Lou, Qiang Fu, Shenqi Yang, Ye Xu, and Jiang Li. 2010. Mining invariants from console logs for system problem detection. In *2010 USENIX Annual Technical Conference (USENIX ATC 10)*.

[35] Mika Mäntylä, Bram Adams, Giuseppe Destefanis, Daniel Graziotin, and Marco Ortu. 2016. Mining valence, arousal, and dominance: possibilities for detecting burnout and productivity?. In *Proceedings of the 13th international conference on mining software repositories*. 247–258.

[36] Andrés Montoyo, Patricio Martínez-Barco, and Alexandra Balahur. 2012. Subjectivity and sentiment analysis: An overview of the current state of the area and envisaged developments. *Decision Support Systems* 53, 4 (2012), 675–679.

[37] Alessandro Murgia, Marco Ortu, Parastou Tourani, Bram Adams, and Serge Demeyer. 2018. An exploratory qualitative and quantitative analysis of emotions in issue report comments of open source systems. *Empirical Software Engineering* 23 (2018), 521–564.

[38] Alessandro Murgia, Parastou Tourani, Bram Adams, and Marco Ortu. 2014. Do developers feel emotions? an exploratory analysis of emotions in software artifacts. In *Proceedings of the 11th Working Conference on Mining Software Repositories*. 262–271.

[39] Martin Obaidi, Lukas Nagel, Alexander Specht, and Jil Klünder. 2022. Sentiment analysis tools in software engineering: A systematic mapping study. *Information and Software Technology* 151 (2022), 107018.

[40] Marco Ortu, Bram Adams, Giuseppe Destefanis, Parastou Tourani, Michele Marchesi, and Roberto Tonelli. 2015. Are bullies more productive? Empirical study of affectiveness vs. issue fixing time. In *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories*. 303–313.

[41] W Gerrod Parrott. 2001. *Emotions in social psychology: Essential readings.* Psychology Press.

[42] Mary Sánchez-Gordón and Ricardo Colomo-Palacios. 2019. Taking the emotional pulse of software engineering—A systematic literature review of empirical studies. *Information and Software Technology* 115 (2019), 23–43.

[43] Kevin Schmidt, Chris Phillips, and Anton Chuvakin. 2012. *Logging and log management: the authoritative guide to understanding the concepts surrounding logging and log management.* Newnes.

[44] Robert W Schrauf and Julia Sanchez. 2004. The preponderance of negative emotion words in the emotion lexicon: A cross-generational and cross-linguistic study. *Journal of multilingual and multicultural development* 25, 2-3 (2004), 266–284.

[45] Phillip Shaver, Judith Schwartz, Donald Kirson, and Cary O'connor. 1987. Emotion knowledge: further exploration of a prototype approach. *Journal of Personality and Social Psychology* 52, 6 (1987), 1061.

[46] Julius Sim and Chris C Wright. 2005. The kappa statistic in reliability studies: use, interpretation, and sample size requirements. *Physical therapy* 85, 3 (2005), 257–268.

[47] Kamonphop Srisopha, Chukiat Phonsom, Keng Lin, and Barry Boehm. 2019. Same app, different countries: A preliminary user reviews study on most downloaded ios apps. In *IEEE International Conference on Software Maintenance and Evolution*. 76–80.

[48] Carlo Strapparava, Alessandro Valitutti, et al. 2004. Wordnet affect: an affective extension of wordnet.. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, Vol. 4. 1083–1086.

[49] Hudan Studiawan, Ferdous Sohel, and Christian Payne. 2020. Anomaly detection in operating system logs with deep learning-based sentiment analysis. *IEEE Transactions on Dependable and Secure Computing* 18, 5 (2020), 2136–2148.

[50] Mark D Syer, Zhen Ming Jiang, Meiyappan Nagappan, Ahmed E Hassan, Mohamed Nasser, and Parminder Flora. 2013. Leveraging performance counters and execution logs to diagnose memory-related performance issues. In *2013 IEEE International Conference on Software Maintenance*. IEEE, 110–119.

[51] Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. 2010. Sentiment strength detection in short informal text. *Journal of the American Society for Information Science and Technology* 61, 12 (2010), 2544–2558.

[52] Junjielong Xu, Ruichun Yang, Yintong Huo, Chengyu Zhang, and Pinjia He. 2024. DivLog: Log parsing with prompt enhanced in-context learning. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. 1–12.

[53] Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael I Jordan. 2009. Detecting large-scale system problems by mining console logs. In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*. 117–132.

[54] Geunseok Yang, Seungsuk Baek, Jung-Won Lee, and Byungjeong Lee. 2017. Analyzing emotion words to predict severity of software bugs: A case study of open source projects. In *Proceedings of the Symposium on Applied Computing*. 1280–1287.

[55] Ding Yuan, Haohui Mai, Weiwei Xiong, Lin Tan, Yuanyuan Zhou, and Shankar Pasupathy. 2010. SherLog: error diagnosis by connecting clues from run-time logs. In *Proceedings of the 15th International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2010, Pittsburgh, Pennsylvania, USA, March 13-17, 2010*. ACM, 143–154.

[56] Ding Yuan, Soyeon Park, and Yuanyuan Zhou. 2012. Characterizing logging practices in open-source software. In *2012 34th international conference on software engineering (ICSE)*. IEEE, 102–112.

[57] Ding Yuan, Jing Zheng, Soyeon Park, Yuanyuan Zhou, and Stefan Savage. 2012. Improving software diagnosability via log enhancement. *ACM Transactions on Computer Systems* 30, 1 (2012), 4:1–4:28.

[58] Di Zhang, Dong Dai, Runzhou Han, and Mai Zheng. 2021. Sentilog: Anomaly detecting on parallel file systems via log-based sentiment analysis. In *Proceedings of the 13th ACM workshop on hot topics in storage and file systems*. 86–93.

[59] Xu Zhang, Yong Xu, Qingwei Lin, Bo Qiao, Hongyu Zhang, Yingnong Dang, Chunyu Xie, Xinsheng Yang, Qian Cheng, Ze Li, et al. 2019. Robust log-based anomaly detection on unstable log data. In *Proceedings of the 2019 27th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering*. 807–817.

[60] Xiang Zhou, Xin Peng, Tao Xie, Jun Sun, Chao Ji, Dewei Liu, Qilin Xiang, and Chuan He. 2019. Latent error prediction and fault localization for microservice applications by learning from system trace logs. In *Proceedings of the ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/SIGSOFT FSE 2019, Tallinn, Estonia, August 26-30, 2019*. ACM, 683–694.

[61] Jieming Zhu, Shilin He, Pinjia He, Jinyang Liu, and Michael R. Lyu. 2023. Loghub: A large collection of system log datasets for ai-driven log analytics. In *IEEE International Symposium on Software Reliability Engineering (ISSRE)*.

[62] Jieming Zhu, Shilin He, Jinyang Liu, Pinjia He, Qi Xie, Zibin Zheng, and Michael R Lyu. 2019. Tools and benchmarks for automated log parsing. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*. IEEE, 121–130.

[63] Kelly H Zou, Julia R Fielding, Stuart G Silverman, and Clare Tempany. 2003. Hypothesis testing. I: Proportions. *Radiology* 226, 3 (2003), 609–613.