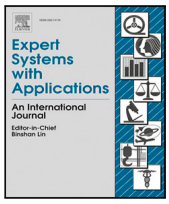
Expert Systems With Applications 261 (2025) 125524

Contents lists available at ScienceDirect

Expert Systems With Applications

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



# Leveraging Large Language Model ChatGPT for enhanced understanding of end-user emotions in social media feedbacks

Nek Dil Khan [a], Javed Ali Khan [b,*], Jianqiang Li [a], Tahir Ullah [c], Qing Zhao [a]

[a] Faculty of Information Technology, Beijing University of Technology, Beijing, 100124, China
[b] Department of Computer Science, School of Physics, Engineering and Computer Science, University of Hertfordshire, Hatfield, AL10 9AB, UK
[c] School of Computer Science, Beijing Institute of Technology, Beijing, 100124, China

## ARTICLE INFO

Dataset link: https://github.com/nekdil566/Experimental-work-and-datasets

*Keywords:*
Data-driven requirements engineering
Generative Artificial Intelligence
Sentiment Analysis
Large Language Models
Software evolution
ChatGPT

## ABSTRACT

For software evolution, user feedback has become a meaningful way to improve applications. Recent studies show a significant increase in analyzing end-user feedback from various social media platforms for software evolution. However, less attention has been given to the end-user feedback for low-rating software applications. Also, such approaches are developed mainly on the understanding of human annotators who might have subconsciously tried for a second guess, questioning the validity of the methods. For this purpose, we proposed an approach that analyzes end-user feedback for low-rating applications to identify the end-user opinion types associated with negative reviews (an issue or bug). Also, we utilized Generative Artificial Intelligence (AI), i.e., ChatGPT, as an annotator and negotiator when preparing a truth set for the deep learning (DL) classifiers to identify end-user emotion. For the proposed approach, we first used the ChatGPT Application Programming Interface (API) to identify negative end-user feedback by processing 71853 reviews collected from 45 apps in the Amazon store. Next, a novel grounded theory is developed by manually processing end-user negative feedback to identify frequently associated emotion types, including anger, confusion, disgust, distrust, disappointment, fear, frustration, and sadness. Next, two datasets were developed, one with human annotators using a content analysis approach and the other using ChatGPT API with the identified emotion types. Next, another round is conducted with ChatGPT to negotiate over the conflicts with the human-annotated dataset, resulting in a conflict-free emotion detection dataset. Finally, various DL classifiers, including LSTM, BILSTM, CNN, RNN, GRU, BiGRU and BiRNN, are employed to identify their efficacy in detecting end-users emotions by preprocessing the input data, applying feature engineering, balancing the data set, and then training and testing them using a cross-validation approach. We obtained an average accuracy of 94%, 94%, 93%, 92%, 91%, 91%, and 85%, with LSTM, BILSTM, RNN, CNN, GRU, BiGRU and BiRNN, respectively, showing improved results with the truth set curated with human and ChatGPT. Using ChatGPT as an annotator and negotiator can help automate and validate the annotation process, resulting in better DL performances.

## 1. Introduction

Recently, there has been a shift in software development, where user needs, feature enhancement requests, and issue reports are not restricted to in-house users only. End-users utilize various popular social media platforms, such as Twitter, Amazon, Google and Apple Play stores, Reddit, etc., to submit feedback (Khan, Khan, Li, Ullah, Alwadain, Yasin, & Zhao, 2024; Khan, Liu, Wen, & Ali, 2019). Researchers have highlighted the significance of end-user feedback in the software evolution process (Carreño & Winbladh, 2013; Khan, Yasin, et al., 2022). Also, user feedback plays a crucial role in requirement engineering (RE) as it provides valuable insights and outlooks from the software users (Kifetew et al., 2021). This feedback helps developers and app owners understand the users' problems, needs, and expectations (Ali Khan, Liu, Wen, & Ali, 2020; Khan, Khan, Li, Ullah, & Zhao, 2024a; Mezouar, Zhang, & Zou, 2018). Additionally, user feedback helps identify bugs and potential issues in the software, enabling developers to address them promptly (Mezouar et al., 2018; Ullah, Khan, Khan, Yasin, & Arshad, 2023), if included in the software evolution process. It is pivotal for improving the existing functionalities and end-user trust in market-based software applications. In contrast, end-users get frustrated by software application quality and incomplete features if ignored, resulting in lower ratings and ultimately

* Corresponding author.
*E-mail addresses:* nekdilkhan@emails.bjut.edu.cn (N.D. Khan), j.a.khan@herts.ac.uk (J.A. Khan), lijianqiang@bjut.edu.cn (J. Li), tahirullah7710@bit.edu.cn (T. Ullah), zhaoqing@bjut.edu.cn (Q. Zhao).

https://doi.org/10.1016/j.eswa.2024.125524
Received 10 November 2023; Received in revised form 30 September 2024; Accepted 5 October 2024
Available online 10 October 2024

uninstalling them. Therefore, User feedback involvement is essential to software development as they often contain insights such as introduced bugs and feature requests (Alvertis et al., 2016). Still, it is essential to analyze and understand the rationale behind user decisions, opinions, and beliefs (Khan, Liu, & Wen, 2020).

Furthermore, end-user feedback is not limited to app stores, i.e., Apple and Google Play stores. End-Users express their user experiences, opinions, and emotions and report bugs across various platforms, such as the Amazon Software App (ASA) store, Twitter, Reddit, and other technical forums, such as Issue tracking systems, DevOps, etc. Jeong and Lee (2022). While the variety of feedback sources is comprehensive, app store reviews hold a special place due to their exposure and rich insights (Gao, Zeng, Lyu, & King, 2018; Villarroel, Bavota, Russo, Oliveto, & Di Penta, 2016). It is recommended to develop approaches that emphasize the need to harness all potential feedback channels and include them in the software evolution process to give a voice to the users (Khan, Liu, et al., 2019). In light of this, Crowd-Based Requirements Engineering (CrowdRE), a technique aggregating user requirements from widespread audiences, is seeing increasing adoption (Groen et al., 2017). Such techniques provide developers with the insights needed to align their software applications, particularly with user expectations (Zimmermann et al., 2010). For this paper, we have explored the end-user feedback on the ASA store for improving the existing software functionalities by focusing on low-ranked software applications, which are rarely utilized compared to the app store.

Many research approaches have recently been proposed for analyzing end-user feedback from these online social media platforms. However, limited work has been conducted on understanding the end-user opinions and emotions associated with the end-user feedback. Sentiment analysis and opinion mining related to user reviews in software applications have become crucial in understanding customer perceptions, better knowing software faults, and helping achieve overall user satisfaction (Nurrohmat & Azhari, 2019). Also, in the RE literature, there have been several approaches to identifying sentiment (positive, negative, or neutral) associated with the identified new features for high-rating and most popular software apps (Guzman & Maalej, 2014). Previously, identifying issues or bugs and their associated emotions for low-rated software applications was ignored, resulting in removing the app from the online marketplace, mainly not listening to the grudges of the end-users frequently reported on social media platforms. To date, according to our knowledge, not many research approaches have been proposed to identify the emotions associated with end-user feedback on social media platforms, such as the ASA store. Additionally, although the existing research approaches offer valuable feedback insights for software evolution, they often lean heavily on human annotators, potentially leading to bias (Khan, Xie, Liu, & Wen, 2019; Lim, Henriksson, & Zdravkovic, 2021). Also, the manual annotation for the supervised approach is considered time-consuming and challenging for human annotators, particularly when there are several classification classes (Haering, Stanik, & Maalej, 2021; Khan, Yasin, et al., 2022; Mezouar et al., 2018; Ullah et al., 2023).

Although human annotators have traditionally played a crucial role in annotating datasets for machine learning experiments; Still, they are inherently subject to personal biases and inconsistencies, particularly when interpreting complex emotional nuances from large volumes of data. Recently, researchers have started using Large Language Models (LLMs), particularly ChatGPT, to human annotators (Fischer, Luczak-Roesch, & Karl, 2023; Guo et al., 2023). Also, ChatGPT has been employed by researchers for various software engineering-related activities, including code generation, solving programming bugs, and code completion (Beganovic, Jaber, & Abd Almisreb, 2023). Also, ChatGPT has been tested to generate requirements for software applications (Bencheikh & Höglund, 2023). We aim that employing LLMs can offer a compelling alternative for annotating end-user feedback. By leveraging the consistent and scalable analysis capabilities of ChatGPT, we aim to minimize these human biases, thereby enhancing the reliability and objectivity of the proposed approach. We believe that the proposed approach of employing ChatGPT will not only support the rapid processing of extensive data sets but also improve the accuracy of emotion detection by harnessing the power of ChatGPT.

To fill this gap, we proposed a novel research approach that analyzes end-user feedback for comparatively low-ranked software apps in the ASA store by utilizing the advantages of the LLMs, such as ChatGPT, aiming to automate the process and minimize human biases. LLMs are loaded with enormous and manifold training samples and demonstrate an improved ability to replicate human linguistic capabilities. Researchers are applying LLMs to various software engineering activities, such as software testing, natural language to code, and code summarization (Hou, Zhao, Liu, Yang, Wang, Li, Luo, Lo, Grundy, & Wang, 2023). Therefore, in the proposed research approach, we utilized ChatGPT to annotate the end-user feedback for the DL experiments and compare its results to the manually annotated dataset. To highlight, the proposed approach first uses the ChatGPT Application Programming Interface (API) to identify the negative end-user feedback and filter out positive and neutral end-user comments. For this paper, we are interested in analyzing negative end-user feedback for the low-raked software application and identifying associated user emotion types to better understand their grudges with the software applications. A coding guideline is developed to capture the end-user emotion type by manually analyzing a sample of feedback from the ASA store. The most commonly occurring end-user emotion types selected for the proposed study are anger, confusion, disgust, distrust, disappointment, fear, frustration, and sadness. Later, using the content analysis approach, two datasets are created using manual annotation by human coders and ChatGPT API. Two datasets are created, one with human annotators utilizing a content analysis method and the other using ChatGPT API with the identified emotion types determined using grounded theory. To resolve disagreements between the human and ChatGPT annotated datasets, we conducted a second round with GhatGPT using a command prompt to negotiate on the conflicting end-user annotation, resulting in a conflict-free emotion detection dataset, which is used as the truth set for the various DL classifiers. Finally, various DL algorithms are applied to identify their performance in classifying end-user reviews to the emotion types by supplying an emotion detection dataset. The proposed study's key contributions are:

- Developed a novel research dataset comparing negative end-user comments from the ASA store representing end-user grudges and issues with the software applications.
- Developed a unique, grounded theory representing frequently occurring end-user emotions associated with negative reviews.
- Utilized ChatGPT API to construct an alternative emotion dataset for automatically identifying user emotions.
- ChatGPT is utilized as a negotiator to develop a single truth set by removing conflict between the human annotators and ChatGPT
- Improved the accuracy of DL algorithms by optimizing their features on eight class classification problems.
- Proposing an automated approach that utilized ChatGPT in the loop to overcome the human bias and manual annotation challenges.

The paper is structured in the following method: Section 2 presents the literature review. Section 3 outlines the proposed research methodology and research questions. Section 4 focuses on the collection of research data. Section 5 focuses on the identification of end-user feedback sentiments using ChatGPT. Section 6 presents a novel grounded theory for identifying end-user emotions. Section 7 analyzes the end-user feedback to develop a truth set. Section 8 discusses the automated classification of end-user emotions. Section IX presents the results of the proposed approach. Section X explains a discussion of the findings. Finally, section XI concludes the paper and discusses and highlights future research.

## 2. Literature review

The domain of software engineering is extensive and ever-evolving. As we explore its complications, it becomes clear that integrating new technologies and methods shapes its future. This literature review aims to deliver some of these improvements and their importance.

### 2.1. Generative AI in software engineering:

Software engineering is a constantly evolving field, driven by the emergence of new technologies (Han, Han, et al., 2021). Generative Artificial Intelligence (GAI), a transformative technology (Ebert & Louridas, 2023), is balanced to revolutionize various aspects of software engineering (Acypreste & Paraná, 2022). GAI can generate novel content or solutions based on its extensive data analysis and learning (Deng & Chen, 2021). Well-known examples like ChatGPT illustrate the growing importance of GAI, offering new possibilities in research in engineering disciplines (Rudolph, Tan, & Tan, 2023). GAI can be instrumental in software engineering tasks such as requirement definition, issue identification, and sentiment analysis (Manole, 2021). For instance, AI algorithms can improve requirement engineering efficiency and accuracy by processing natural language inputs (Russo, 2023; Wang & Liu, 2014). Furthermore, the integration of generative AI into software engineering practices holds immense promise, streamlining processes and enhancing outcomes (Manole, 2021; Wang & Liu, 2014).

### 2.2. User feedback as a goldmine for software enhancement

User feedback has emerged as a valuable resource in software engineering, offering deep insights into software quality and user satisfaction. This literature review explores the burgeoning area of feedback and user review mining, shedding light on their pivotal role in enhancing software products. Researchers have increasingly recognized the significance of mining user-generated content, such as app reviews, tweets, and customer feedback, in refining software (Guzman & Maalej, 2014; Maalej, Kurtanović, Nabil, & Stanik, 2016; Mezouar et al., 2018; Ullah et al., 2023), to glean actionable intelligence for software improvement. Lin et al. (2022) conducted a systematic literature review on opinion mining studies from mobile app store user reviews, highlighting the importance of sentiment analysis and user opinion experience in software development. In a related study, Lin et al. (2022) highlighted how developers can mine user feedback from mobile app reviews to make informed decisions for product enhancements. Hou, Yannou, Leroy, and Poirson (2019) proposed a summarization model for mining customer product reviews, emphasizing the multifaceted aspects of feedback analysis in product development. This growing body of research highlights the transformative potential of user feedback, placing it as a goldmine for software enhancement. This type of user feedback benefits software engineers by providing insights into what needs fixing or improvement. Furthermore, Social media is also used to understand why decisions are made in software development and to collect additional user information requires (Khan et al., 2020; Khan, Xie, et al., 2019; Kurtanović & Maalej, 2018). This data helps software companies decide what new features to add or what issues to fix (Nayebi, Dicke, Ittyipe, Carlson, & Ruhe, 2018). Additionally, these papers underscore user feedback's significance as a critical resource for improving software engineering quality.

### 2.3. Sentiment analysis in software feedback

Sentiment Analysis in Software Feedback, a critical research area, bridges technology and user experience by automating the extraction of positive or negative sentiment from software users' textual feedback. Multiple research contributions have inspired this field's methodologies, applications, and importance. Ligthart, Catal, and Tekinerdogan (2021) systematic study highlights the importance of synthesizing existing knowledge through tertiary reviews in sentiment analysis. Wankhade, Rao, and Kulkarni (2022) provides a comprehensive overview of sentiment analysis methods and their evolving relevance in various domains. Additionally, Rodríguez-Ibánez, Casánez-Ventura, Castejón-Mateos, and Cuenca-Jiménez (2023) work showcases sentiment analysis's relevance in understanding consumer preferences and improving product features. Furthermore, Sentiment analysis, a subset of natural language processing, has garnered increasing interest, particularly in deciphering opinions and emotions from textual data (Liu, 2012). This is crucial for collecting feedback during software development. Studies by Dąbrowski, Letier, Perini, and Susi (2022) and Lee (2020) offer methods and frameworks for leveraging natural language and sentiment analysis to extract valuable insights from app reviews. Kumar, Desai, and Majumdar (2016) discusses the importance of sentiment analysis in various contexts, including addressing challenges like detecting sarcasm in customer reviews. Overall, the literature review on Sentiment Analysis in Software Feedback underscores the multidisciplinary approach in this field, highlighting its growing significance in enhancing user experiences and decision-making in software development.

### 2.4. Comparison with the state-of-the-art

The proposed approach complements the above-mentioned approaches to using end-user feedback for software evolution but with different perspectives. It focuses on end-user feedback for comparatively low-rating applications with the intention of improving software quality and user satisfaction. Unlike in the literature, the focus has been given to high-rated applications leaving a research gap of exploring end-user feedback for low-rating apps. Also, the proposed approach aims to understand various end-user emotion types, including anger, disgust, disappointment, distrust, confusion, fear, frustration, and sadness associated with the negative reviews submitted by end-users about software applications in better understanding their grudges. In contrast, the previous research approaches are limited in identifying the associated sentiments with the end-user reviews, i.e., positive, negative, and neutral. Moreover, the proposed approach can be extended by first identifying frequently occurring issues by implementing NLP and Part-of-speech tagging in low-ranking software applications better to understand the severity of issues and end-user grudges. Unlike previous research approaches, we integrated Generative AI (ChatGPT) to automate the annotation process and minimize human biases. Moreover, we fine-tuned the state-of-the-art DL classifiers to classify end-user feedback into various emotion types. With a deeper understanding of user emotions, this methodology provides insights to software developers and vendors to enhance application performance and address user concerns more effectively by intertwining the proposed approach with software evolution. Additionally, A detailed comparative study of the proposed methodology with state-of-the-art approaches is elaborated in Table 1, highlighting each approach's main focus, border methods used, key findings and proposed approach contribution to the existing literature.

## 3. Proposed methodology

This section discusses the proposed research methodology, which contains two steps. First, we outline the research questions we seek to address using the proposed research methodology. Second, we elaborate on the proposed DL-based approach, which utilizes generative AI (ChatGPT) in the loop by trying to semi-automate the proposed approach that identifies end-user opinions associated with the end-user reviews that represent possible issues or bugs in end-user comments on Amazon store. The steps are elaborated below.

**Table 1**
Comparative analysis of existing research on user feedback and sentiment analysis in software engineering.

| Study | Focus | Methods used | Key findings | Contribution of current work |
|---|---|---|---|---|
| Lin et al. (2022) | Sentiment analysis in high-rating app reviews | Machine learning techniques | Identified positive correlations between user ratings and app features | understanding negative reviews with end-user emotions |
| Hou et al. (2019) | Mining user feedback for software evolution | Automated tools for text analysis | Provided insights into user satisfaction and product features | Applied LLMs and DL to analyze emotional depth in feedback |
| Guzman and Maalej (2014) | User feedback in app stores | Text mining and sentiment analysis | Demonstrated how user feedback can guide software enhancements | Focuses on nuanced feedback from low-rated applications |
| Khan et al. (2020) | Analyzing user feedback for software development | Qualitative analysis and surveys | Showed the importance of user feedback in the software development lifecycle | Proposed an automated approach for utilizing users feedback for software evolution |
| Rodríguez-Ibánez et al. (2023) | Sentiment analysis in customer feedback | Systematic literature review | Highlighted the evolution of sentiment analysis methods and their applications in software engineering | Utilizes ChatGPT and DL for improved opinion & emotional analysis accuracy |
| Ligthart et al. (2021) | Sentiment analysis methodologies | Tertiary review | Synthesized existing knowledge on sentiment analysis applications | Improved sentiment analysis approach by identifying various emotion types |
| Wankhade et al. (2022) | Evolution of sentiment analysis methods | Comprehensive survey | Reviewed sentiment analysis methods across domains | Applied fine-tuned DL algorithms for better precision in sentiment and opinion classification |
| Dąbrowski, Letier, Perini, and Susi (2020) | Leveraging NLP for app reviews | Natural language processing | Developed frameworks to extract insights from app reviews | leverage generative AI for extracting useful information for low-ranked apps |
| Kumar et al. (2016) | Challenges in sentiment analysis | Opinion mining | Discussed detecting sarcasm and complex sentiments in reviews | Enhances sentiment detection accuracy with hybrid AI models |
| Nayebi et al. (2018) | The role of user feedback in agile software development | Empirical study | Emphasized timely incorporation of user feedback in sprints | proposed an automated approach that can be integrated into the agile methodology for improving app quality |
| Proposed Approach | Analyzing negative reviews in low-rated apps for software evolution | Generative AI(ChatGPT) and DL classifiers | Enhanced detection of various emotional types intertwined with end-user feedback | Proposed a software evolution approach by utilizing ChatGPT and DL classifiers to analyze and categorize emotions associated with negative reviews for low-ranked software applications |

### *3.1. Proposed research questions*

This paper aims to uncover issues or bugs from end-user negative feedback and their associated emotions like anger, confusion, disgust, disappointment, distrust, fear, frustration, and sadness regarding low-ranking software applications. We hope to gain insights into alternate solutions and enhanced performance for low-ranked software applications by evaluating software apps in the Amazon software store. The main goal of the research is to automatically recognize issues or bugs from end-user negative feedback associated with the emotions expressed in negative user reviews of low-rated software apps and automate this understanding using generative AI (ChatGPT) and DL algorithms to improve software quality. The study aims to compare the effectiveness of the automated approach to manual methods of identifying issues, bugs, and emotions in low-ranked software applications. We formulate the following research questions for this purpose:

**RQ1. How do end-users express their emotions on social media when discussing bugs in low-rated software apps?**

**RQ2. What types of end-user emotions can be identified in the crowd-user reviews?**

**RQ3. Can a Large Language Model (ChatGPT) be used as an annotator and negotiator in constructing a truth set for DL classifiers?**

**RQ4. How do various DL classifiers identify end-user emotion types from a dataset curated with humans and ChatGPT?**

For the proposed approach, we extracted 71,853end-users' comments across 45 low-rated apps in the Amazon store, focusing on how users express their emotions when submitting reviews. **For RQ1**, we randomly selected 11,800 end-user comments from the dataset. We conducted a detailed manual analysis of how users express their emotions when submitting reviews to recover associated end-user emotion and their types, if any when registering their grudges against software apps. This led to the development of a novel grounded theory that captures the end-user emotion types and associated examples against low-rated software applications. It would help software developers identify the severity of end-users grievances with the software app or its particular feature. After researching the potential presence of user emotion types in the end-user comments, we employed the grounded theory Corbin and Strauss (2008) approach to recover the various end-user emotions intertwined with their submitted feedback to answer **For RQ2**. The emotion types identified in the end-user feedback for the low-rating software applications include anger, confusion, disgust, distrust, disappointment, fear, frustration, and sadness, commonly expressed by the end-users when submitting feedback in the Amazon store for the low-ranked apps. This mapping directly responds to RQ2 by describing the emotions identifiable from user reviews, further enriching the analysis, and precisely categorizing emotional feedback for improved software quality and end-user satisfaction. Later, the developed novel grounded theory and content analysis approach (Maalej & Robillard, 2013; Neuendorf, 2017) is used to annotate the 11,800 comments in the dataset. This emotional mapping is a valuable resource for software vendors and developers to understand the issues plaguing end-users when dealing with low-rated software. Also, it is mentioned in the literature (Haering et al., 2021; Khan et al., 2020; Khan, Liu, et al., 2019) that feedback annotation is a laborious and time-consuming activity for proposing automated approaches. For this purpose, to

answer **RQ3**, we experimented with ChatGPT to annotate end-user feedback with the developed coding guideline aiming to automate the proposed approach. Also, we are interested in exploring ChatGPT as a negotiator in removing conflicts between the human coders and Chatgpt, resulting in a conflict-free truth set for the DL classifiers. Finally, **RQ4** aimed to evaluate and compare the performance of various DL classifiers, specifically CNN, LSTM, BILSTM, GRU, BiGRU, and RNN, for automatically identifying these end-user emotions by following various DL steps. To find the answer, we planned to experiment with human and machine-annotated data sets and provide a comparative study to minimize human annotators' involvement in the proposed approach.

### 3.2. Research methodology

The proposed research methodology to identify the end-user emotions in the Amazon store is shown in Fig. 1. The research approach comprises seven main steps: In the first step, we compiled a unique research dataset with user reviews of low-rated software apps from the Amazon software store. We chose low-rated software apps intending to find and capture end-users emotions when submitting feedback against software applications facing issues while using. Also, in the literature, preferences are given to the user feedback for high-rating software applications, ignoring the low-rating applications. It would help understand why certain apps are getting low ratings and allow software vendors to listen to the usually ignored users. By manually skimming the end-user feedback, it is recovered that users submit positive and natural feedback along with negative feedback. For the scope of this paper, we aim to process and analyze negative feedback against low-ranked software apps and explore their application for improving end-user satisfaction by possibly incorporating the proposed approach in the software evolution process. In Step 2 of the proposed methodology, we harness the power of LLM (ChatGPT) to classify end-user reviews into positive, negative, and neutral sentiments using ChatGPT API. We hypothesize that negative sentiments associated with the review best present end-user grudges against the software apps, leading to the identification of various associated emotions. In the next step, a coding guideline is developed using a grounded theory by critically analyzing a random sample of the negative feedback identified in the previous step. As a result, eight emotion types were identified that are associated with the end-user feedback: anger, confusion, disgust, distrust, disappointment, fear, frustration, and sadness, which end-users frequently use when submitting their negative feedback.

In software engineering literature, manually annotating end-user feedback to make it parsable for DL classifiers is challenging and time-consuming (Khan, Liu, et al., 2019). Therefore, to automate the process and minimize the manual challenges of ML/DL-based approaches, we utilized the ChatGPT API to annotate end-user feedback with identified emotion types. This approach speeds up the annotation process and reduces potential human biases. To validate the results generated by ChatGPT, we manually coded the same dataset using a content analysis approach with human coders. Subsequently, we developed a grounded theory to identify common emotion categories associated with negative feedback, ensuring that our annotation process captures the relevant end-user sentiments. In step 5, a second round of negotiations was conducted using ChatGPT to resolve differences between human annotations and the ChatGPT-generated annotations, resulting in a conflict-free emotion detection dataset. For this purpose, we use the ChatGPT prompt to reason with the ChatGPT on selecting particular end-user emotion types compared to human emotion types, aiming to construct a single truth set for the DL classifiers. Finally, various DL classifiers were applied to evaluate their effectiveness in detecting end-user emotions, demonstrating the robustness of the proposed method. For each experiment, text preprocessing, data balancing, hyperparameter tuning, and cross-validation were employed to optimize the classification of end-user feedback into various emotion types. The proposed approach demonstrates the use of ChatGPT as an annotator and negotiator to validate the manual human annotation process for better and more generalized classification results. The following sections describe the proposed research approach in each step, starting with the research dataset, sentiment analysis with ChatGPT, the grounded theory for identifying emotion types, the annotation process (Manual & ChatGPT), and DL experiments and results.

## 4. Research data collection

This section details the dataset collection process, which is crucial for the methodological framework, i.e., grounded theory & annual annotation processes, and experimental validation of the proposed approach in identifying the efficacy of various DL classifiers when classifying end-user feedback into various emotion types. The dataset comprises end-user feedback on comparatively low-ranked software applications, which we argue is essential for analyzing and possibly improving low-rated apps by identifying associated emotions with end-user feedback. In the literature, it is evident that end-user feedback can be a valuable source for improving app performance and listening to alternative users for software evolution by employing various ML and DL approaches (Khan, Liu, et al., 2019).

To run the research approach, we were required to collect a dataset containing end-user feedback about software applications. To align with the proposed approach goals, we selected software applications in the Amazon Software App (ASA) stores that received low ratings, i.e., less than or equal to 3 stars. With an extensive range of 824,220 apps, the Amazon Appstore is well known as a leading global mobile app provider. A total of 174,237 app publishers contributed to the release of apps on the Amazon App Store, while the user community rated a significant percentage of the platform's 236,550 apps (AG, 2023). For this purpose, We selected 45 applications from 10 categories in the ASA store, ensuring a diverse range of applications for the analysis. The reasons for choosing low-rating software apps are to understand why these applications are getting a low rating, understand end-users emotions when submitting feedback against these apps, and identify various emotion types to better understand user grudges against the software apps. The details of each software application selected and its corresponding category are detailed in Table 2.

To collect the user reviews, we utilized the Instant Data Scraper [1] extension, which allows for efficient and systematic extraction of data from web pages. We gathered a substantial dataset of user reviews and their associated star ratings using this tool. 71853 end-user reviews are collected across 45 software applications from 10 different categories of the ASA store. These were selected based on manually identifying reported issues and associated end-user opinions for low-ranked software applications. The chosen categories represent a diverse range of software applications, allowing for a comprehensive analysis of crowd-user feedback across various fields. The selection of multiple categories also helps to ensure the generalizability of the findings, as it allows for capturing a broad range of end-user experiences and opinions, providing a more holistic experience of the issues and emotions expressed by crowd-users. The categories are selected based on the required number of end-user reviews against particular software applications. We selected software apps that contain a minimum of 400 reviews in the app. The process is performed manually by the first two authors of the papers. Finding software apps and categories in the ASA store was challenging because users do not know how to post many reviews against software applications containing issues or missing functionalities (Ullah et al., 2023). Therefore, we put efforts into the proposed approach to explore end-user reviews for low-rating software applications, aiming to understand the reasons and end-user emotions

[1] https://chrome.google.com/webstore/detail/instant-data-scraper/ofaokhiedipichpaobibbnahnkdoiiah

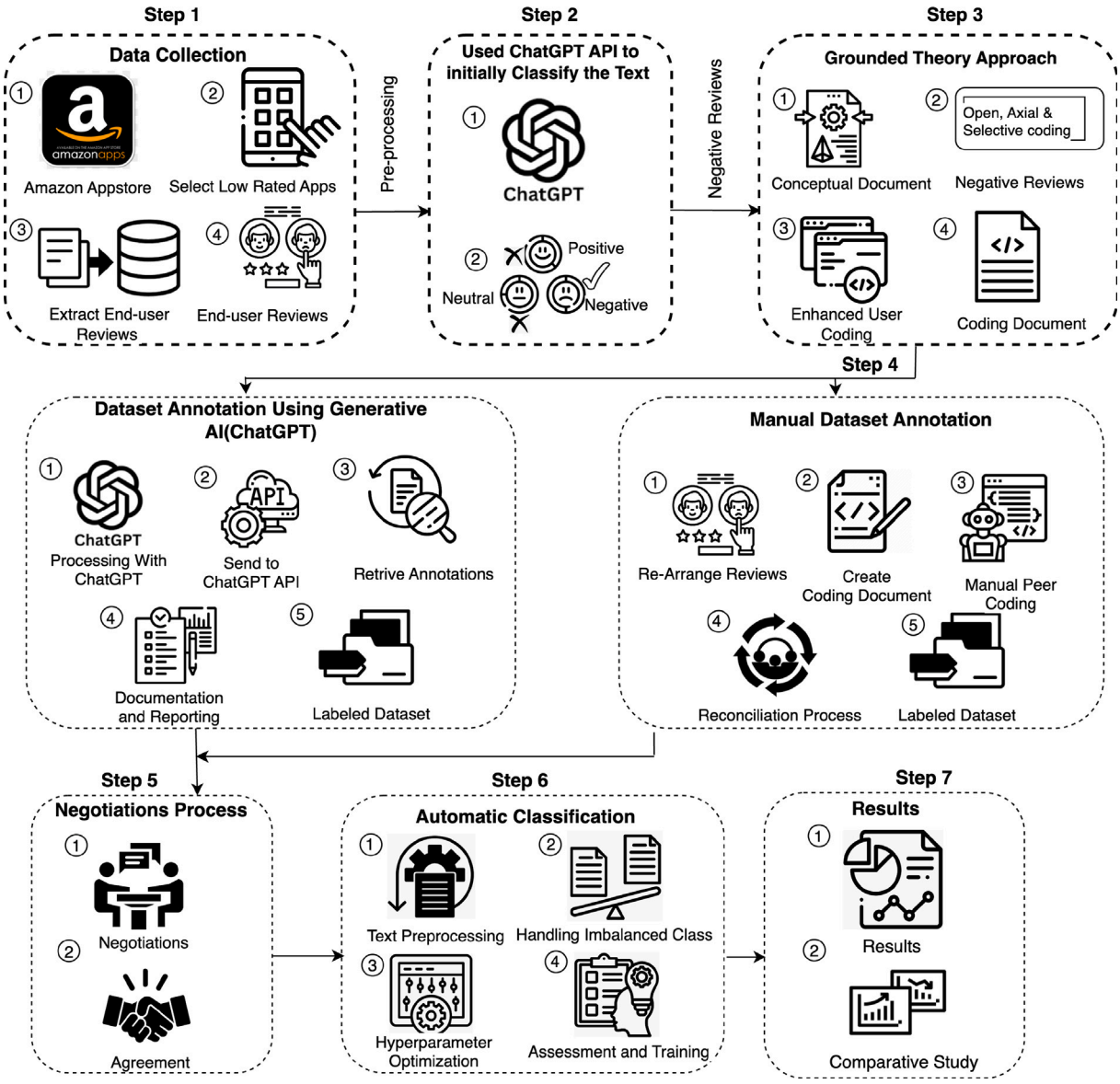



**Fig. 1.** An overview of our research methodology.

better. Unlike requirements engineering (RE) literature, where most of the research studies focus on popular software applications such as AngryBird (Guzman & Maalej, 2014), google maps (Khan, Xie, et al., 2019), firefox and Chrome (Mezouar et al., 2018), etc., ignoring the low-rating apps and a large pool of potential user and their serious grudges with the applications. Therefore, this study collects reviews against software apps that are not popular. The applications category, names, and total number of reviews in each category and app are shown in Table 2. In the dataset, we collected the reviewer's name, ratings, title, and main feedback text with each review from the ASA store. The data set plays a pivotal role in developing and validating the proposed approach. For instance, to create a novel coding guideline for the proposed approach, a random sample of end-user feedback is collected from the dataset to manually analyze it and identify the frequently mentioned emotion types in the user feedback. Similarly, the dataset is used in the experimental setup to determine the performance of various DL classifiers. Additionally, the dataset can be reused by software researchers and vendors for further analysis, such as identifying commonly occurring issues, etc.

## 5. Identifying end-user feedback sentiments using ChatGPT

The research dataset comprised 71,853 end-user reviews against the low-ranked software applications. We randomly select a sample of 140 end-user feedback from the dataset to analyze them manually and identify user possible emotions associated with app reviews. During the process, it was determined that end-users also posted positive feedback against the low-ranked apps in the ASA store, together with negative and neutral reviews. For the proposed approach, we are interested in analyzing end-user negative feedback for the low-ranked apps and exploring the possible end-user emotions associated with the negative reviews. Therefore, to differentiate the negative end-user feedback from the positive and natural one in the dataset. We harness the zero-shot learning capability of ChatGPT to automate the process and identify the sentiments of each end-user feedback, i.e., positive, neutral, and negative (Belal, She, & Wong, 2023). For this purpose, each end-user review in the dataset is passed to the ChatGPT API, which analyzes it and identifies the review's corresponding sentiment (positive, negative, and neutral). An example of interaction with ChatGPT is shown in Algorithm 1. The findings reveal that 11,800 end-user feedbacks are classified as negative sentiment and are selected for further processing

**Table 2**
Details of end-user reviews dataset.

| No. | Applications category | Applications name & End-user reviews | Total reviews |
|---|---|---|---|
| 1 | Business Apps | 1: Sketch Guru (2400) 2: Office Suite Free (2200) 3: PDF Max Pro (900) 4: Hammer Print (3126) | 8626 |
| 2 | Communication Apps | 1: TextMe - Free Text and Calls (2200) 2: Skype (2000) 3: AddMeSnaps (800) 4: Free Text, Text anyone (1622) 5: JusTalk - Free Video Calls and Fun Video Chat (2000) | 8622 |
| 3 | Lifestyle Apps | 1: Screen mirroring (1800) 2: DOGTV (2500) 3: Fanmio boxing: 3D Home designs layouts (2500) 4: Home design 3D-free (3499) | 10 299 |
| 4 | Food and Drinks Apps | 1: Italian Recipes (1425) 2: ChefTap: Recipe Clipper, Planner (2000) 3: Food Network Kitchen (2000) | 5425 |
| 5 | Game Apps | 1: Drive Car Spider Simulator (2200) 2: Chapters Interactive Stories (2500) 3: Crazy Animal Selfie Lenses (1000) 4: Cradle of Empires (1698) 5: Mobile Strike (1525) | 8923 |
| 6 | Novelty Apps | 1: Xray Scanner (400) 2: Cast Manager (500) 3: Age Scanner Classic (450) 4: Clock in Motion (469) 5: Ghost Radar: CLASSIC (400) | 2219 |
| 7 | Photos and Videos Apps | 1: AirScreen (2200) 2: Snappy Photo Filter And Stickers (1625) 3: ScreenCast (900) 4: RecMe Free Screen Recorder (850) 5: AirBeamTV screen mirroring receiver (1292) | 6867 |
| 8 | Productivity Apps | 1: Floor Plan Creator (500) 2: tinyCam Monitor FREE (400) 3: VPN for Fire TV (400) 4: Screen Stream Mirroring (500) 5: PrintBot (469) | 2269 |
| 9 | Sports and Exercise Apps | 1: fuboTV: Watch Live Sports, TV Shows, Movies And News (800) 2: NBC Sports (3000) 3: CBS Sports Stream And Watch Live (3800) 4: FOX Sports: Stream live NASCAR, Boxing (2016) | 9616 |
| 10 | Utilities Apps | 1: Floor Plan Creator (2000) 2: tinyCam Monitor FREE (1750) 3: Tv Screen Mirroring (2298) 4: Optimizer And Trash Cleaner (1989) 5: PrintBot (950) | 8987 |
| Total Applications | | 45 | 71 853 |

to identify the associated emotion. The positive and neutral feedback is discarded, as the scope of the paper is limited to working with negative end-user feedback and exploring and understanding end-user emotions when submitting negative reviews for the low-rating software applications on the ASA store. The paper explores whether identifying emotions with negative reviews can improve low-rating software apps' quality and user satisfaction. To validate the results obtained from the ChatGPT, we process the dataset with the VADER library (Hutto & Gilbert, 2014) frequently used for sentiment identification and classification in literature with moderate results (Marwat et al., 2022). The results obtained from ChatGPT and VADER were compared manually by the first 3 authors of the paper. It was identified that the sentiments identified by the ChatGPT were quite similar to the VADER library. For this experiment, we did not involve human coders to manually annotate the dataset and identify the associated sentiments by employing various deep learning classifiers for the following reasons: (i) the dataset was large, i.e., 71,853 end-user comments requiring a lot of manual efforts and resources, and (ii) VADER library performs well in identifying the associated sentiments. Also, we are interested in exploring the zero-shot learning capabilities of ChatGPT to identify end-user sentiment. Moreover, the human coders processed each end-user review when annotating end-user reviews using the identified emotion type. Therefore, if any review was identified as misclassified, it was removed from the dataset.

The implementation of ChatGPT for sentiment analysis in the proposed approach utilizes zero-shot learning, where the model is prompted to classify sentiment without prior specific training on a similar task within the context of the proposed dataset. We did not train ChatGPT to get the sentiments of each end-user feedback. Instead, ChatpGPT is a generative pre-trained transformer that has been rigorously trained on a diverse corpus of text data across various subjects, making it the best choice for sentiment analysis (Belal et al., 2023). Also, we did not provide any prior training (rounds of discussion on sentiment definitions) to ChatGPT to identify the sentiments of end-user feedback. We used ChatGPT API [2] as a prompt by passing each end-user feedback to identify its associated sentiment. Communication with ChatGPT was conducted using the API interface, where each end-user's feedback was sent as a text prompt, and the ChatGPT returned its associated sentiment. Unlike if this process were performed manually using an ordinary ChatGPT text prompt, it would involve copying and pasting each feedback into the ChatGPT interface. This task might be impractical for large datasets due to time constraints and the potential for human error. Moreover, we understand that ChatGPT can sometimes generate hallucinated responses if we utilize its zero-shot learning capability. To mitigate the risk of 'hallucinated' responses with ChatGPT, we process each end-user review with a VADER library to identify its associated sentiment. In the literature, the VADER library has proven to be a reliable source for identifying sentiments (Obaidi, Nagel, Specht, & Klünder, 2022). Additionally, the first two authors of the paper manually compared the annotation results generated by ChatGPT and VADER, and similar results were obtained with both, i.e., ChatGPT and VADER. Moreover, if any discrepancies were found in the sentiment results, the human annotator would manually correct them. However, during the inspection, it was reported for a few end-user feedback. This dual-validation method helps maintain the integrity of the proposed sentiment analysis process. Still, further validation with human annotators is required by employing various DL classifiers and comparing their performances. The process harnesses the power of ChatGPTs zero-shot learning in identifying sentiment associated with end-user reviews, making it an alternative automated approach to identifying sentiments.

## 6. A novel grounded theory for identifying end-user emotions

After identifying negative end-user feedback for the low-rating software applications in the ASA store, it is important to create a grounded theory document for identifying various emotion types associated with end-user reviews, as shown in Fig. 1. This document becomes a primary source for developing a labeled dataset, used by human annotators to curate a ground truth for the DL classifiers in automatically identifying emotion types associated with end-user reviews. Therefore, we employed the standard Grounded Theory method to develop a comprehensive coding guideline (Strauss & Corbin, 1998). The Grounded Theory approach provides structured guidelines for thoroughly analyzing the end-user feedback obtained from the ASA store and developing

[2] https://platform.openai.com/docs/api-reference/introduction

a novel theoretical framework that explains the overall practices required to annotate the required data for the DL classifiers. This framework is essential for structuring the data analysis and ensuring that other researchers in the field can replicate and validate the proposed methodology.

The coding guideline [3] document is developed through an iterative process that includes a diverse range of coding principles necessary for implementing the proposed approach. This guideline is designed to ensure a systematic and bias-minimized approach by providing a clear set of various emotion type definitions and associated examples that enhance the reliability of the data annotation for the human coders. A uniform and unique coding guideline is essential to develop an annotated dataset used as input by the proposed approach. Moreover, when incorporating multiple annotators in this phase, it is necessary to provide them with a coding guideline to help reduce bias and disagreement between them.

The coding guideline provides a comprehensive overview of the concepts identified through a rigorous analysis supported by concrete examples. For this purpose, the first two authors of the paper critically analyzed the randomly selected 200 end-user comments across the 10 categories of software applications on the ASA store. A uniform sample, i.e., 20 comments from each Amazon category, was collected for the analysis. The author critically analyzes the end-user feedback and identifies the possible emotion types associated with the reviews. Ekman and Davidson (Ekman & Davidson, 1994) have inspired the guidelines for the authors about selecting various emotion types related to the end-user negative feedback, which has been widely used for emotion detection in literature. The emotion types identified during the manual analysis have emerged into concepts, resulting in a novel grounded theory for emotion detection associated with negative end-user reviews for low-rating software applications. The emotion types included in the emotion coding guideline document were identified by their consistent presence in the end-user reviews for the low-rating software applications and their relevance to the aim of the proposed approach. The most frequently associated emotion types identified in the end-user reviews include anger, confusion, disappointment, distrust, disgust, frustration, fear, and sadness. Moreover, if an emotion concept appears less commonly in the end-user reviews for the low-rating application, it emerges into a related corresponding emotion type. For example, the emotion type "Helplessness" can be merged with "frustration", as it often accompanies the feeling of being unable to rectify issues or achieve satisfactory results. Although Ekman and Davidson have introduced four negative emotions: sadness, disgust, fear, and Anger (Ekman & Davidson, 1994). The remaining emotion types have been identified through discussion and their frequent presence in the end-user reviews for the low-rating apps. Also, if a coder identifies an emotional concept initially, later, if it is identified as irrelevant or repetitive based on the aim of the proposed approach. The identified concept is either discarded or emerges in the existing emotional concepts. As shown in Fig. 1, the grounded theory step was performed iteratively. After the individual process was completed, the results from both authors were combined to shortlist the unique emotion types. The conflicts among the coders were resolved through discussion and negotiations, as shown in Fig. 1.

To further validate these emotion types' coverage, additional annotations were manually conducted on larger subsets of the data by the same two authors of the paper. These rounds validated that the eight identified emotions remained consistent across different samples and represented the user emotions expressed for the low-rating software apps across the entire dataset. This comprehensive approach assures that the grounded theory developed is based on a solid methodological foundation and an extensively validated larger dataset, reinforcing its applicability and robustness in capturing the full range of user emotions. Finally, a stable and unique grounded theory containing definitions for each end-user emotion type with examples is developed. The research dataset annotation is a challenging and time-consuming process, for which researchers have provided alternative automated approaches to automate the process (Maalej & Robillard, 2013; Neuendorf, 2017). In line with the previous approaches, we put an effort into automating the content annotation process by involving ChatGPT in the loop to annotate the dataset and compare it with the human-annotated dataset, aiming to validate the human annotation approach. Later, ChatGPT is used as a negotiator to resolve the conflicts between the human annotators and ChatGPT, resulting in a novel conflict-free emotion dataset (explained in Section 7). To conclude, eight emotion types were identified manually when analyzing end-user feedback: anger, confusion, disappointment, distrust, disgust, frustration, fear, and sadness. Moreover, it was not easy to differentiate between the different emotion types described by the end-users in their corresponding feedback because a single user comment depicts multiple emotions. However, we chose the closest and most major emotion described in the feedback. Below, each end-user emotion concept is elaborated in detail:

**Anger:** The 'anger' label is assigned to the end-user feedback in the ASA store that describes anger preventing them from achieving the desired goals with either a single feature or an overall software application. Also, a specific software feature or application behaves unexpectedly or unfairly. Anger is one of the eight emotion types that arises when one fails to reach the desired functionalities or is oppressed. At its most extreme, anger can be one of the most dangerous emotions due to its possible link to violence, resulting in providing a low rating review to the software application on the ASA store. Such information is of pivotal importance for software vendors and developers, as it might result in the uninstallation of the software applications. Therefore, potentially identifying the associated emotion of anger with the end-user feedback is crucial for understanding user dissatisfaction and as a preventive measure. Timely addressing the end-users grudges with the software app or a particular feature by adding the proposed approach to the software evolution process can lead to app improvements and help prevent negative outcomes, such as low ratings and app uninstallation, thus sustaining user engagement and improving app ratings. Some examples of the anger concept in the Amazon store are. *I am furious with this app. It promised a seamless experience, but I have encountered constant crashes and a never-ending stream of error messages. Having my expectations shattered by such a poorly performing application is infuriating. It is a complete letdown, and I would not recommend it to anyone.* In this review, the user expresses anger toward the app and is deeply disturbed by its performance. The user expected a smooth experience but encountered issues like crashes and error messages, leaving them feeling infuriated. This emotional response reflects their strong negative sentiment about the app's lack of reliability and quality, making it a clear example of the "anger" emotion. Another example is *This app is driving me up the wall! It is filled with invasive ads that pop up at the most inappropriate times, making it impossible to use without constant interruptions. Seeing an application that promised utility become a platform for strict advertising is enraging. I am fed up with it.* In this review, the user expresses anger towards the app. They are highly annoyed and discouraged by the app's intrusive ads, which disrupt their usage and make it difficult to achieve their intended tasks. The user desired a functional and ad-free experience, but the constant interruptions have left them feeling angry, making this review another example of the "anger" emotion. Identifying and addressing these sources of anger can significantly improve app functionality and user satisfaction, thus directly influencing the app's success and retention.

**Confusion**: The code" confusion" is assigned to end-user feedback in the ASA store when users express hesitation or difficulty understanding a specific software feature or overall application. This uncertainty often derives from unclear documentation, a non-intuitive user interface, or complex functionalities. For instance, a user review from the ASA store, *I found this app confusing and challenging to navigate.* In

[3] https://github.com/nekdil566/Understanding-End-User-Emotions

this review, the user expresses their struggle with the app's usability. Another example is *Not worth it. Very confusing app, the picture froze, and I could not enjoy the show on the big screen TV*. The end-user complained about confusing app interfaces and the software's complex flow issues in this negative feedback. Recognizing and addressing such instances of confusion or ambiguities can aid developers in improving the user interface and the overall user experience, resulting in improved app ratings. Requirement engineers and Developers should prioritize addressing these usability and technical issues to improve the overall user experience.

**Disappointment**: The ''disappointment'' code is assigned to end-user feedback when users convey a sense of disillusionment or dissatisfaction by using a particular software feature or the overall software application. Disappointment often arises when end-users do not get the desired needs from the software application. For instance, a user might express disappointment, *I am not satisfied with the results. It does not seem to work properly when cleaning my mobile, as it shows 0 bytes removed, and the download continues. It needs improvements.* This review reflects the user's disappointment due to ineffective software functionalities. Also, *So many ads. Please do not download it unless you want to buy it. I downloaded this because it had decent reviews. I am disappointed. Every 3 min or so, while I was screen sharing from my phone, it would start playing some random annoying ad for HBO or something. I was trying to give this a shot, but after seeing how much they are trying to charge for the program.. no. It is not worth it. The video was also choppy and behind sometimes, even with the changes to performance in the advanced settings.* In this review, the user expresses disappointment with excessive ads in the app and dissatisfaction with the performance and over-high cost. Such end-user feedback highlights the need to understand better the disappointment with software functionalities or features by the software vendors or developers to help improve app ratings by achieving higher user satisfaction.

**Distrust**: The code ''distrust'' is assigned to end-user feedback in the Amazon store when users express anxiety regarding the reliability or credibility of a software feature or the entire software application. Distrust can emerge when software consistently underperforms, or there is perceived jeopardy to the user's safety, particularly regarding data privacy or potential misuse. For example, A user expressed distrust by commenting, *The new permissions ruin what is otherwise a great tool. Also, I do not like apps requiring the internet to run. I do not need the internet to be operational on my PC for 90% of my apps; why should I have internet active on my phone?* Such sentiments reflect concerns about unnecessary permissions and reliance on internet connectivity, contributing to an overall sense of distrust in the app. Another example is *Rubbish. Does not work, and strange activity on my router since I have installed it.* In the review, the end-user complained about the usual behavior of the software application, resulting in a distrust over the app's functionalities. Such feedback analysis helps software vendors and developers improve the security, privacy, and behavioral aspects of the software applications to help improve the software ratings on the ASA and other software stores. Countering such useful feedback in the software applications can enhance end-user confidence in the apps.

**Disgust**: The code'' disgust'' is assigned to end-user feedback, where crowd-users express their emotions by strongly disliking a specific software feature or the application due to an issue or a bug, making the software produce the desirable outputs. Disgust is the type of emotion defined by a strong dislike for something unpleasant about the software application they are using. Software vendors and developers need to identify the disgusted emotions of crowd-users in the ASA store, as they mainly describe the software application's negative behavior, frequently occurring issues or bugs. For example, an end-user review, *Dumbest app ever!. Waste of bandwidth to download. Found the ads distracting. Picture quality is HORRIBLE.* In this review, the user strongly criticizes the app, deeming it the ''dumbest app ever'' and a waste of bandwidth. They express dissatisfaction with distracting ads and poor picture quality, urging others to avoid this application and seek a better alternative. Another example is *Not useful. I did not like it at all. It is hard to use it.* The user's concise feedback expresses their dissatisfaction with the app, finding it useless and difficult to use. It is keen to identify such pivotal emotions with the issues reported by the end-user to gain their trust by timely removing their grudges. Conversely, it instigates other users from installing or purchasing this software application, losing user trust and satisfaction.

**Fear**: The code'' fear'' is assigned to end-user feedback in the Amazon store where crowd-users sense or feel a possible threat when using the software app. Fear is induced by the threat of harm, which can be physiological, emotional, or mental. In requirements and software engineering, fear is often considered a negative emotion, making users reluctant to trust the functionalities provided by the software app. Identifying it earlier in the software development phase can give the crowd-users enough confidence in its functionalities and overall performance. Some examples of end-user reviews representing fear end-user emotions in the ASA store are *Crazy!!. I was trying this radar out, and then it said in a deep voice BATTERY then one mili second after that my tablet turned off. I got scared obviously, so yea... CREAPY*. In this review, the user shares a creepy and unsettling experience with the app, mentioning that it unexpectedly vocalized ''BATTERY'' and caused their tablet to turn off immediately afterwards. Such an incident is concerning and may indicate a technical issue or a potential glitch within the app that needs investigation and correction to ensure user safety and satisfaction. Another example is *SO FRICKING SCARY!!!!!. It said bigger and round. Before my mom said go ride around the yard on my bike, it said round.* Identifying fear emotions with end-user reviews highlights the uncertain situation happening with the software applications. Identifying and resolving them in the coming version would increase user trust. Such kinds of end-user emotions are important to identify in the software applications that help software vendors cover various untested paths that might lead to uninstallation if not resolved.

**Frustration**: The code'' frustration'' is assigned to end-user feedback when users express frustration in their reviews because software features do not operate as anticipated, there are recurring issues, and crucial functionality is absent or not working. For instance, an end-user expressed frustration with the software app by reporting, *That app never opened; it was frozen, and nothing could be done. This was one of the worst apps I have ever downloaded. I thank GOD I did not have to pay for it. If I could give negative five stars, I would have.* In this review, the user had a terrible experience with the app, as it failed to open and remained frozen, leading to a strong negative impact on the end-user satisfaction, leading to uninstallation and low-rating. Another example is, *This game is so stupid it does not even tell you what to do. I do not get it. All it is a bug circle that spins, and a dot appears. That is the ghost, and you have to tap it. That gets rid of the ghosts*. The user criticizes the game, describing it as stupid and lacking clear instructions on gameplay. Frustration is another user emotion associated with their feedback that must be identified and reported to the development teams to remove their serious concerns with missing functionalities or complex software flow. Understanding such end-user emotions should help resolve hidden non-functional requirements that are difficult to identify upfront. Therefore, identifying and addressing these frustrations and emotions associated with the issues reported by the end-users on time can significantly improve user satisfaction and the overall ratings of the software application.

**Sadness**: The ''sadness'' code is assigned to an end-user comment in the ASA store where crowd-users express grief or pain over a software feature, bug, or issue while using it. Generally, a sad emotion is expressed when a loved one or an essential item is lost. Similarly, crowd-users express sadness when unsatisfied with a software application or a particular feature when reporting comments in the ASA store. For example, *Costly. I use this game for entertainment and relaxing; however, as you upgrade, it costs more. Then with upgrades, you may lose all your progress and start all over. Depressing and not fun*. In this review, the user expresses sadness about the game, mainly due to its increasing costs

as you advance and the possibility of losing progress with upgrades. Another end-user expresses sad emotion as *Fun game but almost requires purchases. I enjoy this game, but it's sad that it almost requires purchases of crystals. The most significant negative is that I downloaded it at Amazon Underground, but the free feature disappeared when it upgraded.* Identifying and Intertwining sadness emotions with the issue reported in the reviews could help software developers understand the disappointment with the software applications. The end-user emotions and the review can be beneficial to highlight the importance of transparency in software app features and potential sadness when features change unexpectedly across different versions or updates.

## 7. Analyzing the end-user feedback to develop a truth set

After developing the coding guideline and identifying various end-user emotion types to better understand user issues with the software application on the ASA store, we would now analyze each review in the dataset using the content analysis approach (Cohen, 1968) as highlighted by Neuendorf (Neuendorf, 2017) and Maalej & Robillard (Maalej & Robillard, 2013). This step assigns a reasonable and related emotion type to each user feedback. The emotion types identified are anger, confusion, disgust, distrust, disappointment, fear, frustration, and sadness. The purpose is to create a truth set that can be used to train and test different deep-learning classifiers. A coding guideline containing explicit instructions, descriptions, and examples of capturing end-user emotional types in the ASA store was developed in the previous step to ensure consistency and reduce coder misunderstandings and conflicts. To detect and identify crowd-user emotion types by the manual annotation process, the first two authors of the research paper independently assessed and analyzed each end-user feedback in the dataset. This manually annotated dataset serves as the primary reference set for validating the accuracy of emotion classification using DL classifiers and is compared with the dataset curated with ChatGPT to validate the human-annotated dataset. Both authors independently work on this process, and disagreements are resolved through discussion and consensus. Moreover, to automate the coding process and validate the human annotation process, we employed ChatGPT to annotate the end-user feedback and develop a separate truth set by analyzing each crowd-user's feedback and assigning the most relevant emotion type. Later, ChatGPT is employed to negotiate the conflicts between the human annotators and chatGPT for a conflict-free novel emotion data set. The steps are elaborated on in detail below.

### 7.1. Manual annotation of end-user emotion types

The supervised ML and DL classifiers operate on annotated data. Therefore, the proposed approach involved the first two authors of the paper annotating the end-user feedback as shown in Table 3 manually, which is classified as negative in the previous step. Two coders received a coding guideline and a coding form containing 11800 end-user feedback to be processed and annotated. Each coder independently analyzed the "Review Title" and "Full Review", the main content of each end-user feedback, to determine the user emotion associated with the feedback. The coders must choose one of the emotion types (anger, confusion, disgust, distrust, disappointment, fear, frustration, or sadness) for the end-user feedback that is closely aligned with it. The manual annotation process was a tedious and challenging task. When analyzing the end-user feedback, the coders faced challenges identifying the closely related emotion type because a user review usually represents more than one emotion. However, based on their understanding, the coder selects the most relevant and dominant emotion type for each end-user feedback in the dataset.

To maintain uniformity in the annotation process, a thorough and detailed coding guideline containing each emotion type definition with examples has been provided to the human annotator that serves as a criterion for annotating end-user feedback. The main criteria used by the annotators to annotate the end-user reviews are the emotion stated with the highest intensity or the one that directly relates to the central topic covered in the feedback. This criterion is critical, especially when there are many emotions, to identify the most influential emotion that is likely to affect the user's entire experience. When the user displays a range of emotions, including 'anger' and 'frustration', the annotation gives priority to the emotion that directly relates to the primary complaint or issue at hand. Moreover, the paper's scope is limited to the review-level annotation that helps annotators focus on the border emotion type showing the highest intensity. Moreover, a thorough negotiation process using intensive discussion is in place to negotiate on the possible emotion type, limiting the possible conflicts between the human annotators. For example, an end-user reported, "Takes some figuring out, but it's free. I hate the app you have to get credits for stupid offers that don't work; it is a waste of time", demonstrating overlapping emotions of anger and Frustration. However, the annotator selected the "Anger" emotion type over "Frustration" as the primary emotion because of the statement " hate the app". Still, it shows a weak emotion of "Frustration" as a point of the sentence "waste of time". Therefore, for end-user reviews that possibly possess multiple emotion types, coders are advised to analyze the contextual understanding of the end-user reviews, i.e., possibly infer emotion type by considering the language and overall user tone. The general criteria used by the annotator is to identify which emotion type is dominant in the end-comment considering the context or intensity of the language used. Similarly, the feedback submitted by an end-user as "I am crying. Couldn't use or even uninstall, so obviously not happy. Also, difficult to use layout is so difficult" demonstrates multiple associated emotions, i.e., "sadness" and "disappointment". The annotator selected the primary emotion type as "sadness" because of the sentence "I am crying", considering the criteria defined. However, it also demonstrates the "disappointment" emotion type due to the sequence of words in the review as "so obviously not happy". Moreover, we thoroughly analyze the end-user feedback to identify the occurrence of multi-labelled emotions in the dataset. There was a high percentage of end-user reviews, with 4,682 end-user reviews (39.68%) reported by the coders to have multiple emotions. This makes it a strong case study for conducting experiments with DL classifiers by providing a multi-labelled dataset. However, the scope of the paper is limited to review-based emotion detection, which focuses on the most stressed emotion type in end-user reviews. Also, we aim to explore the suitability and usability of ChatGPT as an alternative source for manual annotation and negotiation for improved emotion classification results.

The end-user feedback was organized against each software application and category in the coding form. For example, for the business app category, end-user comments were categorized against each software application, such as Sketch Guru, Office Suite Free, PDF Max Pro, and Hammer Print. Also, the link to each software application was added in the coding form to facilitate the coders if there needed to be more clarity in understanding the user comments. The average time it took the coders to complete the annotation process was 26 working hours. The coders could stop the review annotation at any time and can resume. It took comparatively longer because of the challenge to identify exactly one emotion type for the reviews. Both the coders are experienced software researchers. Coder One is a Ph.D. student with previous experience in review annotation. Coder Two is a senior researcher in software engineering who is involved in multiple research annotation projects. After completing the individual annotation task, the coders grouped the individual coding results to analyze and recover the interceding disagreement. The conflicts between the coders were resolved with discussion as the annotation process was completed iteratively. Also, a senior professor with vast experience in software engineering research and annotation was involved in negotiating the conflicts, if there were any. The inter-coding agreement between the two coders was reported at 88%. At the same time, Cohen's kappa was

**Table 3**
Shows our manually labeled data set and ChatGPT annotation.

| Star | End-user review | ChatGPT identified emotion | Sentiment score | ChatGPT annotation | Manual annotation |
|---|---|---|---|---|---|
| 3 | I'm having issues with this app. Help! It says it has been saved, but then I go back in, and my highlights and underlines are gone. It asked if I wanted to "back up" I said yes, and it doesn't appear to have done anything. A few seconds later, the app "encountered a problem" and forced close on me. | The dominant emotion expressed in this review is frustration. The user expresses frustration with the lack of an introduction or basic instructions for the app and the inability to try apps before buying them. They also express frustration for less experienced users who may feel stupid when they can't figure something out due to the software. | −0.20 | Frustration | Anger |
| 4 | Update fails. Manual update needed. Does not work, states it does. I'm not very pleased with this product at all. There is definitely a lag time between the transmission to the TV I have the high-speed Internet and so it is not connectivity or an Internet issue. | The dominant emotion expressed in this review is disappointment. The user states that the product does not work as it is supposed to and expresses displeasure with it. This indicates a sense of letdown and dissatisfaction with the software. | −0.09 | Disappointment | Sadness |
| 1 | Not a fan. Had secret hidden fees once attempting to use. | Based on the content and tone of the comment, it appears to be negative. The dominant emotion expressed in this review is distrust. | −0.28 | Distrust | Distrust |
| 4 | What happened? I have tried to open this app but now it has gone black. I see that quite a few others are having the same problem. I don't want to lose all my work in the game. | The dominant emotion expressed in this review is fear. The user is worried about losing all the work they have in the game because the app has gone black and they not able to open it. | −0.09 | Fear | Anger |
| 1 | Spam emails following purchase and channel sucks. I dislike getting emails asking me to review. Also, the channel sucks. | The dominant emotion expressed in this review is disgust. The user dislikes receiving spam emails and finds the channel to be unsatisfactory. | −0.20 | Disgust | Disgust |
| 1 | Confusing. Didn't like this game! It seemed difficult to understand how to play. | The dominant emotion expressed in this review is confusion. | −0.43 | Confusion | Disgust |
| 1 | I HATE THIS. Over the summer I was gone and Skype is the only thing I could use it crashed I cried I missed my family don't get it updated. | The dominant emotion expressed in this review is sadness. The user expresses sadness over the fact that Skype crashed while they were away during the summer, causing them to miss their family, also state that they hate Skype and advise against updating it. | −0.40 | Sadness | Anger |
| 2 | Hard to use and lags quite a bit. All around average product. | The dominant emotion expressed in this review is frustration. The user mentions that the product is hard to use and lags, indicating dissatisfaction and hindrance in achieving their goals. Additionally, describing the product as "average" suggests a lack of satisfaction and potentially unresolved frustrations. | −0.22 | Frustration | Frustration |
| 1 | My mistake. I should have paid more attention to the other reviewers who said this was a ripoff. It is useless, but there are no zero star reviews possible. | The dominant emotion expressed in this review is disappointment. The user expresses their regret for not paying attention to negative reviews and describes the software as useless. They also mention the absence of zero-star reviews, indicating their dissatisfaction with the software. | −0.03 | Disappointment | Disappointment |
| 1 | Confused and disappointed. Did not activate. I wish in the description of these 'apps' it said whether you'd have to pay a subscription fee or be subscribed to a cable service. | The dominant emotion expressed in the review is disappointment. | −0.57 | Disappointment | Confusion |

identified as 66%, a substantial agreement between the annotators on Cohen's kappa scale. Finally, after the reconciliation step, a conflict-free, labeled data set was curated to be used as input to the different deep-learning algorithms to automatically classify end-user reviews in the ASA store. In the following step, we employed ChatGPT to annotate the end-user feedback and identify its suitability and applicability in software engineering research. The same dataset was chosen as the ChatGPT already annotated.

**Challenges:** End-user emotion detection by analyzing user feedback in the ASA store was a challenging task because a single user comment depicts multiple emotion types. Moreover, the study aims to explore the performance of DL classifiers in classifying end-user reviews into various emotion types using manual and ChatGPT annotated datasets. Therefore, we consider only the closest and most prominent emotion types associated with each review. In the future, we aim to extend the proposed approach by annotating the end-user comments at a sentence level to cover better the multiple emotions associated with each user comment. Also, to overcome the challenge, multi-emotion labels can be assigned to the end-user feedback to understand the associated emotions better. Additionally, such an experimental setup will be a good contribution to the knowledge of requirements and software evolution as not many researchers have explored it before (Ullah et al., 2023).

### 7.2. Automated annotation using generative AI (ChatGPT)

The coder found the manual annotation process challenging and time-consuming, as reported in the software engineering literature (Khan, Xie, et al., 2019; Maalej et al., 2016). It is stressed in the literature that an automated approach must be utilized to overcome the challenges faced by software coders (Ullah et al., 2023). For this purpose, we utilized ChatGPT API to take advantage of Large Language Models (LLM) by analyzing each end-user feedback (11,800 reviews) to identify the associated end-user emotion type, i.e., anger, confusion, disgust, distrust, disappointment, fear, frustration, and sadness. For this specific task, we utilized ChatGPT, more precisely, the GPT-3.5 Turbo model,[4] which takes a single end-user feedback as input and produces a potential associated end-user comment as output. Initially, the API key is set up and includes all the necessary libraries. While processing the end-user feedback by ChatGPT, we encountered a server issue that halted the processing for the time being, causing significant delay and financial loss. To cater to this, we used the "try_request" function, which automatically handles response issues from the ChatGPT server and resubmits the request after some moments. Also, to validate the previous ChatGPT sentiment identification task, the algorithm reevaluates the end-user feedback and identifies if the review is negative.

The prompts used for training ChatGPT on various emotion types were crafted based on the definitions identified in the grounded theory,[5] using the ChatGPT API. Before annotating the end-user feedback with various emotion types, we trained ChatGPT with the grounded theory previously developed for identifying emotions associated with the end-user feedback, aiming to make the annotation process consistent with human annotators. Each prompt was constructed to guide ChatGPT in identifying one of the specific emotions from end-user feedback. These prompts encapsulated the essence and nuances of each emotion: anger, confusion, disgust, distrust, disappointment, fear, frustration, and sadness, using detailed descriptions and examples from our previous analysis of the user feedback, as shown in Section 6 of the paper. The prompts, exemplified in Box 7.2 illustrate how we utilized ChatGPT to annotate end-user feedback into various emotion types using ChatGPT API.

ChatGPT processes the end-user comment upon the negative review and assigns a suitable emotion type to the user feedback. The ChatGPT has already been trained on the various emotion types with examples identified in the grounded theory. The detailed process is explained in Algorithm 1. After processing all the end-user feedback in the dataset, an annotated dataset is curated to pass as input to the DL classifiers. The annotation process for 11,800 end-user comments is completed in approximately 7 working hours. However, the process was expensive, more than 40 USD and required some technical skills in Python and programming. The aim is to compare the performance of various DL classifiers when identifying emotions associated with the end-user comments by processing the annotated dataset by human and chatGPT. We aimed to bring automation to the supervised learning task, which is considered tedious and time-consuming (manual annotation). We are also interested in checking the capabilities of the LLMs in processing complex tasks, such as identifying correct and prominent emotion types where human annotators were challenged, as most of the end-user comments represent multiple emotion types.

**Prompts for Emotion Identification Using ChatGPT**

**Sentiment Analysis Prompt:**

```
"The following comment was left on the Amazon
    store:
'{review_text}'
Based on the content and tone of the comment, is
    it negative or positive?"
```

**Emotion Analysis Prompt:**

```
"Please identify the dominant emotion expressed
    in the following review based on these
    definitions:

- Sadness: Defined as expressions of grief over
    software features or performance.
- Anger: A strong emotional response to feeling
    obstructed or wronged by the software, often
    resulting from unmet expectations, poor
    performance, or perceived unfairness.
- Disgust: Strong dislike towards specific
    features or overall software quality.
- Fear: Concerns about reliability or perceived
    threats in software performance.
- Distrust: Apprehensions regarding software
    reliability, especially around data privacy.
- Frustration: A feeling of annoyance due to
    repeated obstacles or the inability to
    achieve desired outcomes, often resulting
    from unclear functionality or persistent
    issues.
- Confusion: Lack of clarity regarding software
    functionality or user interface.
- Disappointment: Letdown due to software failing
    to meet user expectations.

Review: '{review_text}'"
```

[4] https://platform.openai.com/docs/models/gpt-3-5
[5] https://github.com/nekdil566/Understanding-End-User-Emotions

**Validation and Reconciliation of Annotations**

**Overview of Disagreements:** Approximately 42.06% of the user reviews contained disagreements between human annotations and ChatGPT's classifications.

**Resolution through Negotiation:** During the negotiation process with ChatGPT, a consensus was reached in 80% of the cases where human annotations agreed with ChatGPT's suggestions. In the remaining 20% of cases, ChatGPT adjusted its suggestions to align with the human coders' annotations.

**Example of Review Disagreement and Resolution:**

- User Review: "I was going to mirror my mac with my Fire TV...but this doesn't work with the audio immediately. You need to download a driver to do that. As I was doing that, the trial period ran out on the app. So...no. I'll just watch my streaming services on my TV."
- Initial Annotation: ChatGPT classified this review as "Frustration" due to the user's experience of obstacles and unmet expectations during the setup process.
- Manual Annotation: Initially annotated "Disgust" by human coders but, upon review, the human coders agreed with ChatGPT's classification of "Frustration," leading to a resolved consensus.

**Implication:** This collaborative approach highlights the effectiveness of using ChatGPT to refine and reconcile discrepancies in user feedback classification, ultimately enhancing the accuracy and reliability of the annotated dataset.

### 7.3. Validation and reconciliation of annotations

For the DL classifiers, we aim to provide a single truth set to identify the performance of various DL classifiers. Therefore, in this step, another round with ChatGPT is conducted to validate and reconcile the differences between manual and ChatGPT-based annotations of end-user feedback, focusing on predefined emotion categories such as anger, sadness, disappointment, confusion, fear, disgust, distrust, and frustration. The prompts, exemplified in the following instance (Box 7.3 above, illustrate how ChatGPT can assist in resolving conflicts between the human annotators and ChatGPT. In the previous steps, the dataset comprised 11,800 user reviews annotated separately by human coders and ChatGPT. Initial comparisons revealed notable differences between the two annotation methods, prompting a negotiation and reconciliation step as shown in Fig. 1 to remove the disagreement and inconsistency across annotations. The Cohen's Kappa identified for the manual, and ChatGPT annotated data is 0.40, considered a Moderate agreement between the human annotators and ChatGPT. Therefore, we need to run a negotiation cycle to develop a conflict-free novel truth set by removing the conflicts. The reconciliation process is systematically conducted through the negotiation process, as depicted in Fig. 2. This process involved iterative discussions where ChatGPT clarified specific emotion categories whenever there was a discrepancy with the human coder's annotation. The human coder played a critical role by using their judgment and understanding of context to evaluate ChatGPT's responses. Depending on the effectiveness of the explanation provided by ChatGPT, the human coder would either revise their annotation or, conversely, ChatGPT would adjust its classification. This iterative adjustment continued until both parties reached an agreement. The total number of reviews analyzed involved 4,963 disagreements; notable instances of these disagreements included 1,814 instances in the category of 'Frustration', 1,414 in 'Disappointment', and 829 in 'Anger', highlighting the areas where the negotiation was most intensely focused. Table 4 systematically summarizes these differences and the negotiation outcomes. The final, mutually agreed-upon dataset is visually represented in Fig. 3. This conflict-free dataset significantly enhanced the quality of labeled data, improving the accuracy and performance of DL classifiers used to detect end-user emotions. This improvement underscores the proposed approach's efficacy and validates the dataset's reliability for subsequent analysis.

**Algorithm 1** Processing Reviews for Sentiment and Emotion using ChatGPT API

**Require:** *DataFrame* `df` with reviews
**Ensure:** *DataFrame* with sentiments and emotions for negative reviews

```
Initialize: Import necessary libraries
Set the OpenAI API key
Load data from "dataset.csv" into df
function TRY_REQUEST(func, args)
    try
        return func(args)
    catch error
        Handle error and retry logic
end function
function GET_SENTIMENT(review_text)
    Output: Sentiment of the review
    Set prompt for sentiment analysis
    sentiment ← TRY_REQUEST(OpenAI model call with prompt)
    return sentiment
end function
function GET_EMOTION(review_text)
    Output: Emotion of the review
    Set prompt for emotion analysis
    emotion ← TRY_REQUEST(OpenAI model call with prompt)
    return emotion
end function
Split df into chunks of size n
for each chunk in df do
    for each row in chunk do
        sentiment ← GET_SENTIMENT(row['Full_Review'])
        row['sentiment'] ← sentiment
    end for
    Filter rows with negative sentiment into chunk_negative
    for each row in chunk_negative do
        emotion ← GET_EMOTION(row['Full_Review'])
        row['emotion'] ← emotion
    end for
    Append chunk_negative to "Annotated-dataset.csv"
end for
```

### 7.4. Frequency of the crowd-user emotions

After applying the negotiation cycle, a conflict-free emotion dataset is developed to identify the performances of various DL classifiers in automatically classifying end-user feedback into various emotion types. In this section, we identified the frequencies of various end-user emotion types in the dataset to better understand the nature of the end-user feedback and associated emotions. Additionally, the aim is to highlight the frequently reported emotions alongside the issues or problems submitted by end-users in the ASA store. This insight allows software vendors and researchers to devise approaches to minimize user distrust and dissatisfaction in future versions of the software apps, considering all suggestions, issues, or problems reported. Understanding user emotions and problems with software applications can significantly improve the ratings of low-ranked applications. Fig. 3 depicts the end-user feedback distribution across various emotion types in the conflict-free novel emotion dataset, which will be used to evaluate the performances of various DL classifiers.

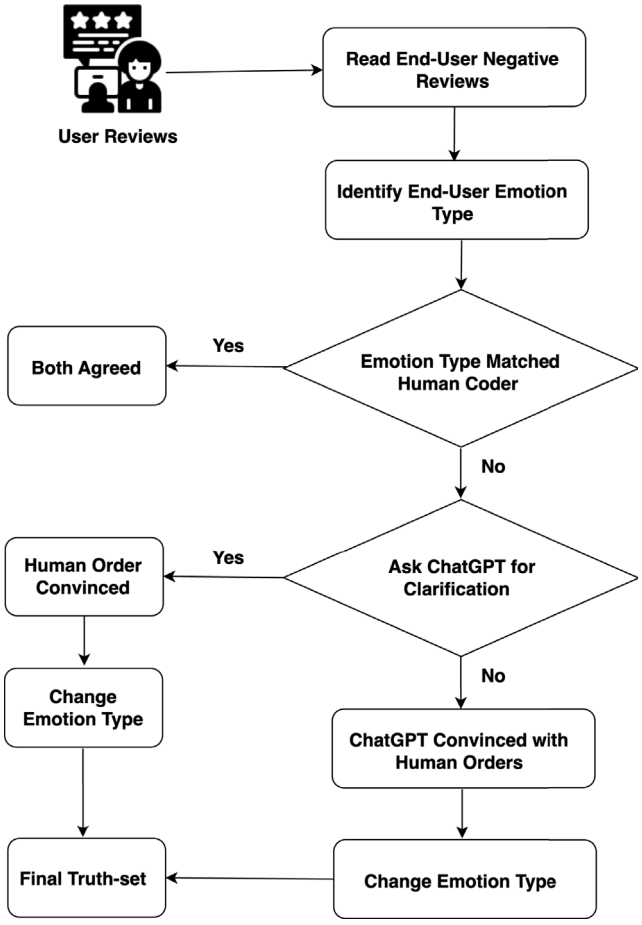



**Fig. 2.** Annotation and Negotiation Process With ChatGPT.

The detailed analysis revealed that "Frustration" is the most prevalent emotion, accounting for 38.2% (4508 comments) of the emotions identified. This significant percentage indicates widespread dissatisfaction among users, primarily due to unmet expectations or persistent functional problems within the software. The high incidence of frustration highlights critical areas in user experience that require urgent attention to improve application usability and functionality. Following closely, "Disappointment" emerged as the second most frequent emotion, representing 27.7% (3264 comments) of the dataset. This emotion points to a substantial disparity between user expectations and the software's actual performance or features, suggesting that users feel let down by what the software promises versus what it delivers. Addressing these gaps is essential for software developers aiming to enhance user retention and satisfaction.

Further, "Anger" was also notably significant, comprising 14.5% (1716 comments) of the emotional feedback. Anger typically reflects more intense dissatisfaction and may pinpoint critical issues that, if resolved, could drastically improve user perceptions and the overall market success of the software. Other emotions, such as "Disgust" and "Confusion", each made up approximately 6.4% (750 comments) and 5.1% (597 comments), respectively. These emotions often relate to more specific user interface or interaction flaws that, while not as prevalent as frustration or disappointment, could severely impact the user experience if not addressed. "Distrust" appeared in 2.8% (327 comments) of the feedback, highlighting potential concerns over security, privacy, or reliability that could deter users from fully engaging with the software. The least frequent emotions were "Fear" and "Sadness", observed in 4.6% (541 comments) and 0.8% (97 comments) of the comments, respectively. While these emotions are less common, they indicate more profound issues that, though rare, could have severe implications on user trust and satisfaction. Fig. 3 graphically depicts these findings, showing the distribution of emotions within the annotated feedback. This graphical representation is invaluable for developers as it highlights the most critical areas needing improvement and provides a clear picture of user sentiment that can guide future development strategies.

## 8. Automated classification of end user emotions

With the proposed approach, we aim to extend the functionalities and interface of the existing social media platforms by identifying the associated emotions with the end-user feedback. It will help software vendors and developers focus on critical end-user feedback by incorporating it into the software evolution process to retain users by achieving higher satisfaction. For this purpose, in the previous step, it is evident that the ASA store contains useful information about various emotion types, which, when processed, would fill the research gap. Also, manually handling such large information is a tedious and time-consuming process. For this purpose, we utilized existing deep-learning classifiers to capture their performances in identifying emotions of Anger, confusion, disappointment, disgust, distrust, frustration, fear, and sadness associated with end-user feedback. Later, an optimized deep-learning classifier will be recommended to be implemented in the ASA store to automatically process end-user feedback and identify associated emotion types.

To highlight the DL experiment, first, various DL classifiers are shortlisted to test their performance on the ChatGPT and human-annotated datasets. Next, a pre-processing step ensures the text data is clean and uniform. Next, data balancing approaches are applied to balance the instances of end-user feedback in each class. Next, various feature engineering approaches are applied to the DL classifiers to identify the best setting, resulting in better classification performance. Finally, the cross-validation approach is applied to train and validate the DL classifiers for more stable and reliable results. The details about the deep learning experiments are included below.

### 8.1. Experimental setup

To run the DL experiments, we first shortlist the potential classifiers based on their performance on the text data, which is concise. We selected long short-term memory (LSTM), bidirectional LSTM (BiLSTM), Recurrent Neural Network (RNN), Bidirectional Recurrent Neural Network (BiRNN), Convolutional Neural Network (CNN), and gated recurrent units (GRU), Bidirectional gated recurrent units (BiGRU), DL classifiers to identify their performance in identifying associated emotion types based on their better performance in requirements engineering literature (Fatima, Kanwal, Khan, et al., 2024; Ullah et al., 2023; Zhao et al., 2021). The experimental steps are elaborated below:

#### 8.1.1. Preprocessing

We applied a series of critical preprocessing steps for the DL experiment to prepare the input data for the classifiers. Firstly, any HTML tags contained in the crowd-user comments were removed to clean the text. Afterwards, URLs in the end-user comments were filtered out. All text was converted into lowercase to normalize the text and enable analysis. Special characters, punctuation, alphanumeric words, and brackets were eradicated from the textual documents to reduce noise in the data. Also, text lemmatization was applied as an additional measure to reduce words in the dataset to their root forms, contributing to enhanced DL algorithm performance (Khan, Yasin, et al., 2022; Ullah et al., 2023). This practice aligns with specified methods in requirements engineering, highlighting the significance of well-processed input data for identifying end-user emotions in the feedback.

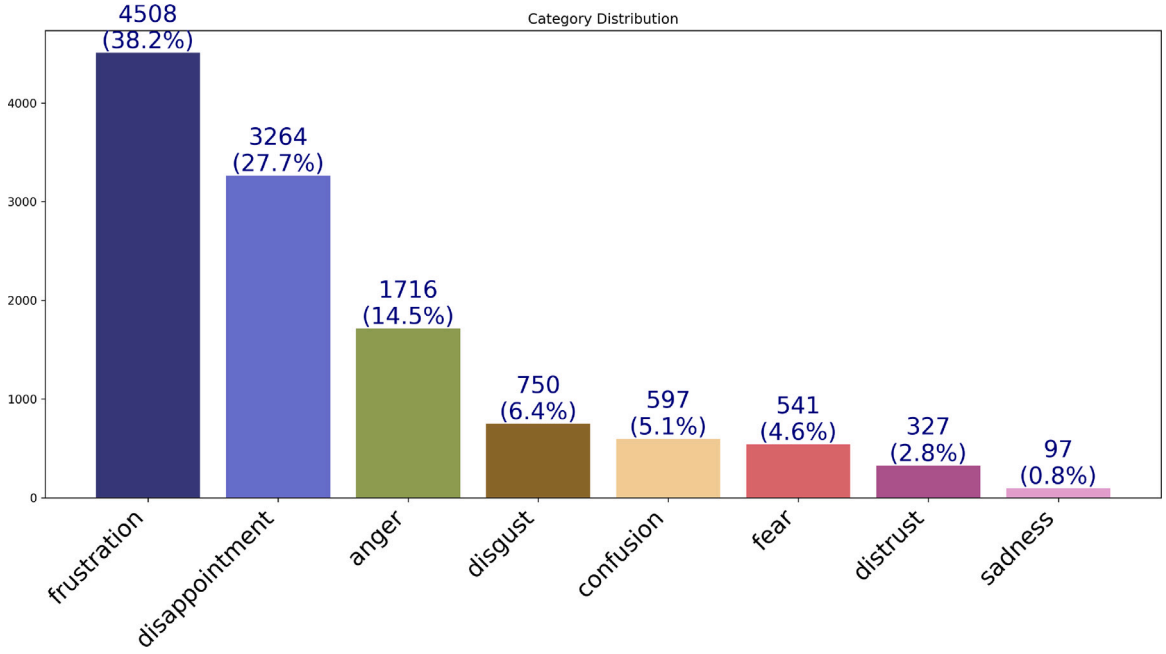



**Fig. 3.** Comparison of Category Distributions in the Annotated Dataset.

**Table 4**
Comparison of annotation frequencies between ChatGPT and human coders with highlighted disagreements.

| Emotion category | ChatGPT frequency | Human coders frequency | Disagreements |
|---|---|---|---|
| Frustration | 4780 | 3998 | 1814 |
| Disappointment | 3561 | 3144 | 1414 |
| Anger | 1835 | 1485 | 829 |
| Distrust | 239 | 192 | 173 |
| Disgust | 582 | 759 | 297 |
| Fear | 146 | 1403 | 75 |
| Confusion | 581 | 688 | 328 |
| Sadness | 76 | 131 | 33 |

*8.1.2. Feature engineering*

To make the DL experiment effective and produce better results, we identify commonly used textual features in RE literature that work well on short texts, such as end-user feedback from social media platforms (Ali Khan et al., 2020; Kurtanović & Maalej, 2018; Ullah et al., 2023). For instance, to identify the fine-tuned hyperparameters for the DL algorithms, we employed a grid search approach by specifying a range of values for the different hyperparameters. For example, to choose the fine-tuned embedding value for the DL classifiers, 16, 32, 64, and 100 values were assigned to the "output_dim", and the grid search algorithm produced better results with "100". Similarly, the grid research algorithm indicates improved results when a learning rate 0.001 is selected compared to 0.01, 0.03, and 0.1. As delineated in Table 5, the optimal configuration comprised a Max Features value of 2000 to restrict redundant and irrelevant features, help avoid overfitting, and focus on selecting important features only for faster training and improved classifier efficiency. The Max Length (maxlen) is set to 100 to ensure uniformity in the textual input length, better memory efficiency, and improved model performance. It is important because end-user feedback in the ASA store has a variable length, and to be processed by the DL classifier effectively; it must be padded (adding extra zeros to the input sentences) to make the length consistent. Where the "maxlen" feature of the DL classifier is used to identify the padding maximum length. The training process spanned across ten epochs. We supplied data into 10 epochs to ensure that the DL classifiers learn from the entire data set, generalize well, and converge to a point where the classifier training loss stabilizes, not fluctuating and decreasing. Also, with the grid search algorithm, the classifiers produce better results and generalize well when 64 neurons are selected in the dense layer. For the proposed approach, when training and testing the DL algorithms, the classifiers overfit the data, affecting their generalizability. To handle this, researchers have proposed utilizing different techniques to minimize the effect of overfitting, such as adding the dropout value, implementing regularization, or reducing the classifiers' complexity. However, when experimented with the different overfitting methods, the classifiers produced better classification results when adding dropout layers than regularization and reducing model complexity. For the proposed approach, a Dropout rate of 0.2 was instituted, resulting in better classification results. Moreover, the Embedding Layer was designed with a dimension of 100. The Adam optimizer facilitated efficient gradient descent compared to the other optimization approaches, such as Stochastic Gradient Descent (SGD) and Root Mean Squared Propagation (RMSProp). While Categorical Crossentropy' was employed as the loss function, suitable for our multi-label classification. Lastly, the softmax activation function was integrated into the output layer, delivering class probabilities.

*8.1.3. Data imbalance*

For supervised classification problems, the challenge of dealing with imbalanced datasets is of critical significance (Chawla, Japkowicz, & Kotcz, 2004). Such data imbalances derive from the inconsistent distribution of annotation classes within a given dataset, as illustrated in Fig. 3. The emotion dataset shows this imbalance when annotating end-user comments. Notably, most user comments 38.2% were categorized as expressing frustration, while only a tiny fraction 0.8% represented sadness in the annotated dataset. This ingrained imbalance

**Table 5**
Best hyperparameters for DL algorithms.

| Hyperparameter | Value |
|---|---|
| Max Features | 2000 |
| Max Length (maxlen) | 100 |
| Epochs | 10 |
| Dense Layer | 64 |
| Dropout | 0.2 |
| Learning Rate | 0.001 |
| Embedding Dimension | 100 |
| Optimizer | Adam |
| Loss Function | Categorical Crossentropy |
| Activation Function | Softmax |

can disproportionately show DL classifiers to prefer the majority class, ignoring the minority classes. Therefore, if DL classifiers are trained on an imbalanced textual dataset, they will be biased towards the majority class samples while ignoring the minority classes with fewer occurrences in the dataset. To address the issue of the unbalanced dataset, we implemented two generally used methods (Oversampling and Under-sampling) in the field of software literature (Chawla et al., 2004; Khan, Yasin, et al., 2022; Ullah et al., 2023) to make the data set balance. Two data balancing techniques are employed to enhance the performance of DL models and improve the accuracy of predictions on minority data compared to imbalanced datasets. Oversampling is a non-heuristic strategy that aims to achieve a balanced class distribution by randomly duplicating minority class examples (Chawla, Bowyer, Hall, & Kegelmeyer, 2002). For example, the SMOTE oversampling approach synthetically adds several instances to the minority class to match it with the majority class instances, such as the emotion types sadness (97 instances), fear (541 comments), etc., instances would have increased to match the majority emotion class in the dataset that is frustration comprising 4508 end-user reviews in the annotated dataset. Similarly, under-sampling is a non-heuristic strategy employed to address the issue of imbalanced class distribution by selectively excluding or eliminating samples from the majority class (Kotsiantis, Kanellopoulos, Pintelas, et al., 2006). For example, when applying the under-sampling approach to the dataset, it reduces the number of instances in the majority emotion classes, i.e., frustration (4508 end-user feedback), fear(541 review instances), etc., to the minority emotion class, i.e., sadness comprises 97 end-user comments in the annotated dataset.

In addition, to determine the most appropriate technique for balancing data (either oversampling or under-sampling) for the DL experiment, we employed Receiver Operating Characteristic (ROC) curves (Hanley & McNeil, 1982) and Precision-Recall curves (Keilwagen, Grosse, & Grau, 2014). To achieve this objective, we ascertain the proportion of True Positives (TP) relative to the proportion of False Positives (FP) for each DL classifier employed in the experiment. By this, Fig. 4(a) presents the Receiver Operating Characteristic (ROC) Curve for Sampling Methods for the LSTM Model and Fig. 4(b) for BILSTM model for the Annotated Dataset. These curves explore over and under-sampling techniques to determine the most effective re-sampling approach. These two DL classifiers were chosen as models because of their superior performance in classifying crowd-user reviews into various emotion types. The experiment results indicate that DL classifiers that employ oversampling regularly demonstrate superior performance compared to DL algorithms that utilize under-sampling. The potential reason is that under-sampling approaches synthetically lose the data instance from the majority emotional classes such as frustration, fear, etc., that might pose important information for the DL classifiers for making informed decision-making and results in a less effective learning process (Khan et al., 2024a; Khan, Yasin, et al., 2022).

#### 8.1.4. Training and evaluation of the DL algorithms

The proposed approach uses a ten-fold cross-validation approach to train and validate the DL algorithms. To train the DL classifiers, nine-fold are used to train the algorithms. At the same time, one fold of cross-validation is used to validate the algorithm. The testing and training for the proposed approach are performed iteratively ten times by rotating the training and testing folds. Using the cross-validation methodology in training and validating a supervised learning classifier offers the advantage of assessing the performance of a model under conditions of low data availability. Also, it can be a resampling technique for evaluating a model in situations where the available data is restricted in quantity. The K-fold cross-validation approach is widely used in training and validating DL classifiers (Khan et al., 2024a; Ullah et al., 2023). Every fold shows a relatively equal distribution of labels representing each class. To evaluate the efficacy of the classifiers, we calculate and report the mean results derived from the tenfold cross-validation iterations. We employed Precision (P), Recall (R), and F1-score measurements to assess the performance of the supervised DL algorithms and make comparisons. The values of P and R are calculated using the following formulas:

$$P_k = \frac{TP_k}{TP_k + FP_k} \tag{1}$$

$$R_k = \frac{TP_k}{TP_k + FN_k} \tag{2}$$

The variable P represents the proportion of true positives, which refers to the number of correctly classified end-user comments divided by the total number of crowd-users comments containing correctly and incorrectly classified user comments. Similarly, R measures the reliability of DL classifiers in accurately identifying relevant sentiment. The variable $TP_k$ represents the total of end-users whose sentiment has been accurately categorized as type $k$. Similarly, $FP_k$ represents the count of users whose sentiment has been mistakenly classified as type $k$, and $FN_k$ represents the count of users whose sentiment has been improperly classified as not being of type $k$. To clarify, the subscript "k" in the formulas refers to each specific emotion class (e.g., anger, frustration, sadness, etc.). Each class is treated as a separate classification problem, where $TP_k$, $FP_k$, and $FN_k$ refer to the true positives, false positives, and false negatives for that specific emotion class.

Additionally, the F1 score is the harmonic mean of Precision ($P_k$) and Recall ($R_k$), and is calculated using the following formula:

$$F1_k = 2 \times \frac{P_k \times R_k}{P_k + R_k} \tag{3}$$

This formula ensures a balance between precision and recall by calculating their harmonic mean, where a higher F1 score indicates that the classifier performs well in both identifying relevant user sentiment and avoiding incorrect classifications. The F1 score penalizes extreme values of either precision or recall and is especially useful when dealing with imbalanced datasets where the cost of false negatives and false positives might differ.

## 9. Results

The comparative analysis of various DL algorithms on the annotated dataset for identifying end-user emotion types is presented in Table 6. The results indicate that all the models perform well, with varying effectiveness. The LSTM and BiLSTM models stand out, achieving a high accuracy of 94%. This showcases their robustness and reliability in emotion classification, instilling confidence in their performance. The RNN model follows closely with an accuracy of 93%, demonstrating its reliability in handling sequential data. The CNN model, known for its ability to capture spatial features, achieves a slightly lower accuracy of 92%, but it excels in other metrics such as precision, recall, and F1. The GRU and BiGRU models achieve an accuracy of 91%, indicating their strong performance in managing dependencies in sequential data. The BiRNN model, while still competitive, shows a lower accuracy of

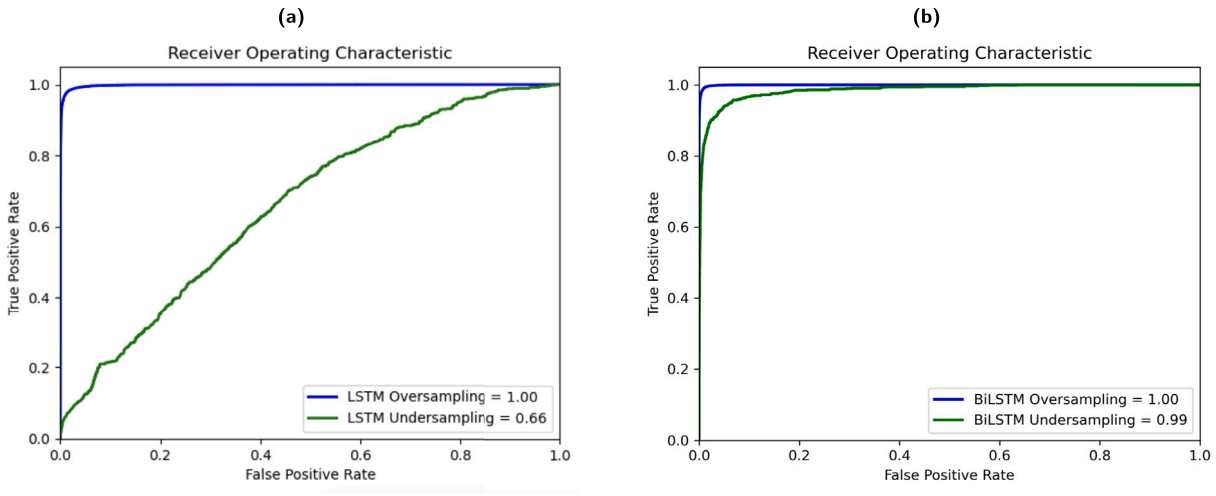



**Fig. 4.** ROC curves for oversampling and undersampling in LSTM (a) and BiLSTM (b) Models.

85%, suggesting it might not be as effective as the other models in this particular task.

Further overall accuracy, the performance of each model varies significantly across different emotion types. The CNN model consistently delivers high precision, recall, and F1 scores across most categories. For instance, it achieves 95% precision, recall, and F1 in classifying 'Anger,' and scores even higher at 99% across these metrics for 'Confusion,' 'Disgust,' and 'Frustration.' This demonstrates the CNN model's robustness in capturing and classifying nuanced emotional expressions. The LSTM model also performs well, particularly in classifying 'Fear,' where it achieves a precision of 99%, recall of 98%, and F1 of 98%. However, its performance dips slightly in classifying 'Frustration,' where it records 90% precision, 82% recall, and 86% F1. The BiLSTM model mirrors the LSTM's performance, with solid results in identifying 'Sadness,' achieving 96% precision, 99% recall, and a 97% F1.

The RNN model maintains a balanced performance, showing a moderate decline in some categories but maintaining a high precision, recall, and F1 for 'Anger.' It achieves 93% precision, 94% recall, and 93% F1 for 'Anger,' and a slightly lower but still respectable 93% precision, 90% recall, and a 91% F1 for 'Disappointment.' The GRU model outperforms the RNN in some areas, particularly in classifying 'Disgust' and 'Sadness,' where it achieves 98% and 99% precision, respectively. Similarly, the BiGRU model shows strong performance, especially in classifying 'Confusion' and 'Sadness,' with precision, recall, and F1 values of 99% across the board. The BiRNN model, although performing well in some categories, shows a slight dip in effectiveness when classifying 'Distrust' and 'Frustration,' with precision and recall values of 94% and 90%, respectively. However, it excels in classifying 'Fear,' with perfect precision of 99%, recall of 98%, and F1 of 99%, highlighting its potential in specific emotion classification tasks.

Moreover, training, validation loss, and accuracy of the LSTM and BiLSTM classifiers for the annotated dataset are shown in Fig. 5(a) and Fig. 5(b). Additionally, the receiver operating characteristic (ROC) curves for the LSTM and BiLSTM classifiers are shown in Fig. 6(a) and Fig. 6(b) for the annotated dataset. These ROC curves show the classification performance of end-user feedback for various emotional categories such as anger, confusion, disappointment, disgust, distrust, frustration, fear, and sadness.

To calculate the ROC curve for a multiclass problem, first, it needs to be converted into a binary classification problem, i.e., consider Class 0 (anger) and combine remaining classes 1, 2, 3,4,5,6, and 7 (confusion, disappointment, distrust, disgust, frustration, fear, and sadness). Thus, the True Positive Rate (TPR) is the crowd-user comments correctly classified as anger emotions by the CNN classifier of the type anger.

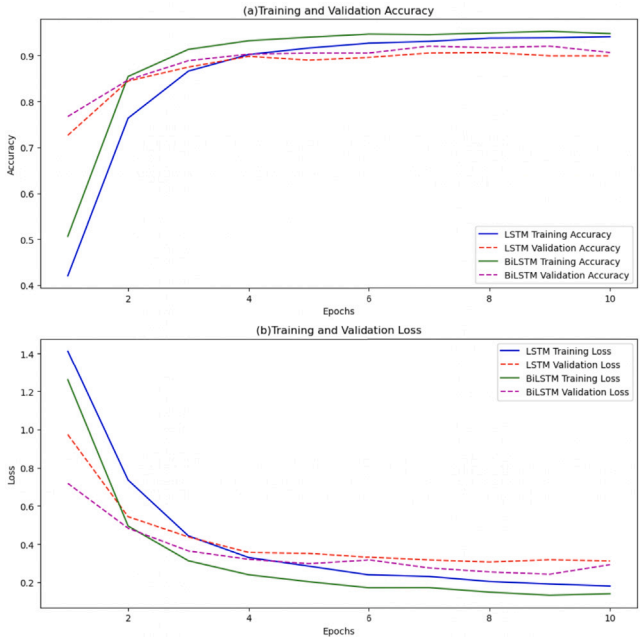



**Fig. 5.** Training and validation Accuracy and Loss of LSTM and BiLSTM classifiers for annotated datasets.

Similarly, the False Positive Rate (FPR) is the crowd-user comments in the Amazon store classified as anger arguments of the type, not anger, i.e., confusion, disappointment, distrust, disgust, frustration, fear, and sadness. The formulas to calculate FPR and TPR are

$$TPR = \frac{TP}{TP + FN} \tag{4}$$

$$FPR = \frac{FP}{TN + FP} \tag{5}$$

Additionally, the CNN classifier confusion matrix for the ChatGPT annotated dataset is depicted in Fig. 7 and the manually annotated dataset in Fig. 8. The row labels show the actual classes and the column class labels show the predicated classes by the CNN model. The purpose of the confusion matrix is to evaluate the performance of DL algorithms in classifying end-user comments into various emotion elements by summarizing the sum of correct and incorrect predictions and breaking them down into each classification class.

**Table 6**
Results comparison of DL algorithms.

| Labeled tags | DL algorithms | Precision | Recall | F1 |
|---|---|---|---|---|
| Anger | LSTM Model | 89% | 93% | 91% |
| | BiLSTM Model | 90% | 93% | 91% |
| | RNN Model | 93% | 94% | 93% |
| | CNN Model | 95% | 95% | 95% |
| | GRU Model | 95% | 95% | 95% |
| | BiGRU Model | 93% | 97% | 95% |
| | BiRNN Model | 94% | 95% | 94% |
| Confusion | LSTM Model | 97% | 98% | 97% |
| | BiLSTM Model | 97% | 98% | 97% |
| | RNN Model | 98% | 99% | 98% |
| | CNN Model | 99% | 99% | 99% |
| | GRU Model | 98% | 99% | 99% |
| | BiGRU Model | 99% | 99% | 99% |
| | BiRNN Model | 99% | 99% | 99% |
| Disappointment | LSTM Model | 90% | 85% | 87% |
| | BiLSTM Model | 90% | 84% | 87% |
| | RNN Model | 93% | 90% | 91% |
| | CNN Model | 97% | 96% | 96% |
| | GRU Model | 95% | 93% | 94% |
| | BiGRU Model | 94% | 94% | 94% |
| | BiRNN Model | 95% | 91% | 93% |
| Disgust | LSTM Model | 97% | 99% | 98% |
| | BiLSTM Model | 98% | 99% | 98% |
| | RNN Model | 98% | 99% | 98% |
| | CNN Model | 99% | 99% | 99% |
| | GRU Model | 94% | 99% | 96% |
| | BiGRU Model | 97% | 96% | 96% |
| | BiRNN Model | 97% | 99% | 98% |
| Distrust | LSTM Model | 92% | 97% | 94% |
| | BiLSTM Model | 92% | 96% | 94% |
| | RNN Model | 95% | 97% | 96% |
| | CNN Model | 94% | 99% | 96% |
| | GRU Model | 98% | 99% | 98% |
| | BiGRU Model | 99% | 98% | 98% |
| | BiRNN Model | 94% | 90% | 92% |
| Frustration | LSTM Model | 90% | 82% | 86% |
| | BiLSTM Model | 88% | 83% | 85% |
| | RNN Model | 93% | 88% | 90% |
| | CNN Model | 99% | 99% | 99% |
| | GRU Model | 98% | 99% | 99% |
| | BiGRU Model | 99% | 99% | 99% |
| | BiRNN Model | 94% | 90% | 92% |
| Fear | LSTM Model | 99% | 98% | 98% |
| | BiLSTM Model | 99% | 99% | 99% |
| | RNN Model | 97% | 99% | 98% |
| | CNN Model | 96% | 90% | 93% |
| | GRU Model | 96% | 90% | 93% |
| | BiGRU Model | 94% | 91% | 92% |
| | BiRNN Model | 99% | 98% | 99% |
| Sadness | LSTM Model | 96% | 99% | 98% |
| | BiLSTM Model | 96% | 99% | 97% |
| | RNN Model | 98% | 98% | 98% |
| | CNN Model | 99% | 98% | 98% |
| | GRU Model | 99% | 98% | 98% |
| | BiGRU Model | 99% | 99% | 99% |
| | BiRNN Model | 96% | 99% | 98% |

| Stratified K-fold Cross-Validation (Split Size = 10) | |
|---|---|
| DL Classifiers | Accuracy |
| LSTM Model | 94% |
| BiLSTM Model | 94% |
| RNN Model | 93% |
| CNN Model | 92% |
| GRU Model | 91% |
| BiGRU Model | 91% |
| BiRNN Model | 85% |

In conclusion, when comparing the performances of the optimized DL classifiers on the end-user emotions dataset, the classifiers achieved reasonably better performance in classifying end-user reviews into various emotion types. Also, for better DL classification results, an annotated data set is the primary input to the classifiers to identify the quality of results along with feature engineering and text pre-processing. Similarly, annotation is considered challenging and time-consuming, and annotators might have a second guess when annotating the end-user feedback, which makes the annotation ambiguous. To semi-automate and validate the manual annotation process, ChatGPT

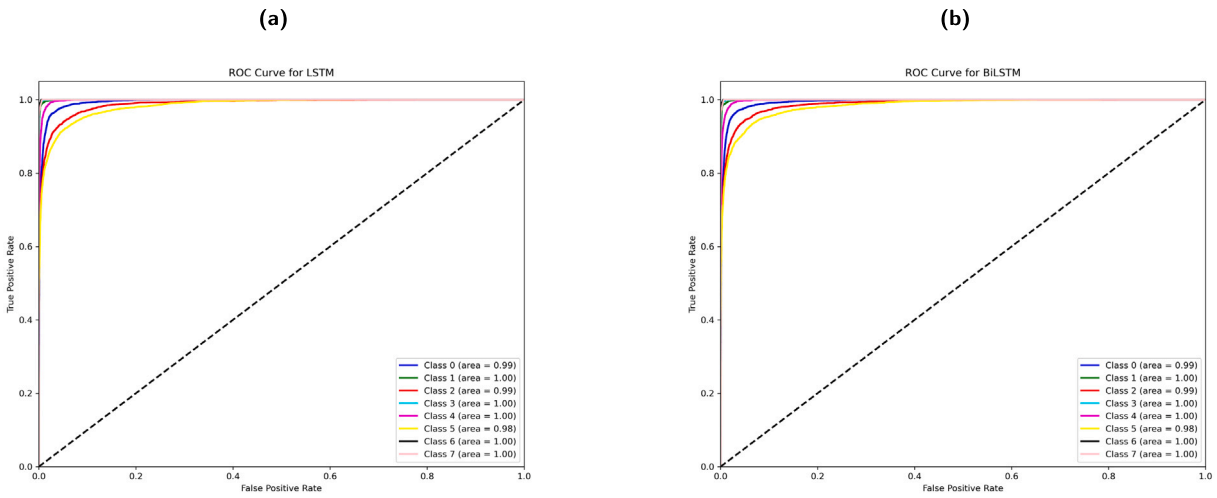



**Fig. 6.** ROC curves of LSTM (a) and BiLSTM (b) Classifiers for Annotated Dataset.

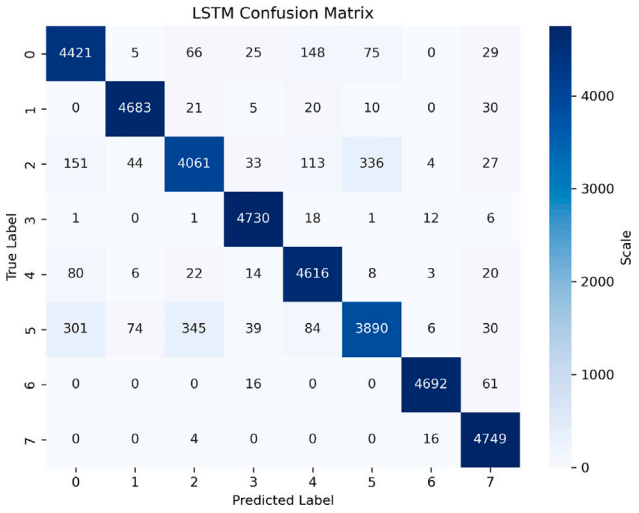



**Fig. 7.** Confusion matrix generated by the LSTM classifier(Class 0 = anger, Class 1 = confusion, Class 2 = disappointment, Class 3 = distrust, Class 4 = disgust), Class 5 = frustration), Class 6 = fear), Class 7 = Sadness).

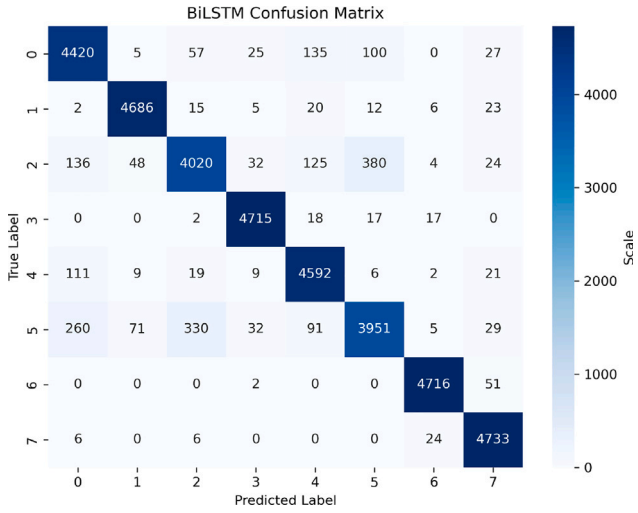



**Fig. 8.** Confusion matrix generated by the BiLSTM classifier (Class 0 = anger, Class 1 = confusion, Class 2 = disappointment, Class 3 = distrust, Class 4 = disgust), Class 5 = frustration), Class 6 = fear), Class 7 = Sadness).

can be employed as an alternative source, which uses the power of LLMs by identifying the hidden inferences between the end-user feedback and various emotion types, making the annotation process more reliable for the DL classifiers, resulting in better classification results. ChatGPT can also be employed as a negotiator to resolve the conflicts between the ChatGPT and human-annotated dataset, resulting in a conflict-free dataset. Also, when comparing the different DL algorithm's performance, LSTM and BiLSTM classifiers outperform other classifiers when classifying end-user feedback into various emotion types on annotated datasets. Therefore, the LSTM algorithm with optimized textual features can selected as the best classifier to identify end-user emotions associated with the feedback. Moreover, the results show that ChatGPT can be an alternative source for annotating end-user feedback. However, it is essential to emphasize the variability and potential randomness of the ChatGPT by validating their annotation with software experts to achieve consensus and remove conflicts in annotating end-user comments with the specific emotion types. Moreover, a few rounds of discussion might be needed on the justification of assigning specific emotional types to the end-user feedback in the dataset.

## 10. Discussion

The primary focus of this study is to examine the negative end-user feedback across various issues and bugs in low-ranked software applications on the ASA store. The proposed research is particularly interested in identifying the emotions associated with end-user feedback and their implications for data-driven Requirements Engineering (RE). We have incorporated Generative AI, specifically ChatGPT, to validate and negotiate the manual content analysis process. In the following sections, we will explore various dimensions of the applicability of the proposed approach.

### *10.1. The importance of understanding end-user opinions*

Understanding end-users opinions and emotions is considered an essential component of software development and evolution, particularly for low-ranking applications against which users submit feedback concerning issues, problems, and bugs. Previous research has emphasized the importance of user reviews in RE and software development, suggesting that they offer a rich source of data for understanding bugs, issues, and even feature requests that could be crucial for the software's

evolution (Khan et al., 2020; Kurtanović & Maalej, 2018). Similarly, the end-user emotions associated with the feedback in the ASA store offer a valuable source of knowledge that can be utilized for several aspects of software development. For example, understanding end-user emotions helps software applications evolve according to the preferences and needs of the users to make them more user-centric, impacting user satisfaction and overall software success. Satisfied and happy end-users can be brand ambassadors for the software application, recommending it to other users, which helps improve the software ratings. Also, recovering end-user disappointment, frustration, or anger emotions can help software developers highlight the pain points, which provides opportunities to improve the software quality. Furthermore, identifying end-user emotion types and incorporating them into the software application could provide a competitive advantage, as emotionally intelligent software can attract more users by standing standalone in the market. However, the challenge often lies in interpreting these large number of end-user reviews freely available on social media platforms. The proposed research addresses this challenge by implementing an automatic review analysis process for low-ranked software applications to identify emotion types and use them in the software evolution process to improve the software ratings by providing remedies to the pain points. This will help achieve higher user satisfaction and retain the user, improving the software's business values.

### 10.2. Efficacy of generative AI in requirements engineering

Incorporating Generative AI, specifically the ChatGPT model, into RE is a significant advancement in automated software engineering, as demonstrated in the proposed research study. Previously, RE researchers often relied on human annotators to prepare a dataset for the supervised machine or deep learning classifiers by analyzing end-user feedback in the social media platforms, which can be time-consuming, challenging and subject to human biases (Khan, Yasin, et al., 2022; Mezouar et al., 2018). The ChatGPT can be used as an alternative source to annotate end-user feedback for the classification tasks that can validate the human-annotated datasets. Additionally, ChatGPT can serve as a negotiator to resolve conflicts between human annotators and ChatGPT to develop a conflict-free dataset to evaluate the performance of DL classifiers. ChatGPT simplifies the annotation process by adding a layer of neutrality that human annotators may need to improve. The AI's capacity to effectively manage an extensive number of end-user comments and classify them into different emotional responses shows its capacity for resilience and effectiveness in emotional computing. However, it is essential to emphasize the variability and potential randomness of the ChatGPT by validating their annotation with software experts to achieve consensus and remove conflicts in annotating end-user comments. DL classifiers are observed to achieve comparatively better accuracy with the dataset developed using ChatGPT and human annotation. This improved accuracy validates the use of Generative AI in the loop with human experts for requirements and software engineering-related tasks. Such approaches could quickly become standard tools for analyzing and interpreting end-user feedback for improved performances. Automating these processes could lead to more efficient, accurate, and data-driven approaches in software development, thereby enhancing the quality of software products. Additionally, end-user feedback in app stores, Reddit forums, Twitter, etc., are sometimes lengthy, containing extra information not important for the software developers and vendors. Therefore, software researchers can harness the power of Generative AI to summarize the lengthy end-user feedback on these social media platforms, making it easier and more interesting for the developers to concentrate on the point of concern.

### 10.3. ChatGPT as annotator and negotiator

Machine and deep learning-based approaches often confront the challenges of human biases and inconsistencies when annotating datasets related to end-user feedback analysis (Khan et al., 2024a; Khan, Yasin, et al., 2022; Ullah et al., 2023). Despite human expertise, human annotators can often apply subjective interpretations when classifying end-user feedback into various requirements-related information. These inconsistencies can affect the overall reliability of the feedback analysis and might impact the performances of the deep and machine learning classifiers. However, ChatGPT can be used as an annotator and negotiator to address these issues and reconcile discrepancies between human and AI-generated annotations for improved classification results. This dual role helped to reduce biases and ensure greater consistency in categorizing emotional and other requirements of engineering content for software evolution. ChatGPT provided a systematic approach that improved the objectivity of the analysis, offering a scalable and reliable alternative to traditional human-driven methods. This approach not only underscores the potential of AI tools in enhancing the accuracy and efficiency of user feedback analysis in software development but also empowers to confidently navigate the future of this field.

### 10.4. Emotional categories and software development

Identifying different emotional categories in end-user reviews is a significant contribution of the proposed study, providing a slight understanding of past feature requests (Guzman & Maalej, 2014; Khan, Yasin, et al., 2022) or bug reports (Khan et al., 2021; Stanik, Haering, & Maalej, 2019). Emotional categories like anger, confusion, disgust, distrust, disappointment, fear, frustration, and sadness offer a deeper layer of context that can be valuable for software developers and vendors in better understanding the requested requirements-related information. Understanding these emotional elements can help prioritize issues that need immediate attention. For instance, issues that generate strong negative emotions, such as anger and disappointment, may indicate the user's critical discomfort points and thus require urgent fixes to improve user satisfaction and overall app ratings (Cabrio & Villata, 2013). Emotional categorization not only aids in immediate issue resolution but also provides long-term insights into software evolution. By understanding the emotional context behind user feedback, developers can make more informed decisions during the software development lifecycle. This emotional understanding can be particularly beneficial for low-rated apps that need functionality and user experience improvements. The emotional categories serve as a guidepost for developers, helping them to navigate the complex landscape of user needs and anticipations, thereby aligning software improvements more closely with user sentiment (Khan et al., 2024, 2020).

### 10.5. Threats to validity

One of the primary limitations of this study is its focus on Amazon's app store, which may not provide a comprehensive view of the software application landscape. While Amazon's platform is indeed significant, the findings may only be somewhat generalizable to other app stores or types of software. This limitation could affect the study's external validity, as the emotional categories and user manners identified may change in other backgrounds (Khan et al., 2020; Kurtanović & Maalej, 2018). Additionally, the proposed approach employs the ChatGPT to partially automate, validate, and negotiate the manual annotation process. Although, we analyzed the ChatGPT annotation with human experts. Still, further evaluation by human experts is needed to validate the results of ChatGPT for annotating and negotiating end-user feedback for software evolution. Although the proposed approach showed high accuracy, precision, and recall performance, it is worth noting that machine learning models, including ChatGPT, are not entirely free

from biases and errors. These biases could introduce anticipation into the findings, potentially affecting the study's internal validity (Rogers, Gung, Qiao, & Burge, 2012). To mitigate this limitation of the proposed approach, we aim to use the existing explainable AI approach to make the process more reliable and transparent. Another factor to consider is the emotional categories identified in the analysis. While the research does an excellent job of categorizing end-user emotions like anger, confusion, disgust, distrust, disappointment, fear, frustration, and sadness, this list may be incomplete. Emotional responses can be complex and multi-faceted, and the analysis may not capture the full range of end-user emotions, which could limit its applicability in broader contexts (Peldszus & Stede, 2013). Furthermore, the study assumes that the proposed approach's high performance in classifying these emotions implies trustworthiness. However, the model's performance could change with different datasets or under different conditions, introducing another potential threat to the study's validity. Therefore, while the study offers valuable insights, these limitations should be considered when interpreting the findings and applying them to software development practices. Additionally, the proposed approach is limited by processing end-user reviews at a review level. However, it is evident from the proposed study that end-user reviews represent more than one type of emotion. Therefore, we aim to conduct further experiments in future with the existing DL approaches at a sentence level to compare the performances with the review level. Also, we aim to conduct multi-label classification with the existing DL classifiers to better represent the end-user-associated emotions.

## 11. Conclusion and future work

This research has significantly advanced the field of user feedback analysis for software evolution, focusing on applications that have received low ratings. We proposed an automated approach that utilizes Generative AI, specifically ChatGPT, to analyze end-user comments from low-rated software applications in the Amazon store. This approach involves ChatGPT as both an annotator and negotiator, partially automating the annotation process and minimizing the biases often present in manual methods. The primary goal of this approach was to identify end-user viewpoint categories associated with negative evaluations, such as anger, confusion, disappointment, disgust, distrust, fear, frustration, and sadness. The proposed approach demonstrated superior performance by applying various DL classifiers, including LSTM, BILSTM, CNN, RNN, GRU, BiGRU and BiRNN. These classifiers were chosen for their ability to effectively process sequential data and capture the nuances of user emotions. We achieved average accuracies of 94%, 94%, 93%, 92%, 91%, 91%, and 85% respectively. These results highlight the effectiveness of using AI-driven annotation to accurately capture the complex emotions expressed in user feedback, which is essential for developers and vendors looking to improve their products based on genuine user experiences. Moreover, the findings emphasize the critical importance of understanding end-user emotions in software development, supporting insights from previous research in this domain (Alkadhi, Lata, Guzmany, & Bruegge, 2017; Khan et al., 2020; Kurtanović & Maalej, 2018). Compared to prior studies that often focus on well-rated or average software applications, the proposed approach targets low-rated software, thereby addressing a significant gap in the literature. This focus on low-rated applications gives developers unique insights into users' most pressing issues and frustrations, enabling more targeted and effective software improvements. In conclusion, the proposed approach not only enhances the methodological approach to analyzing user feedback but also offers a powerful tool for software developers to understand better and respond to the needs and emotions of their users. This ultimately contributes to the development of more user-centric and high-quality software products.

Furthermore, there are many opportunities for additional research in the future. First, the scalability of the proposed approach needs to be tested by collecting more diverse user feedback across different software ratings and categories in the Amazon store. This would provide a more exhaustive interpretation of user requirements and emotions by understanding more diverse end-user feedback that might include other emotion types. In the future, we aim to identify key stakeholders who frequently contribute emotion and requirements-related information in the Amazon Play store work (Khan, Khan, et al., 2022). It will help software developers to listen to the end-users who regularly contribute pivotal information related to software evolution in a timely manner. Also, developing tools that incorporate the research's conclusions into the software development process could be highly beneficial. This would allow requirement engineers and software developers for real-time updates and improvements, improving user satisfaction and software quality. Additionally, exploring deceptive reviews, as indicated in prior research (Marwat et al., 2022), could add another layer of depth to understanding user feedback. By pursuing these future directions, we aim to provide a more powerful, automated, real-time system for improving software quality based on user feedback. Furthermore, an essential area of research interest is redesigning and restructuring crowd-user comments and conversations on social media platforms (Khan et al., 2021; Kurtanović & Maalej, 2018; Panichella et al., 2015). In the future, we aim to identify the positive emotions associated with end-user feedback; this can motivate software developers and vendors, encouraging them to further improve the existing features and services of software applications. Additionally, we aim to explore the performance of the DL classifiers by supplying datasets curated at the sentence level and comparing their performance at the review level.

## CRediT authorship contribution statement

**Nek Dil Khan:** Developed the method, Detailed investigation, Manuscript writing. **Javed Ali Khan:** Developed the method, Detailed investigation, Manuscript writing. **Jianqiang Li:** Revised the methodology, Supervised revised the manuscript writing. **Tahir Ullah:** Curated the research data set. **Qing Zhao:** Revised the methodology, Supervised, Revised the manuscript writing.

## Ethical approval:

This article does not contain any examinations with human members or creatures performed by any of the others.

## Funding

The authors have not disclosed any funding.

## Declaration of competing interest

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

## Acknowledgments

All authors have read and agreed to the published version of the manuscript.

## Data availability

Access to the data sets and replication information is available through this link https://github.com/nekdil566/Experimental-work-and-datasets.

## References


Acypreste, R. D., & Paraná, E. (2022). Artificial intelligence and employment: a systematic review. URL https://scite.ai/reports/10.1590/0101-31572022-3320.

AG (2023). Amazon appstore statistics and trends 2023. https://42matters.com/amazon-appstore-statistics-and-trends, (Accessed 03 October 2023).

Ali Khan, J., Liu, L., Wen, L., & Ali, R. (2020). Conceptualising, extracting and analysing requirements arguments in users' forums: The crowdre-arg framework. *Journal of Software: Evolution and Process*, *32*(12), Article e2309.

Alkadhi, R., Lata, T., Guzmany, E., & Bruegge, B. (2017). Rationale in development chat messages: an exploratory study. In *2017 IEEE/ACM 14th international conference on mining software repositories* (pp. 436–446). IEEE.

Alvertis, I., Koussouris, S., Papaspyros, D., Arvanitakis, E., Mouzakitis, S., Franken, S., et al. (2016). User involvement in software development processes. *Procedia Computer Science*, *97*, 73–83.

Beganovic, A., Jaber, M. A., & Abd Almisreb, A. (2023). Methods and applications of ChatGPT in software development: a literature review. *Southeast Europe Journal of Soft Computing*, *12*(1), 08–12.

Belal, M., She, J., & Wong, S. (2023). Leveraging chatgpt as text annotation tool for sentiment analysis. arXiv preprint arXiv:2306.17177.

Bencheikh, L., & Höglund, N. (2023). Exploring the efficacy of chatgpt in generating requirements: An experimental study.

Cabrio, E., & Villata, S. (2013). A natural language bipolar argumentation approach to support users in online debate interactions. *Argument & Computation*, *4*(3), 209–230.

Carreño, L. V. G., & Winbladh, K. (2013). Analysis of user comments: an approach for software requirements evolution. In *2013 35th international conference on software engineering* (pp. 582–591). IEEE.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*, 321–357.

Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, *6*(1), 1–6.

Cohen, J. (1968). Weighted kappa: nominal scale agreement provision for scaled disagreement or partial credit. *Psychological Bulletin*, *70*(4), 213.

Corbin, J., & Strauss, A. (2008). Basics of qualitative research: Techniques and procedures for developing grounded theory,(3e ed.) sage. Thousand Oaks, California.

Dąbrowski, J., Letier, E., Perini, A., & Susi, A. (2020). Mining user opinions to support requirement engineering: An empirical study. *Lecture Notes in Computer Science*, 401–416, URL https://link.springer.com/content/pdf/10.1007%2F978-3-030-49435-3_25.pdf.

Dąbrowski, J., Letier, E., Perini, A., & Susi, A. (2022). Mining user feedback for software engineering: Use cases and reference architecture. In *2022 IEEE 30th international requirements engineering conference* (pp. 114–126). IEEE.

Deng, J., & Chen, X. (2021). Research on artificial intelligence interaction in computer-aided arts and crafts. URL https://scite.ai/reports/10.1155/2021/5519257.

Ebert, C., & Louridas, P. (2023). Generative AI for software practitioners. *IEEE Software*, *40*(4), 30–38. http://dx.doi.org/10.1109/MS.2023.3265877.

Ekman, P. E., & Davidson, R. J. (1994). *The nature of emotion: Fundamental questions*. Oxford University Press.

Fatima, E., Kanwal, H., Khan, J., et al. (2024). An exploratory and automated study of sarcasm detection and classification in app stores using fine-tuned deep learning classifiers. *Automated Software Engineering*, *31*, 69. http://dx.doi.org/10.1007/s10515-024-00468-3.

Fischer, R., Luczak-Roesch, M., & Karl, J. A. (2023). What does ChatGPT return about human values? Exploring value bias in ChatGPT using a descriptive value theory. arXiv preprint arXiv:2304.03612.

Gao, C., Zeng, J., Lyu, M. R., & King, I. (2018). Online app review analysis for identifying emerging issues. In *Proceedings of the 40th international conference on software engineering* (pp. 48–58).

Groen, E. C., Seyff, N., Ali, R., Dalpiaz, F., Doerr, J., Guzman, E., et al. (2017). The crowd in requirements engineering: The landscape and challenges. *IEEE software*, *34*(2), 44–52.

Guo, B., Zhang, X., Wang, Z., Jiang, M., Nie, J., Ding, Y., et al. (2023). How close is ChatGPT to human experts? Comparison corpus, evaluation, and detection. URL https://arxiv.org/abs/2301.07597v1.

Guzman, E., & Maalej, W. (2014). How do users like this feature? a fine grained sentiment analysis of app reviews. In *2014 IEEE 22nd international requirements engineering conference* (pp. 153–162). Ieee.

Haering, M., Stanik, C., & Maalej, W. (2021). Automatically matching bug reports with related app reviews. In *2021 IEEE/ACM 43rd international conference on software engineering* (pp. 970–981). IEEE.

Han, J., Han, J., et al. (2021). Evaluation of artificial intelligence techniques applied in watson and alphago. *Academic Journal of Computing & Information Science*, *4*(8), 29–36.

Hanley, J. A., & McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, *143*(1), 29–36.

Hou, T., Yannou, B., Leroy, Y., & Poirson, E. (2019). Mining customer product reviews for product development: A summarization process. *Expert Systems with Applications*, *132*, 141–150.

Hou, X., Zhao, Y., Liu, Y., Yang, Z., Wang, K., Li, L., et al. (2023). Large language models for software engineering: A systematic literature review. arXiv preprint arXiv:2308.10620.

Hutto, C., & Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. *8*, In *Proceedings of the international AAAI conference on web and social media* (1), (pp. 216–225).

Jeong, J., & Lee, Y. K. (2022). Identifying temporal corpus for enhanced user comments analysis. *International Journal of Software Engineering and Knowledge Engineering*, *32*(03), 439–456.

Keilwagen, J., Grosse, I., & Grau, J. (2014). Area under precision-recall curves for weighted and unweighted data. *PLoS One*, *9*(3), Article e92209.

Khan, F. M., Khan, J. A., Assam, M., Almasoud, A. S., Abdelmaboud, A., & Hamza, M. A. M. (2022). A comparative systematic analysis of stakeholder's identification methods in requirements elicitation. *IEEE Access*, *10*, 30982–31011.

Khan, N. D., Khan, J. A., Li, J., Ullah, T., Alwadain, A., Yasin, A., et al. (2024). How do crowd-users express their opinions against software applications in social media? A fine-grained classification approach. *IEEE Access*.

Khan, N. D., Khan, J. A., Li, J., Ullah, T., & Zhao, Q. (2024a). Mining software insights: uncovering the frequently occurring issues in low-rating software applications. *Peer Journal of Computer Sciences*, *10*, Article e2115. http://dx.doi.org/10.7717/peerj-cs.2115.

Khan, J. A., Liu, L., & Wen, L. (2020). Requirements knowledge acquisition from online user forums. *Iet Software*, *14*(3), 242–253.

Khan, J. A., Liu, L., Wen, L., & Ali, R. (2019). Crowd intelligence in requirements engineering: Current status and future directions. In *International working conference on requirements engineering: foundation for software quality* (pp. 245–261). Springer.

Khan, J. A., Xie, Y., Liu, L., & Wen, L. (2019). Analysis of requirements-related arguments in user forums. In *2019 IEEE 27th international requirements engineering conference* (pp. 63–74). IEEE.

Khan, J. A., Yasin, A., Assam, M., Khan, W., Shah, S. Y., Khan, R. A., et al. (2021). Requirements decision-making as a process of argumentation: A google maps case study with goal model. *International Journal of Innovations in Science & Technology*, *3*(4), 15–33.

Khan, J. A., Yasin, A., Fatima, R., Vasan, D., Khan, A. A., & Khan, A. W. (2022). Valuating requirements arguments in the online user's forum for requirements decision-making: The crowdre-varg framework. *Software - Practice and Experience*, *52*(12), 2537–2573.

Kifetew, F. M., Perini, A., Susi, A., Siena, A., Muñante, D., & Morales-Ramirez, I. (2021). Automating user-feedback driven requirements prioritization. *Information and Software Technology*, *138*, Article 106635.

Kotsiantis, S., Kanellopoulos, D., Pintelas, P., et al. (2006). Handling imbalanced datasets: A review. *GESTS international transactions on computer science and engineering*, *30*(1), 25–36.

Kumar, K. S., Desai, J., & Majumdar, J. (2016). Opinion mining and sentiment analysis on online customer review. In *2016 IEEE international conference on computational intelligence and computing research* (pp. 1–4). IEEE.

Kurtanović, Z., & Maalej, W. (2018). On user rationale in software engineering. *Requirements Engineering*, *23*(3), 357–379.

Lee, J. Y. (2020). User review mining: An approach for software requirements evolution. *International Journal of Advanced Smart Convergence*, *9*(4), 124–131.

Ligthart, A., Catal, C., & Tekinerdogan, B. (2021). Systematic reviews in sentiment analysis: a tertiary study. *Artificial Intelligence Review*, 1–57.

Lim, S., Henriksson, A., & Zdravkovic, J. (2021). Data-driven requirements elicitation: A systematic literature review. *SN Computer Science*, *2*, 1–35.

Lin, B., Cassee, N., Serebrenik, A., Bavota, G., Novielli, N., & Lanza, M. (2022). Opinion mining for software development: a systematic literature review. *ACM Transactions on Software Engineering and Methodology*, *31*(3), 1–41.

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, *5*(1), 1–167.

Maalej, W., Kurtanović, Z., Nabil, H., & Stanik, C. (2016). On the automatic classification of app reviews. *Requirements Engineering*, *21*(3), 311–331.

Maalej, W., & Robillard, M. P. (2013). Patterns of knowledge in API reference documentation. *IEEE Transactions on Software Engineering*, *39*(9), 1264–1282.

Manole, E.-M. (2021). Detecting the most important classes from software systems with self organizing maps. URL https://scite.ai/reports/10.24193/subbi.2021.1.04.

Marwat, M. I., Khan, J. A., Alshehri, D. M. D., Ali, M. A., Ali, H., Assam, M., et al. (2022). Sentiment analysis of product reviews to identify deceptive rating information in social media: A SentiDeceptive approach. *KSII Transactions on Internet and Information Systems*, *16*(3), 830–860.

Mezouar, M. E., Zhang, F., & Zou, Y. (2018). Are tweets useful in the bug fixing process? an empirical study on firefox and chrome. *Empirical Software Engineering*, *23*(3), 1704–1742.

Nayebi, M., Dicke, L., Ittyipe, R., Carlson, C., & Ruhe, G. (2018). Essmart way to manage user requests. arXiv preprint arXiv:1808.03796.

Neuendorf, K. A. (2017). *The content analysis guidebook*. sage.

Nurrohmat, M. A., & Azhari, S. (2019). Sentiment analysis of novel review using long short-term memory method. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, *13*(3), 209–218.

Obaidi, M., Nagel, L., Specht, A., & Klünder, J. (2022). Sentiment analysis tools in software engineering: A systematic mapping study. *Information and Software Technology*, *151*, Article 107018.

Panichella, S., Di Sorbo, A., Guzman, E., Visaggio, C. A., Canfora, G., & Gall, H. C. (2015). How can I improve my app? Classifying user reviews for software maintenance and evolution. In *2015 IEEE international conference on software maintenance and evolution* (pp. 281–290). IEEE.

Peldszus, A., & Stede, M. (2013). From argument diagrams to argumentation mining in texts: A survey. *International Journal of Cognitive Informatics and Natural Intelligence*, *7*(1), 1–31.

Rodríguez-Ibánez, M., Casánez-Ventura, A., Castejón-Mateos, F., & Cuenca-Jiménez, P.-M. (2023). A review on sentiment analysis from social media platforms. *Expert Systems with Applications*, Article 119862.

Rogers, B., Gung, J., Qiao, Y., & Burge, J. E. (2012). Exploring techniques for rationale extraction from existing documents. In *2012 34th international conference on software engineering* (pp. 1313–1316). IEEE.

Rudolph, J., Tan, S., & Tan, S. (2023). ChatGPT: Bullshit spewer or the end of traditional assessments in higher education? *Journal of Applied Learning and Teaching*, *6*(1), 342–363.

Russo, D. (2023). Navigating the complexity of generative ai adoption in software engineering. arXiv preprint arXiv:2307.06081.

Stanik, C., Haering, M., & Maalej, W. (2019). Classifying multilingual user feedback using traditional machine learning and deep learning. In *2019 IEEE 27th international requirements engineering conference workshops* (pp. 220–226). IEEE.

Strauss, A., & Corbin, J. (1998). Basics of qualitative research techniques. Thousand oaks, CA: Sage publications.

Ullah, T., Khan, J. A., Khan, N. D., Yasin, A., & Arshad, H. (2023). Exploring and mining rationale information for low-rating software applications. *Soft Computing*, 1–26.

Villarroel, L., Bavota, G., Russo, B., Oliveto, R., & Di Penta, M. (2016). Release planning of mobile apps based on user reviews. In *2016 IEEE/ACM 38th international conference on software engineering* (pp. 14–24). IEEE.

Wang, X., & Liu, S. (2014). Computer-aided formalization of requirements based on patterns. URL https://scite.ai/reports/10.1587/transinf.e97.d.198.

Wankhade, M., Rao, A. C. S., & Kulkarni, C. (2022). A survey on sentiment analysis methods, applications, and challenges. *Artificial Intelligence Review*, *55*(7), 5731–5780.

Zhao, L., Alhoshan, W., Ferrari, A., Letsholo, K. J., Ajagbe, M. A., Chioasca, E.-V., et al. (2021). Natural language processing for requirements engineering: A systematic mapping study. *ACM Computing Surveys*, *54*(3), 1–41.

Zimmermann, T., Premraj, R., Bettenburg, N., Just, S., Schroter, A., & Weiss, C. (2010). What makes a good bug report? *IEEE Transactions on Software Engineering*, *36*(5), 618–643.