The Journal of Systems and Software 213 (2024) 112040

Contents lists available at ScienceDirect

The Journal of Systems & Software

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

How to effectively mine app reviews concerning software ecosystem? A survey of review characteristics☆

Xiaohui Wang [a], Tao Zhang [a,*], Youshuai Tan [a], Weiyi Shang [b], Yao Li [a]

[a] School of Computer Science and Engineering, Macau University of Science and Technology, Macao Special Administrative Region of China
[b] Department of Electrical and Computer Engineering, University of Waterloo, Ontario, Canada

ARTICLE INFO

Keywords:
App review
Literature review
Software ecosystem

ABSTRACT

App reviews in app stores offer valuable insights into many activities in the software ecosystem, e.g., software development, app marketing, security. As app reviews are known to be error-prone, commonly short, dynamic, and to hold domain-specific knowledge, we need mining strategies tailored to these characteristics. To help developers or researchers mine reviews more effectively, in this study, we conduct a systematic literature review on app review mining from the perspective of the characteristics. This survey was conducted on 167 papers published between 2012 and 2022 and focuses on three phases in app review mining: (a) the 167 papers were thoroughly examined to extract practices for the collection of app reviews; (b) a detailed list of review characteristics was summarized through a key-point investigation; (c) the survey presents common handling and applications for each review characteristic. Compared with other literature reviews on app review mining, our paper provides insights from the micro perspective. We have noted a growing trend in the analysis of app reviews, with review rating being the most frequently employed review characteristic. We also observed that the Google Play Store stands out as the most commonly used app distribution platform, and simple random sampling prevails as the most popular review sampling strategy compared to stratified sampling and key-point investigation. Additionally, we have identified domain knowledge, textual content, dynamic nature, and sentiment of reviews as the most promising review characteristics for future studies.

1. Introduction

Popular mobile app stores, such as Google Play, allow users to share their opinions regarding the apps they acquire from these platforms through the submission of user reviews. Despite that this mechanism aims at recommending apps among users via sharing user experience and sentiment, these reviews also play an important role in mobile software development (Scalabrino et al., 2019).

App reviews serve as the "voice of the users", thus, beneficial for steering development efforts and improve future app updates from the users' perspective. Prior studies (Khalid, 2013; Fu et al., 2013; Kong et al., 2015; Martens and Maalej, 2019b) showed that app reviews include relevant information for software ecosystem, concerning *Software Development*, *Software Security*, and *Competing Ecosystem*. Moreover, users actually describe a new problem in the app reviews before developers notice and record it in the issue tracker (Haering et al., 2021). Hence, developers can benefit from app reviews during the app development process. Besides, app stores serve as an essential channel between app developers and users, which not only allows users to provide feedback about apps, but also supports developers to respond to users via commenting on reviews. Fig. 1 shows an example of an app review and relevant information in Google Play for *Pandora - Music & Podcasts*, which includes the app review and its metadata, other users' reactions, and a developer's response to this user review. Even though app reviews provide valuable information for the software ecosystem, there is no universal methodology to effectively mine app reviews.

In this paper, we suggest researchers, developers, and other stakeholders mine app reviews from the perspective of app review characteristics. The reasons are twofold. On the one hand, characteristics of app reviews provide information about how app users leave feedback (Vasa et al., 2012; Pagano and Maalej, 2013), and the challenges of mining app reviews may result from these characteristics. For example, app reviews are frequently edited using mobile phones, with limited screen space and typing constraints. As a result, these reviews are error-prone and often lack adherence to grammar rules. This poses a significant

☆ Editor: Heiko Koziolek.

* Corresponding author.
E-mail addresses: 2109853wim20002@student.must.edu.mo (X. Wang), tazhang@must.edu.mo (T. Zhang), 2109853jim20001@student.must.edu.mo (Y. Tan), wshang@uwaterloo.ca (W. Shang), 2109853gia30001@student.must.edu.mo (Y. Li).

https://doi.org/10.1016/j.jss.2024.112040
Received 8 August 2023; Received in revised form 31 January 2024; Accepted 27 March 2024
Available online 30 March 2024

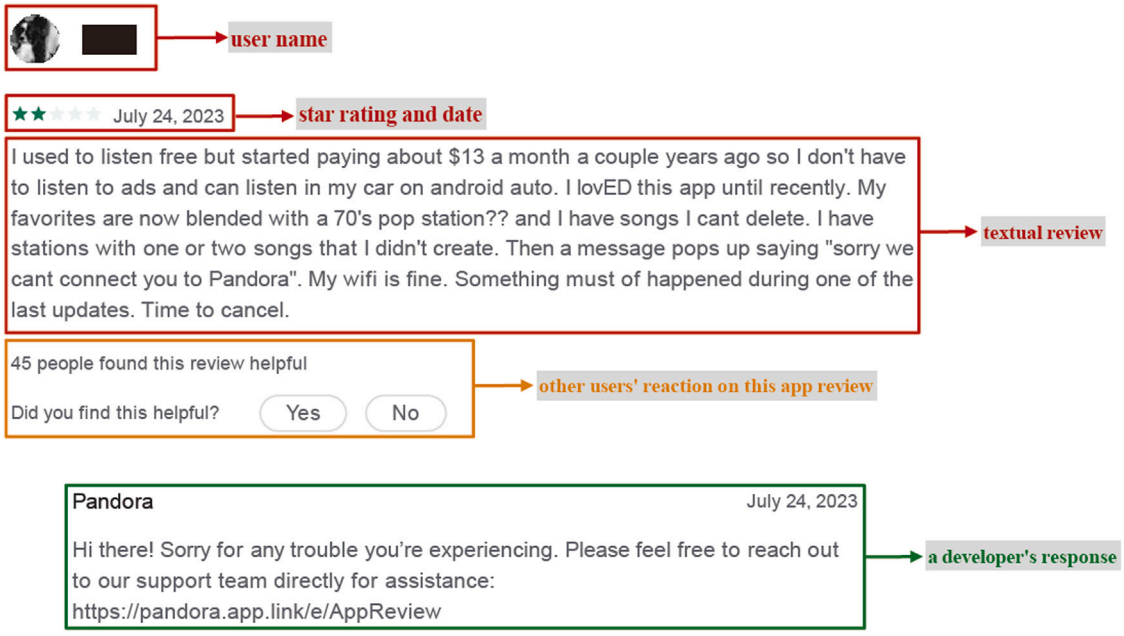

Fig. 1. An example of an app review and relevant information for *Pandora - Music & Podcasts* in Google Play.
Note: We have withheld the reviewer's name for the preservation of privacy.

challenge for developers to extract the information they need. On the other hand, though app review mining is essentially text mining, not all text mining methods are suitable for review mining. As Petz et al. (2013) suggest, there is no "one-size-fits-all" approach for all types of text due to the differences between their unique characteristics. Fundamentally, developers and researchers need to have a good knowledge of these characteristics of app reviews. Then, they are able to adopt appropriate analysis tools when analyzing reviews. In contrast, inappropriate analysis tools can be detrimental to the analysis results. Moreover, these characteristics are commonly used indicators/features when analyzing app reviews, and a comprehensive understanding of them can help researchers better make use of app reviews, so as to better help app developers in diverse phases of software development and evolution.

To enhance the efficiency of mining app reviews for developers and researchers, this study performs a comprehensive literature review focusing on the characteristics of app review mining. The survey encompasses 167 papers published between 2012 and 2022 and concentrates on three key phases in app review mining. Firstly, we thoroughly examined the 167 papers to extract common practices for collecting app reviews. Secondly, we summarized a detailed list of review characteristics through a key-point investigation. Lastly, the survey presents common approaches for handling and applying each review characteristic. Moreover, the study sheds light on the challenges encountered in review mining and provides valuable insights to inspire future research endeavors.

During the process of collecting papers, we find several papers related to our work. Dąbrowski et al. (2022a) surveyed related work about app reviews from the perspective of software engineering and focused on high-level mining techniques, e.g., Manual Analysis, and Machine Learning. On the contrary, our SLR presents insights from the micro perspective, detailing common review characteristics as well as experience in handling and exploiting these characteristics. For example, the review rating can be handled to create features, such as computing the average rating. Its applications vary from serving as an input feature for classification algorithms to acting as a criterion for selecting app reviews with low star ratings. Besides, they overlooked some significant activities in the software ecosystem, e.g., app recommendation, and app rank promotion. Similarly, other literature reviews (Genc-Nayebi and Abran, 2017; Noei and Lyons, 2019; Tavakoli et al., 2018) focused on high-level mining techniques and surveyed a limited number of papers. Besides, Lin et al. (2022) introduced sentiment analysis techniques for software development and compared publicly available tools. Our work differs in focusing on app reviews rather than all types of artifacts. Moreover, our survey provides a more fine-grained analysis to allow researchers to systematically understand the characteristics of app reviews, as well as the handling and application practices accordingly. Besides, there is no universal methodology to effectively mine app reviews since the adopted methods vary in papers. Therefore, we provide a systematic review of papers on user reviews from the view of review characteristics.

At the end of the SLR, we outline existing practices in app review analysis and suggest future directions aligned with our research questions. Additionally, we offer reflections on using Large Language Models (LLMs) in this context, drawing insights from relevant recent studies. Besides, we discuss the availability and replicability of tools and datasets, and outline future directions for them. Furthermore, we summarize the impact of our SLR on SE community.

In summary, the key contributions of this paper are as follows:

- We present a systematic literature review of research on app review mining from the perspective of app review characteristics, providing insights for app review collection, review characteristic exploration, and review characteristic handling and application.
- We summarize the characteristics of app reviews by a key-point investigation.
- We propose future research directions for studies on mobile app review analysis in terms of review characteristics.

The rest of the paper is organized as follows. In Section 2, we present our research questions, systematic survey process, and an overview of our primary studies. The following three sections answer our research questions. Section 3 introduces the collection of app reviews. Section 4 presents our key-point investigation for summarizing common review characteristics. Section 5 exhibits the handling and applications of common review characteristics. Section 6 exhibits publicly available tools. Section 7 provides future research directions. Besides, Section 8 discusses related work. Finally, Section 9 gives a conclusion of our work. The supplementary data[1] is available.

[1] Supplementary data: https://github.com/harrietwhh/App_Review_SLR

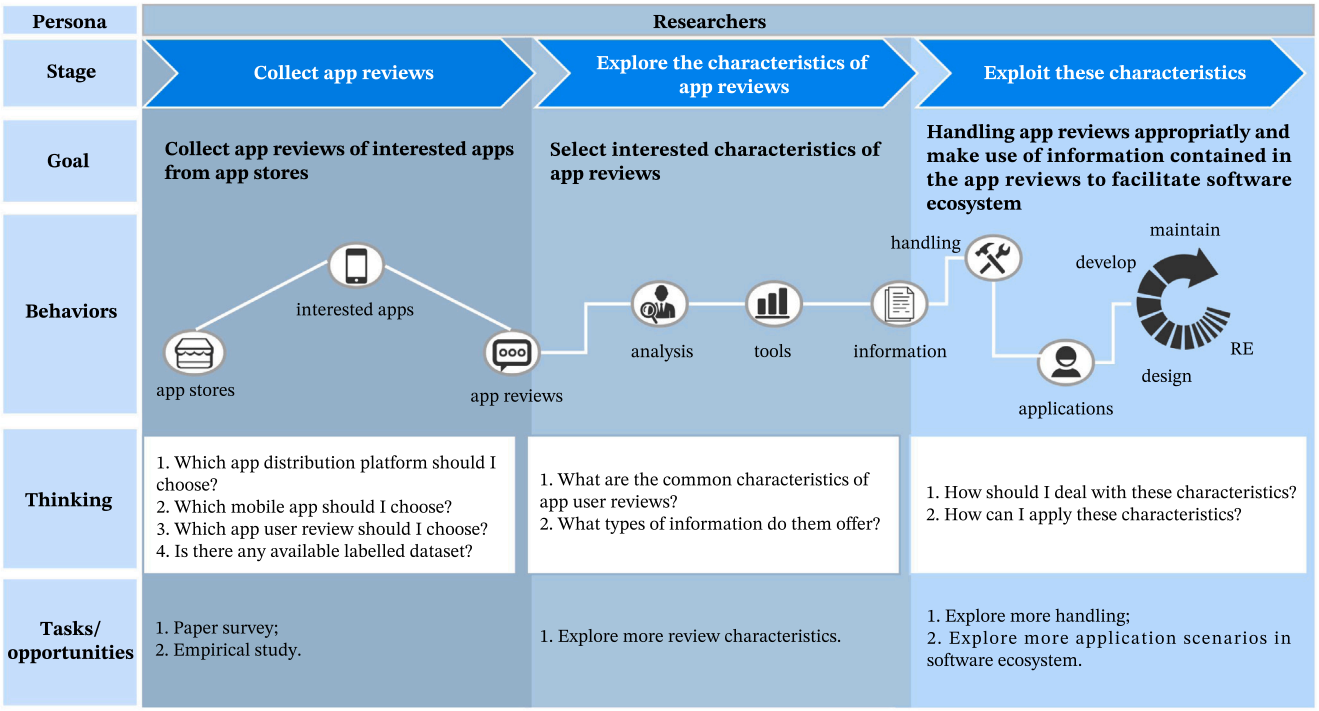

Fig. 2. Journey map of mining app reviews.

2. An overview of our systematic survey

2.1. Research questions

We adopt the journey map (Clarke, 2014; Lemon and Verhoef, 2016) to present the flowchart of the stages researchers go through during their interactions with the app reviews. As shown in Fig. 2, there are three stages when researchers interact with the app reviews, i.e., collecting app reviews (Martin et al., 2015), exploring the characteristics of app reviews (Licorish et al., 2017; Srisopha et al., 2020a; Gao et al., 2022a), and exploiting these characteristics for review mining (Licorish et al., 2017; Srisopha et al., 2020a; Gao et al., 2022a). Besides, following the practice of classic journey map (Clarke, 2014; Lemon and Verhoef, 2016), we present the key components in our journey map, including *touchpoints* (i.e., behaviors), *thinking* (i.e., research questions), and *opportunities*. We designed the following research questions (RQs) according to the three stages.

- **RQ 1: How are reviews collected in previous studies?**

App review mining heavily relies on the collection of app reviews. In particular, the method and credibility of app review mining are significantly impacted by the app distribution platform chosen (Martin et al., 2015), the specific app being analyzed (Tushev et al., 2022), and the number of reviews gathered (Martin et al., 2015). Therefore, we explore the app review collection process from the perspective of experimental subjects. We present our findings for this RQ in Section 3. At this initial phase of app review analysis, more exploratory analyses (e.g., paper surveys, and empirical studies) (Zhang et al., 2018) are needed to uncover the potentially different results when collecting various sets of app reviews, e.g., outdated reviews and recent reviews.

- **RQ 2: What are the common characteristics of app reviews?**

For this RQ, we aim to investigate the common app review characteristics. As Petz et al. (2013) suggest, there is no "one-size-fits-all" approach for all types of text due to the differences between their unique characteristics. Fundamentally, researchers need to have a good knowledge of these characteristics of app reviews. To achieve this goal, we adopt key-point investigation and survey the 20 most influential papers identified by Dąbrowski et al. (2022a) to derive a list of common characteristics of app reviews. We present our analysis process and findings for this RQ in Section 4. In this phase, additional review characteristics can be explored, e.g., the certain groups of people who provide app reviews (Van Oordt and Guzman, 2021). In this context, app reviews represent a form of biased user feedback, emphasizing the need for careful consideration of potential biases in the results.

- **RQ 3: How do previous studies handle the characteristics of app reviews and exploit them in software ecosystem?**

Merely being aware of the prevailing characteristics of app reviews is insufficient. Therefore, the literature review goes beyond that by elucidating the existing handling and application of each review characteristic, providing valuable practical insights to researchers involved in app review mining. For example, *Review Rating*, a fundamental review characteristic, can be handled to create new features, such as computing the average rating. Its applications vary from serving as an input feature for classification algorithms to acting as a criterion for selecting app reviews with low star ratings. In this example, "computing the average rating" is one way for handling *Review Rating*, "serving as an input feature for classification algorithms" and "acting as a criterion for selecting app reviews with low star ratings" are two applications for *Review Rating*. Armed with this knowledge, researchers can make informed decisions and choose appropriate analysis tools while conducting review analysis. We present our findings for this RQ in Section 5. In this phase, more handling and applications of review characteristics can be explored, e.g., combining with multimodal data in app distribution platforms.

- **RQ 4: How available and what are the limitations of tools proposed in previous studies?**

Table 1
List of activities and practices in software ecosystem.

Activities	Practices
Consumer request	Requirements sharing
Developer delivery	Development process
Market maintenance	App marketing, business model innovation
Quality maintenance	Application quality, app testing, security
Applications, components, and processes certification	Development monitoring, app approval, app curation, licensing, platform evolution
Technical training	Development support

In facilitating further research, we manually inspect and summarize open-source tools from 167 papers, ensuring link accessibility. This compilation aims to guide future researchers in selecting and replicating open-source tools by providing detailed summaries of their specific components. Additionally, we discuss their limitations. In the end, we discuss open-source tools involving developers in the evaluation process and present an industrial view of user feedback and automatic tools. We present our findings for this RQ in Section 6.

2.2. Methodology

We follow the well-established guidelines (Wohlin, 2014; Keele et al., 2007) to conduct Systematic Literature Review (SLR). The following outlines the SLR methodology employed in our study:

- Define the research scope. This step helps to clarify our research goal.
- Establish the list of keywords for searching.
- Conduct the search process. This step aims to search for relevant papers in conferences and journals by using commonly-used publication repositories.
- Apply exclusion criteria. This step helps to filter out papers out of our research scope, e.g., non-English papers.
- Conduct the *back snowballing* process (Sayagh et al., 2018) on the references of these remaining papers to alleviate the omission of related papers.

2.2.1. Search scope and inclusion criteria

We define our research scope: The papers should focus on both user reviews and software ecosystem. We aim to explore papers focusing on both user reviews and activities in software ecosystem. However, to the best of our knowledge, there are no available lists of software ecosystem activities. Therefore, we begin by manually identifying activities using the Brechó-VCM approach (dos Santos and Werner, 2010), which can model software ecosystem. Next, we map practices in Jansen (2020) into the previously identified activities. Since Jansen (2020) lists the practices involved in software ecosystem governance, we use this way to link them together. We present our list of activities and practices in software ecosystem in Table 1.

We apply the following inclusion criteria:

1. The paper should be related to mobile app reviews, concerning the use of textual user feedback gathered from one or more app distribution platforms.
2. The paper should aim at helping app researchers in one or more practices belonging to the software ecosystem (dos Santos and Werner, 2010; Jansen, 2020) utilizing the information extracted from user feedback.
3. The paper should be published in peer-reviewed journals or conference proceedings and the publication venue of the paper can be found on the newest list of CORE.[2] Therefore, for conference papers, we refer to the list of CORE 2021, and for journal papers, we refer to the list of CORE 2020.
4. The paper should be a peer-reviewed research paper since we want to extract insight from high-quality papers. This rule is achieved by examining paper metadata in DBLP.[3]

We manually check whether each candidate paper satisfies the above-mentioned inclusion criteria.

2.2.2. Search strategy

Search keywords. We establish the list of search keywords to identify potentially relevant papers within our search scope. The construction of search keywords is an iterative process. Based on the first rule of our search scope, research on app reviews, we list some search keywords and identify their synonyms, i.e., *app*, *application*, *user*, *review*, and *feedback*. During the search process, we find that users often appear together with the combination of "*app AND review*" or "*app AND feedback*", so we keep adjusting our search keywords. Since *app* is the abbreviation of *application*, searching for the keyword *app* can find all papers that contain the keyword *application*. After adjusting our search keywords for the first rule, we finally set up two combinations of keywords:

(a) app AND review

(b) app AND feedback

As for the second rule of our research scope, i.e., software ecosystem-related research, it is a very general topic and we hope to collect as many related papers as possible. Hence, instead of giving specific keywords to limit our search results, we choose to manually examine each paper.

Search Repositories. We conduct a search for papers within the following repositories: IEEE Xplore[4] and ACM Digital Library[5] Google Scholar is not considered as a target repository due to its inability to limit searches to abstracts, which can result in overwhelming search results. For instance, a simple search for "app AND review" generates around 4,700,000 results on Google Scholar.

Search Process. We conduct a systematic search for the keywords in each repository. However, since the keywords are very general, there are as many as 5,753,529 results in IEEE Xplore and 407,549 results in ACM Digital Library, which is infeasible for manual validation. Therefore, we decide to narrow our search to abstracts.

For each search result, we carefully examine the metadata (i.e., venue) and abstract of the papers. We then remove papers that are irrelevant based on the scope specified in Section 2.2.1. After that, the remaining papers are fully read and a judgment is made on whether the paper focuses on both user feedback and the software ecosystem.

2.2.3. Exclusion criteria

We define a set of exclusion criteria as follows to better meet the search scope.

(1) Non-English papers are filtered out.
(2) For papers that are duplicates or extensions stemming from the same study, we only select the most recent version.

[2] CORE Rankings Portal: https://www.core.edu.au/conference-portal

[3] dblp computer science bibliography: https://dblp.org/

[4] IEEE Xplore Digital Library: https://ieeexplore.ieee.org/Xplore/

[5] ACM Digital Library: https://dl.acm.org/

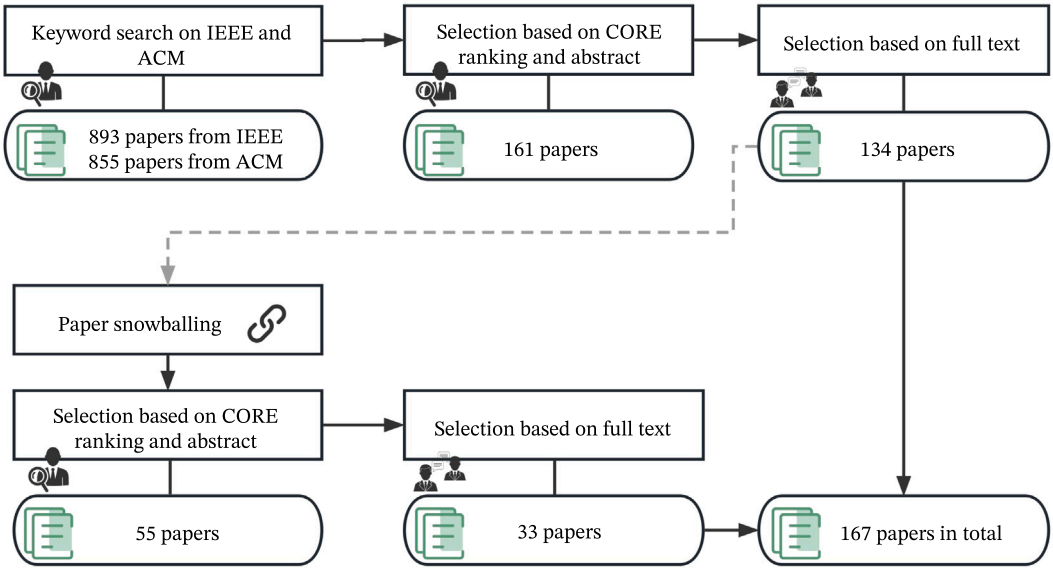

Fig. 3. Systematic review process.

Table 2
Number of papers found in IEEE and ACM via the keyword search.

Keywords	IEEE	ACM
app AND review	573	320
app AND feedback	491	364

(3) Survey, interview, literature review, position paper, and commentary are excluded since we want to explore practices on collecting app reviews, and handling and application of app review characteristics in previous studies.

2.2.4. Back snowballing

In case of omission, we conduct the backward snowballing (Sayagh et al., 2018) on the retrieved papers. Specifically, we manually inspect the references of each paper using Google Scholar[6] and Research Rabbit.[7] This approach aims to capture potentially relevant papers that are not in our search repositories or papers that cannot be identified by searching our keywords.

2.3. Overview of primary studies

In this subsection, Fig. 3 presents the corresponding searching and selection results, along with the systematic review process. To be specific, the first author conducted keyword searches in abstracts on IEEE Xplore and ACM Digital Library, yielding 893 and 855 papers, respectively. Table 2 lists the exact numbers of papers collected from the above-mentioned repositories utilizing different combinations of keywords. Initially, the first author manually checked each paper's venue on the CORE ranking list and examined the abstract for alignment with the research scope. If uncertain, the paper was temporarily kept. Besides, the first author excluded certain types of papers based on our exclusion criteria. This process resulted in 161 papers. Subsequently, the first and third authors collectively evaluated the 161 papers against inclusion criteria and exclusion criteria based on full text of each paper. Initial independent judgments showed an agreement rate of 0.76. Afterwards, the two authors discussed and reached an agreement, resulting in 134 papers in the first stage. Based on these remaining 134 papers, the first author implemented the *back snowballing* process in case of omission. In the snowballing stage, the first author analyzed the references of these 134 papers to identify additional relevant papers. The same evaluation process was then applied, with the consensus rate in the second round reaching 0.82. After discussion, a final consensus was reached, resulting in 33 more papers in our repository. Of the 33

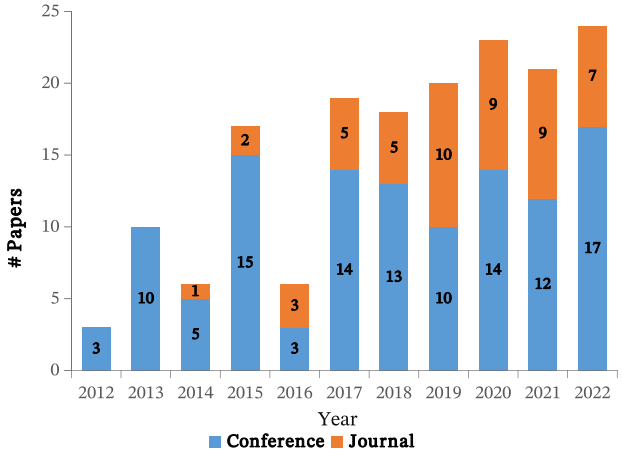

Fig. 4. Trend of the repository of literature.

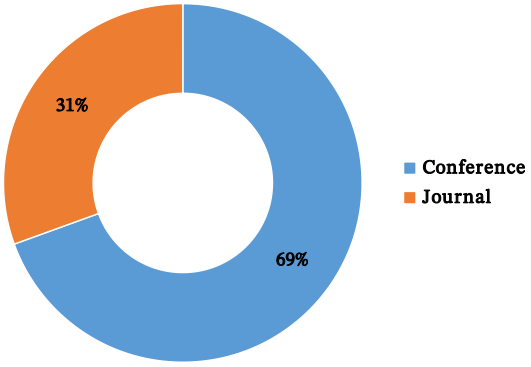

Fig. 5. Distribution of the repository of literature.

[6] Google Scholar: https://scholar.google.com
[7] Research Rabbit: https://researchrabbitapp.com/

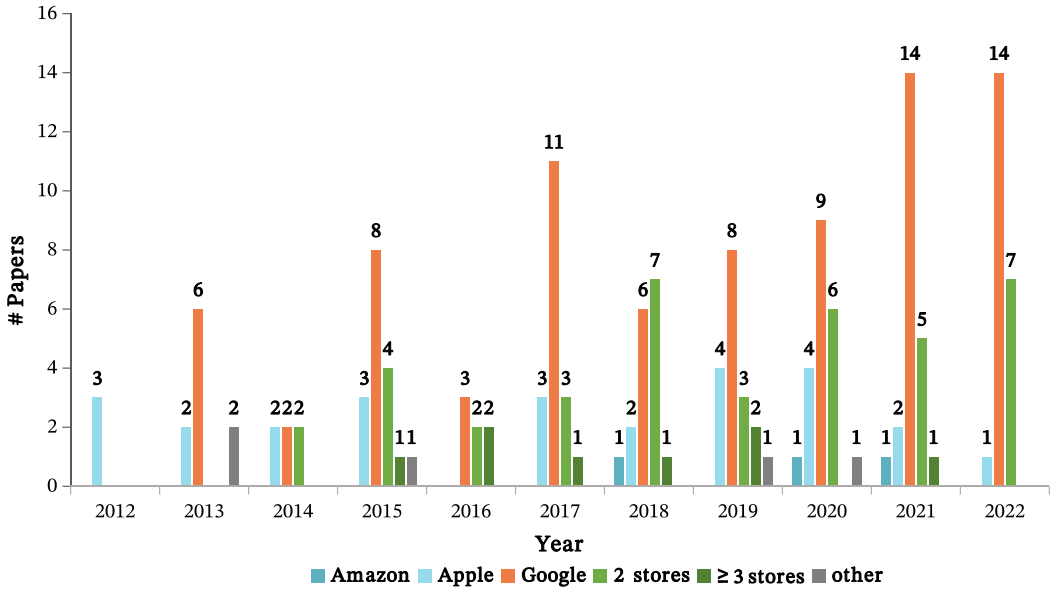

Fig. 6. Trend of app distribution platforms selected as study subjects.

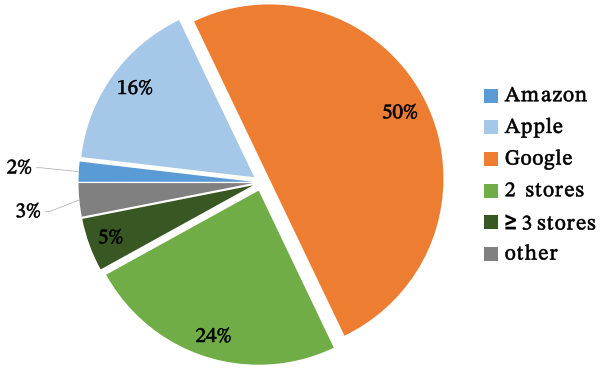

Fig. 7. Distribution of app distribution platforms selected as study subjects.

papers that were missed, 28 were missed because their venues were not indexed by IEEE Xplore or ACM Digital Library, and the remaining five papers did not contain the keywords we defined in their abstracts. Finally, we collected 167 papers in total, on which we then conducted a systematic review.

Fig. 4 shows the distribution of the collected papers through the published year (2012–2022). Among all the papers we have collected, the earliest one was published in 2012, which is consistent with the finding in an existing work (Martin et al., 2017). Furthermore, Martin et al. (2017) attributed this finding to the tenure of the app stores in their work. As shown in Fig. 5, conference papers dominate this field, making up for 69% of the studies. In particular, there are no relevant journal papers in the first two years. Overall, there is an upward trend in the number of surveyed papers. We present the fields that our surveyed papers concerned in Table 3. Most papers (116 out of 167 papers) belong to the field of Computer Software and Software Engineering. Human-Centered Computing ranks second in terms of the paper amount while the earliest two papers (Vasa et al., 2012; Hoon et al., 2012) in our repository belong to this field.

Besides, the three fields, including Data Management and Data Science, Distributed Computing, and Cybersecurity and Privacy, also contributed a lot in terms of both the quantity and influence of research papers. For instance, the paper of Fu et al. (2013) published in 2013 is noted as one of the ten most cited papers in the literature review (Dąbrowski et al., 2022a). We also present the top ten venues in terms of paper amount belonging to our repository in Table 4. All ten venues belong to the field of Software Engineering except Australian Computer Human Interaction Conference. Finally, the number of papers per research question is shown in Table 5. Notably, not all surveyed papers explicitly provide data information for each of our research questions. Therefore, this table presents the number of papers that do include corresponding data. Among the left-out papers, two papers Morales-Ramirez et al. (2017), Dąbrowski et al. (2022b) solely present their proposals without conducting experiments, leading to an absence of data. For the other papers, we cannot find corresponding data information in their paper text.

3. Collecting app reviews

In this section, we answer **RQ 1: How are reviews collected in previous studies?** App review mining heavily relies on the collection of app reviews. In particular, the method and credibility of app review mining are significantly impacted by the app distribution platform chosen (Martin et al., 2015), the specific app being analyzed (Tushev et al., 2022), and the number of reviews gathered (Martin et al., 2015). We explore the app review collection process from the perspective of experimental subjects. Consequently, our answer for RQ 1 follows these three stages: (i) app distribution platform selection, (ii) mobile app selection, and (iii) app review collection.

3.1. App distribution platform selection

We present the selection of app distribution platforms of previous work. As shown in Table 5, 162 out of 167 papers explicitly provide information about the app distribution platforms they focused on. The trend and distribution of app distribution platforms being studied in our remaining sample set are presented in Figs. 6 and 7. While comparing the frequency of the various app distribution platforms used as study subjects by previous studies, we highlight benefits/reasons and limitations for choosing specific app distribution platforms as study subjects.

3.1.1. Overall distribution

As shown in Fig. 6, the papers studying app reviews were first published in 2012, and they all studied the Apple App Store. After that, the number of papers studying the Apple App Store generally showed a slow downward trend. In the following year (2013), papers on Google

Table 3
Distribution of fields of the surveyed papers.

Field	Venue	Abbreviation	Type	CORE rank	# Papers
Software engineering	International Conference on Software Engineering	ICSE	Conference	A*	13
	IEEE International Requirements Engineering Conference	RE	Conference	A	12
	European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering	FSE	Conference	A*	9
	International Conference on Automated Software Engineering	ASE	Conference	A*	6
	International Conference on Evaluation and Assessment in Software Engineering	EASE	Conference	A	6
	International Conference on Software Analysis, Evolution and Reengineering	SANER	Conference	A	5
	International Conference on Software Maintenance and Evolution	ICSME	Conference	A	4
	Asia-Pacific Software Engineering Conference	APSEC	Conference	C	3
	International Symposium on Empirical Software Engineering and Measurement	ESEM	Conference	A	3
	International Working Conference on Mining Software Repositories	MSR	Conference	A	3
	International Workshop on Requirements Engineering: Foundation for Software Quality	REFSQ	Conference	B	3
	International Conference on Software and Data Technologies	ICSOFT	Conference	C	2
	International Symposium on Software Reliability Engineering	ISSRE	Conference	A	2
	International Conference on Software Engineering and Knowledge Engineering	SEKE	Conference	C	2
	Australian Software Engineering Conference	ASWEC	Conference	Australasian B	1
	International Conference on Program Comprehension	ICPC	Conference	A	1
	Software Quality, Reliability, and Security	QRS	Conference	C	1
	Euromicro Conference on Software Engineering and Advanced Applications	SEAA	Conference	B	1
	Conference on Software Engineering Research, Management and Applications	SERA	Conference	C	1
Human-centered computing	Australian Computer Human Interaction Conference	OZCHI	Conference	Australasian B	5
	International Conference on Human Factors in Computing Systems	CHI	Conference	A*	2
	International Conference on Intelligent User Interfaces	IUI	Conference	A	2
	International SIGACCESS Conference on Computers and Accessibility	ASSETS	Conference	A	1
	International BCS Human Computer Interaction Conference	HCI	Conference	National	1
	Mobile and Ubiquitous Multimedia	MUM	Conference	B	1
	International Conference on User Modelling, Adaptation, and Personalization	UMAP	Conference	B	1
	International Working Conference on Software Visualization	VISSOFT	Conference	B	1
Data management and data science	International Conference on Big Data	Big Data	Conference	B	2
	International Conference on Research and Development in Information Retrieval	SIGIR	Conference	A*	2
	Asia Information Retrieval Symposium	AIRS	Conference	C	1
	International Conference on Advanced Information Systems Engineering	CAiSE	Conference	A	1
	International Conference on Data Mining	ICDM	Conference	A*	1
	International Conference on Knowledge Discovery and Data Mining	KDD	Conference	A*	1
	Applications of Natural Language to Data Bases	NLDB	Conference	C	1
	Pacific-Asia Conference on Knowledge Discovery and Data Mining	PAKDD	Conference	B	1
	SIAM International Conference on Data Mining	SDM	Conference	A	1
	International Conference on Web Search and Data Mining	WSDM	Conference	A	1
Distributed computing and systems software	Consumer Communications and Networking Conference	CCNC	Conference	B	1
	Conference on Cognitive and Computational Aspects of Situation Management	CogSIMA	Conference	National:USA	1
	International Conference on Mobile and Ubiquitous Systems: Networks and Services	Mobiquitous	Conference	C	1
	International Symposium on Network Computing and Applications	NCA	Conference	B	1
	International Conference on Services Computing	SCC	Conference	B	1

(*continued on next page*)

Table 3 (*continued*).

Field	Venue	Abbreviation	Type	CORE rank	# Papers
Cybersecurity and privacy	Symposium on Security and Privacy	SP	Conference	A*	2
	Conference on Computer and Communications Security	CCS	Conference	A*	1
Applied computing	International Computer Software and Applications Conference	COMPSAC	Conference	B	1
	Symposium on Applied Computing	SAC	Conference	Multiconference	1
Distributed Computing	Conference on Hypertext and Social Media	Hypertext	Conference	A	1
Information Systems	Hawaii International Conference on System Sciences	HICSS	Conference	A	1
Computer Software	Empirical Software Engineering	EMSE	Journal	A	12
	IEEE Transactions on Software Engineering	TSE	Journal	A*	9
	IEEE Software		Journal	B	5
	Information and Software Technology	IST	Journal	A	3
	Requirements Engineering		Journal	B	2
	ACM Transactions on Software Engineering and Methodology	TOSEM	Journal	A*	2
	Computers and Security		Journal	B	1
	IEEE Transactions on Dependable and Secure Computing		Journal	A	1
	Journal of Software: Evolution and Process		Journal	B	1
	IEEE Transactions on Reliability	TR	Journal	A	1
Distributed Computing	IEEE Transactions on Mobile Computing		Journal	A*	4
	Future Generation Computer Systems		Journal	A	1
Data Format	IEEE Transactions on Knowledge and Data Engineering		Journal	A*	3
Information Systems	ACM Transactions on Information Systems		Journal	A	1
	Health Informatics Journal		Journal	C	1
	International Journal of Information Security and Privacy		Journal	C	1
Artificial Intelligence and Image Processing	IEEE Transactions on Evolutionary Computation		Journal	A*	1
Information and Computing Sciences	IEEE Transactions on Cybernetics		Journal	B	1
Library and Information Studies	American Medical Informatics Association		Journal	A	1
Total					167

Table 4
Top ten venues in terms of the amount of the surveyed papers.

Venues	# Papers
International Conference on Software Engineering	13
IEEE International Requirements Engineering Conference	12
Empirical Software Engineering	11
European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering	9
IEEE Transactions on Software Engineering	9
IEEE/ACM International Conference on Automated Software Engineering	6
International Conference on Evaluation and Assessment in Software Engineering	6
IEEE Software	5
Australian Computer Human Interaction Conference	5
IEEE International Conference on Software Analysis, Evolution and Reengineering	5
Total	81

Table 5
Number of papers per research question.

Research questions	# Papers
RQ 1.1: Which app distribution platforms to choose from?	162
RQ 1.2: Which apps to choose from?	158
RQ 1.3: Which app reviews to choose from?	158
RQ 2: What are the common characteristics of app reviews?	20
RQ 3: How do previous studies deal with the characteristics of app reviews and exploit them in software ecosystem	167
RQ 4: How available and what are the limitations of tools proposed in previous studies?	36

Play were published, and the number fluctuated since then, with a sudden increase in 2021, reaching 11 papers. The papers that study multiple app stores have been published since 2014. While 29.01% of studies employ multiple app distribution platforms to allow the generalization or comparison of results across multiple app distribution platforms, 67.90% papers focus on one target app market.

3.1.2. Google play and apple app store

Apps commonly run on specific devices and are developed for particular operating systems. With Android and Apple's iOS prevailing as the leading mobile operating systems, it is unsurprising that the largest global platforms for app distribution are Google Play and the Apple App Store.[8] As expected, most of the papers (147 out of 162) we have surveyed are related to Google Play or Apple App Store. Specifically, out of the 162 papers examined, those pertaining to Google Play accounted for 50.00% of the total, while papers related to Apple

[8] Number of apps available in leading app stores Q2 2022: https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/

App Store constituted 16.04%. Additionally, 22.60% of the papers covered content from both Google Play and Apple App Store.

Google Play is the app store with the biggest number of available apps. Among 162 papers providing information about app distribution platforms, 81 papers study Google Play alone while 95.74% of papers that study two or more app stores use Google Play as one of the target app stores. Furthermore, app reviews in Google Play are easier to access. On the one hand, Google Play officially provides the Google Play Developer API[9] which enforces a default restriction of 200,000 requests per day and 60 requests per hour for retrieving reviews. On the other hand, there are third-party Google Play Store app scraping solutions available, such as open source crawlers (e.g., Google Play Scraper,[10] Google Play Crawler,[11] and Play Drone[12]) and app monitoring platforms (e.g., Appfigures,[13] Kuchuan,[14] and QiMai[15]).

Concerning the application of app reviews, there are more related papers focusing on apps distributed on Google Play. This trend can be attributed to the abundance of open-source data (e.g., bug reports, commits in Github) and the comparatively more intricate software development process for Android apps when compared to apps for other operating systems, e.g., Apple's iOS. As shown in Fig. 6, the papers on mining app reviews boomed in 2015, and meanwhile, the number of papers on Google Play also increased sharply from two in 2014 to seven in 2015.

3.1.3. Limitations of scraping data from app distribution platforms

Scraping reviews from app distribution platforms is quite convenient via various official APIs or third-party APIs, and most of the previous studies (93.18%) retrieved app reviews in this way. However, there are some limitations when scraping data from app stores.

- **The application distribution platforms limit the number of reviews crawled.** On the one hand, the review collection service[16] officially provided by Google Play only allows reviews of last week to be crawled for each app (Hu et al., 2021). On the other hand, prior studies (Phong et al., 2015; Maalej and Nabil, 2015; McIlroy et al., 2017; Hassan et al., 2018, 2020; Chen et al., 2021) all found that Google Play has limitations in providing all reviews, restricting the availability to only the latest 500 reviews per app in a single connection. Among them, prior studies Phong et al. (2015), Hassan et al. (2018, 2020), Chen et al. (2021) used the same open-source crawler, Google Play Crawler.[17] Maalej and Nabil (2015) used another defunct open-source crawler. McIlroy et al. (2017) designed the scraper themselves. However, Khalid et al. (2014) scraped a maximum of 2,400 reviews for each app, which is consistent with the observations of Martin et al. (2017). It is explained that for each star rating, Google Play limits the total number of reviews a user can view to a maximum of 480, and there are 5 levels of ratings. To figure out the cause of the difference (i.e., 500 v.s. 2,400 reviews), we compare the time when they scraped data, which is consistent. Therefore, it is not likely that Google Play's mechanism has changed, and we conjecture that the difference is caused by the different crawling methods they used. Apart from the restrictive Google Play, Windows Phone Store only provides a maximum of 36 reviews per app (Martin et al., 2017).
- **The application distribution platforms only allow downloading the reviews of the latest version for each app.** Pagano and Maalej (2013) found that in the Apple App Store, reviews are reset with every release of the app. Hence, researchers only have access to app reviews belonging to the latest version. Similarly, Ciurumelea et al. (2017) found that Google Play only allows the selection of reviews for either the latest app version or all versions.
- **The application distribution platforms only allow crawling the latest review of each user for each app.** A user may review an app more than once, indicating high user engagement. However, Srisopha et al. (2021) found that Google Play does not support collecting change history for user reviews, and such data is only available to corresponding app developers.

3.1.4. Impact and solutions for the challenges

Due to the aforementioned limitations, if researchers collect app reviews by crawling the app distribution platforms and the apps only once, it may result in biased samples compared to the overall reviews provided by users. Martin et al. (2017) pointed out that app metrics such as star rating exhibit significant differences between the complete review data and the partially complete review data. Moreover, the app prevalence and request prevalence also present different trends.

To minimize the sampling error, researchers need to collect as many reviews as possible. There exist two solutions. The first is to resort to third-party app monitoring platforms. Hu et al. (2021) collected reviews for 8,400 apps from eight top Chinese third-party Android app markets via Kuchuan,[18] which maintains the app metadata including user reviews. The second is to increase the number of times the crawler visits the studied apps (Phong et al., 2015; McIlroy et al., 2017; Chen et al., 2021). McIlroy et al. (2017) ran their crawler daily within the span of two months to collect reviews from 10,713 top free-to-download apps, and they found 20 apps (0.19%) actually received no less than 500 reviews per 24-hour period. To collect app reviews under different updates for 2,526 top free apps, Hassan et al. (2020) adjusted their crawler to visit the application store many times per day and continuously run the crawler for 12 months. They observed that in 99.84% of the crawling times (i.e., 759,413 crawling cases in total), the crawler could collect all crawlable store data for the studied apps without missing any data.

3.2. Mobile app selection

After thoroughly analyzing the 167 papers and their explicitly mentioned referenced data sets, the app selection criteria employed in 158 of these papers are clear.

3.2.1. Overall distribution.

To illustrate the distribution of the number of apps collected in prior studies, we adopt unequal grouping, dividing the app counts into seven distinct groups. The resulting distribution is presented in Table 6. According to our statistics, the median of the numbers of selected apps is 39.50, while the mean is 38353.37, presenting a right-skewed distribution. The skewness coefficient (7.08) also confirmed this finding. Besides, the following statistics show that the numbers of selected apps are polarized. The numbers of selected apps in 60.13% of the papers are less than 100, and 51.58% of these papers do not exceed ten. These papers aim at proposing automatic tools to facilitate software ecosystem via analyzing the app reviews. Hence, they tend to conduct an exhaustive evaluation of a few specific apps, which often involves other information about the app, e.g., app descriptions (Yin and Pfahl, 2018), bug reports (Haering et al., 2021; Zhang et al., 2021; Palomba et al., 2015), and source code (Wei et al., 2017; Palomba et al., 2017; Zhang et al., 2021; Zhou et al., 2021). At the same time,

[9] Google Play Developer API: https://developers.google.com/android-publisher
[10] Google-Play-Scraper: https://github.com/JoMingyu/google-play-scraper
[11] Google Play Crawler JAVA API: https://github.com/Akdeniz/google-play-crawler
[12] PlayDrone: https://github.com/nviennot/playdrone
[13] Appfigures: https://appfigures.com/
[14] Kuchuan: https://www.kuchuan.com/
[15] QiMai: https://www.qimai.cn/
[16] Google Play Developer API: https://developers.google.com/android-publisher
[17] Google Play Crawler JAVA API: https://github.com/Akdeniz/google-play-crawler
[18] Kuchuan: https://www.kuchuan.com/

Table 6
Number of apps collected in the surveyed papers.

# Apps	<10	10~60	60~100	100~1k	1k~10k	10k~100k	>100k
# Papers	49	33	13	21	20	17	5
Ratio	31.01%	20.89%	8.23%	13.29%	12.66%	10.76%	3.16%
Accumulated Ratio	31.01%	51.90%	60.13%	73.42%	86.08%	96.84%	100.00%

there are also 26.58% of the papers examining more than 1,000 apps. Such analysis of a wide range of apps typically occurs when performing exploratory analysis of app reviews to understand their characteristics and content (Pagano and Maalej, 2013; Maalej and Nabil, 2015; Fu et al., 2013; Ali et al., 2017), developing recommendation systems for apps (Liu et al., 2015; Park et al., 2015; Yao et al., 2017; Bao et al., 2022), and generating developer responses for specific user reviews (McIlroy et al., 2017; Hassan et al., 2018; Srisopha et al., 2021). Under the circumstances, massive amounts of information would be helpful.

3.2.2. Strategies of app selection.

Concretely, existing studies consider factors such as app popularity, app category, and research purpose during the process of mobile app selection.

- **App Selection According to App Popularity**: Researchers tend to choose popular apps since popular apps have more reviews, from which more information can be extracted. Researchers can either directly acquire popular apps (Vasa et al., 2012; Srisopha et al., 2019; Oehri and Guzman, 2020), or sample apps from popular categories (Iacob et al., 2013; Gu and Kim, 2015). However, some researchers believe that considering popularity introduces a bias, and random selection of apps can be a better statistical representative of all apps (Noei et al., 2021).
- **App Selection According to App Category**: Different types of apps have different reviews, both in quantity and content. Analyzing 1,126,453 reviews from 1,100 apps, Pagano and Maalej (2013) found that most reviews for free apps come from the category "Social networking" (7.73%), and the least come from the category "Catalogs" (1.42%). Besides, most reviews for paid apps belong to the category "Utilities" (9.89%), least reviews are written in the category "Medical" (0.91%) (Pagano and Maalej, 2013). In terms of review content, the topic distribution of app reviews for each app category is unique (Fu et al., 2013; Di Sorbo et al., 2021a). Therefore, there are two diametrically opposite practices. In general, the selection of apps would take distinct app categories into account (Fu et al., 2013; Khalid et al., 2015; Kurtanović and Maalej, 2017; Noei et al., 2021; Oehri and Guzman, 2020). There are also studies that choose specific app categories, such as category Productivity (Goul et al., 2012; Johann et al., 2017), Health & Fitness (Hoon et al., 2013), Game (Khalid et al., 2014; Li et al., 2015). Additionally, Game is a different app category. Even in the Apple App Store, the category Game is displayed separately from apps. Games bring most of the revenue to application distribution platforms. In 2020, gaming apps accounted for 83% of global Google Play app revenues.[19] Compared with other apps, the user complaints with a game can be attributed from more aspects (Fu et al., 2013). Therefore, Maalej and Nabil (2015) suggested that focusing on scenarios in games would be helpful in classifying users' intentions conveyed in app reviews. In addition, games received fewer security- and privacy-related reviews (Nguyen et al., 2019). Considering the exceptional nature of the category Game, Uddin et al. (2020a,b) excluded the Games category and selected 2000 apps from the other ten categories. On the contrary, there are studies (Li et al., 2015; Lin et al., 2019; Khalid et al., 2014) targeting game reviews.

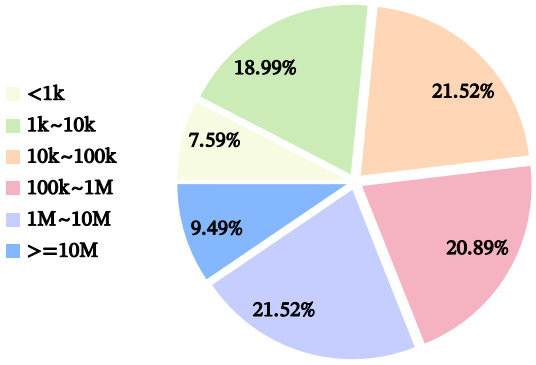

Fig. 8. App store distribution comparison for the papers in the category Review Mining and the other categories.

- **App Selection According to Research Purpose**: Apps can also be selected according to research objectives. When studying cross-platform apps, researchers can select apps receiving feedback on all app distribution platforms they studied (Oehri and Guzman, 2020). When studying developer responses, researchers can select apps with a large number of developer responses (Hassan et al., 2018). When locating source code based on the app reviews, researchers can select open-source apps (Wei et al., 2017; Ciurumelea et al., 2017; Palomba et al., 2017; Zhou et al., 2021; Zhang et al., 2021). Combining app reviews with other software artifacts is also a common practice. At this time, apps with detailed and adequate artifacts are required (Gao et al., 2018b; Yu et al., 2022; Gao et al., 2022b; Haering et al., 2021). Besides, FDroid[20] and App Annie[21] are two common tools adopted by previous studies (Eler et al., 2019; Chen et al., 2021; Grano et al., 2018; Yu et al., 2022; Ciurumelea et al., 2017; Palomba et al., 2017; Gao et al., 2021; Hassan et al., 2020, 2018; McIlroy et al., 2017; Gao et al., 2019) for app selection, which provide various lists of apps.

3.3. App review collection

This subsection focuses on a comprehensive study and analysis of the optimal number of reviews that researchers should collect. Additionally, we delve into the number of reviews present in the subsets selected by prior studies, accompanied by an examination of the rationales behind their sampling decisions.

3.3.1. Total dataset.

We conduct a comprehensive summary and analysis of the number of reviews gathered by each paper from the 158 studies that furnish pertinent data. Among these studies, a minimum of 50 reviews was collected while a maximum of around 580 million reviews was collected. In order to show the distribution of the number of reviews, we use unequal grouping to divide the number of reviews collected by each paper into six groups and present the distribution in Fig. 8. Papers with less than 1,000 reviews and papers with more than ten million reviews account for the lowest (7.59%) and second lowest percentages (9.49%). The proportions of the remaining four groups are relatively balanced, accounting for 18.99%, 21.52%, 20.89%, and 21.52% respectively.

[19] Mobileapp revenue worldwide 2017--2025, by segment: https://www.statista.com/forecasts/1262892/mobile-app-revenue-worldwide-by-segment

[20] F-Droid: https://f-droid.org

[21] data.ai, formerly App Annie: https://www.data.ai/

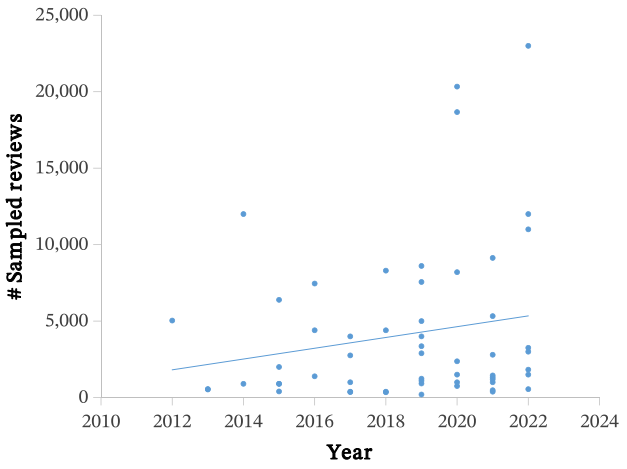

Fig. 9. Trend of the number of sampled reviews.

3.3.2. Subset.

A total of 60 papers further sample much smaller subsets from their entire datasets subsequently, and we summarize the number of reviews in the subset for each paper. We find that 83.33% of papers sample less than 10,000 reviews, and 55.00% of papers sample less than 3,000 reviews. The smallest subset consists of 200 reviews. The number of reviews in the largest subset is an outlier at 940,630 and it deviates from the mean by 6.38 standard deviations. This happens because this paper filters reviews with one- and two-star ratings from the population to study user complaints. Moreover, as shown in Fig. 9, the number of sampled reviews keeps increasing over the years.

- **Purposes of Sampling a Subset**: The general purpose of sampling a subset is to perform manual labeling for the evaluation of proposed automatic tools (Ali et al., 2017; Maalej and Nabil, 2015; Kurtanović and Maalej, 2017; Chen et al., 2014; Guzman and Maalej, 2014; Phong et al., 2015; Guzman et al., 2015a; Guo and Singh, 2020; Gu and Kim, 2015; AlOmar et al., 2021; Lu and Liang, 2017; de Araújo and Marcacini, 2021; Groen et al., 2017; Panichella et al., 2015; Wang et al., 2020; Nguyen et al., 2019; McIlroy et al., 2017; Hu et al., 2021). Another reason is that the authors exploit manual analysis, such as manually identifying user gender via usernames (Guzman and Paredes Rojas, 2019), manually classifying reviews according to app aspects (Pagano and Maalej, 2013; Di Sorbo et al., 2016; Srisopha et al., 2019), user complaints (Khalid et al., 2015) and UI-related issues types (Chen et al., 2021), manually eliciting drivers that make a developer respond to a review (Hassan et al., 2018), and manually analyzing to correct typos and informal vocabularies in app reviews (Noei et al., 2018). Most of them require manual coding by multiple domain experts and iterating the coding process multiple times to reach an agreement.
- **Strategies of Sampling a Subset**: Most papers adopt simple random sampling (Goul et al., 2012; Chen et al., 2014; Ali et al., 2017; Lu and Liang, 2017; McIlroy et al., 2017; Hassan et al., 2018; Srisopha et al., 2019; Noei et al., 2021; Nguyen et al., 2019; Hassan et al., 2020; de Araújo and Marcacini, 2021; Hu et al., 2021; Chen et al., 2021). Simple random sampling is the most basic sampling method. After determining the sampling probability, each sample point has an equal probability of being chosen through a randomization procedure. There are also a lot of papers that borrow ideas from stratified sampling and take into account some balancing criteria, e.g., app categories, ratings, and numbers of downloads (Oehri and Guzman, 2020; Pagano and Maalej, 2013; Maalej and Nabil, 2015; Kurtanović and Maalej, 2017; Guzman and Maalej, 2014; Groen et al., 2017; Panichella et al., 2015). Stratified sampling involves independently and randomly selecting samples from various subpopulations. This method partitions the population based on specific characteristics or rules, ensuring that the sample accurately represents the diversity present within the entire population. Moreover, there exist papers that borrow ideas from the key-point investigation. Key-point investigation is to select a small number of key sample points from the population and investigate them. Although the number of these key sample points is small, they are important in the population. Ha and Wagner (Ha and Wagner, 2013) selected the ten most helpful reviews for each of the 59 different applications. It was explained that these reviews were most likely to be seen by consumers since the reviews in Google Play are ordered by "Helpfulness" by default. App reviews can also be selected according to research objectives. Prior studies show that low-star reviews are more informative, and there are a certain number of papers that study negative reviews and choose reviews with low ratings (Khalid et al., 2015; Srisopha et al., 2021). There are papers selecting app reviews related to software requirements (Wang et al., 2018a; Panthum and Senivongse, 2021), and papers selecting reviews related to software maintenance (Haering et al., 2021; Nguyen et al., 2019).

  It is important to draw a statistically representative sample to ensure its ability to accurately represent the entire population. By achieving this, the results obtained from the sample can be utilized to make inferences about the population with a specific level of confidence (Chen et al., 2021). Prior studies (Pagano and Maalej, 2013; Noei et al., 2021; Guzman and Paredes Rojas, 2019; Srisopha et al., 2019; McIlroy et al., 2017; Hassan et al., 2018, 2020; AlOmar et al., 2021; Chen et al., 2021) investigated a statistically representative sample with a confidence level of 95%. Creative Research Systems' Sample Size Calculator[22] can help calculate the required sample size for a statistically representative sample.

3.4. Challenges

- **Issue: Data Availability.** Data availability provides the foundation for the study of artifacts (Abou Khalil and Zacchiroli, 2022). The raw app reviews can be crawled directly, and AndroZooOpen (Liu et al., 2020) provides continuously updated metadata retrieved from Google Play for 3,316 apps. However, our community lacks a well-curated, representative public dataset of labeled app reviews. Inspecting the 167 papers we have surveyed, there are 38 open-source datasets provided, and six of them are not accessible. On the one hand, the labels adopted in these datasets are not uniform since they are labeled independently for their own tasks. Moreover, the amount of annotation is not large enough. New powerful techniques like deep learning require a large amount of annotated data to help train better models. **Suggestion**: Large-scale datasets with fine-grained labels (so that researchers can merge the fine-grained labels as they want) need to be built. Besides, new techniques can be utilized for generating reviews for categories with fewer samples.
- **Issue: Data Timeliness.** Most of the data is too old and not continuously updated. For example, the dataset in Guzman and Maalej (2014) is the most frequently adopted dataset among 167 papers and this dataset is collected in 2014. **Suggestion**: Empirical studies are needed to see if there are any differences between the old and new app review datasets. Besides, keeping updating data is also practical.
- **Suggestion**: When collecting app reviews, some previous studies also collected various software artifacts for their studying purposes. We also find one data paper (Liu et al., 2020), which proposed AndroZooOpen, providing raw metadata retrieved from Google

[22] CreativeResearchSystems'SampleSizeCalculator:www.surveysystem.com/sscalc.html

Play, as well as repositories of open-source Android apps available on GitHub and F-Droid. This is a good trend since the joint software artifacts can provide more information for the software process (Geiger and Malavolta, 2018; Abou Khalil and Zacchiroli, 2022). However, in the 167 papers of our survey, no one uses AndroZooOpen, indicating that it still needs to spend efforts on merging app reviews and other software artifacts to improve the usability of the dataset.

- **Suggestion**: Researchers can select apps with different strategies. For example, to pursue app success, researchers analyze apps with high ratings (Hassan et al., 2020; Wu et al., 2021). Besides, researchers can conduct exploratory analysis to compare the differences between apps with high ratings and apps with low ratings in the same category. Moreover, the number of downloads could also be an important indicator of app success, and researchers can investigate apps with a high number of downloads.

Answer to RQ 1: Most papers (67.90%) focus on a single target app market, primarily the Apple App Store and Google Play. These platforms allow various approaches for collecting reviews, including official APIs, open-source crawlers, and app monitoring platforms. Regarding app selection, prior studies exhibit polarization, either studying a large number of apps or a few specific apps. Concretely, the selection process takes into account factors such as app popularity, app category, and research purpose. As for review collection, the total number of reviews gathered by researchers is evenly distributed. Furthermore, 60 papers opt to sample a smaller subset of reviews. Specifically, 83.33% of them sample less than 10,000 reviews, and 55.00% sample less than 3,000 reviews.

4. Exploring app review characteristics

In this section, we answer **RQ 2: What are the common characteristics of app reviews?** For those who explore app reviews for the first time, they are concerned about what kind of text this is. Therefore, we derive a list of common characteristics of app reviews through a key-point investigation. Specifically, we provide a detailed description of our analysis approach, the results of the key-point investigation, and comprehensive information for the discussed characteristics.

4.1. Analysis approach

To derive a list of common characteristics of app reviews, we adopt key-point investigation and survey the twenty most influential papers identified by Dąbrowski et al. (2022a). Key-point investigation is to select a small number of key sample points from the population and investigate them. Although the number of these key sample points is small, they are important in the population. To be specific, for each paper, we first identify and code the review characteristics explicitly mentioned in the paper text. Then, we take the union of the features mentioned in different papers and summarize them. As shown in Table 7, we present brief information about review characteristics in each paper.

4.2. Results of key-point investigation

As shown in Table 8, we summarize the extracted keywords into seven categories, i.e., review rating, review length, textual content, domain knowledge, dynamic nature, review sentiment, and large amount. Meanwhile, the ratio of each characteristic is presented in Fig. 10.

Review rating and *textual content* are two major components of user review, and they are mentioned in 90% and 80% of these 20 papers respectively. *Review ratings* are the overall perception given by users for apps. The keywords are mainly "star rating" (six out of 20 papers) and "rating" (11 out of 20 papers). Following the definition of Ghose and Ipeirotis (2010), we treat anything relevant to lexicon, grammar, semantics, and style as *textual content*, for example, "typos, acronyms, abbreviations, emoji icons" (Phong et al., 2015), "seldom obey grammar and punctuation rules" (Iacob and Harrison, 2013), using unconventional syntax or sarcasm (Iacob and Harrison, 2013), informal nature (McIlroy et al., 2016), and unstructured nature (Guzman et al., 2015b; Panichella et al., 2015; McIlroy et al., 2016). *Review length*, *dynamic nature*, *review sentiment*, and *large amount* are also commonly mentioned characteristics, and the mentioned ratios are 45%, 50%, 45%, and 45%, respectively. *Review length* measures the number of words/ characters (Guzman et al., 2015b) in a review. Apart from "review length" and "length", keywords related to "short" are also mentioned in these 20 papers, for example, "shorter in length" (Fu et al., 2013), "short" (Iacob and Harrison, 2013), "shorter than issue descriptions" (Palomba et al., 2015). We summarize the keywords related to "time" and "app version/release" as characteristic *dynamic nature* (Gao et al., 2018a; Hadi and Fard, 2020), for example, "timestamp" (Fu et al., 2013; Chen et al., 2014), "posting date" (Iacob and Harrison, 2013; Di Sorbo et al., 2016; McIlroy et al., 2016), "reviews are often specific to a particular version and vary over time" (Fu et al., 2013), app version (Iacob and Harrison, 2013; Di Sorbo et al., 2016). Keywords related to "sentiment" are directly and indirectly proposed, for example, "sentiment" (Carreño and Winbladh, 2013; Fu et al., 2013; Guzman and Maalej, 2014; Gu and Kim, 2015; Di Sorbo et al., 2016; Maalej et al., 2016), "reflecting positive and negative emotions" (Maalej and Nabil, 2015). Similarly, keywords such as "a large amount of received feedback" (Panichella et al., 2015), "a large number of reviews" (Iacob and Harrison, 2013; McIlroy et al., 2016), and "hundreds of reviews submitted per day for popular apps" (Maalej et al., 2016; Villarroel et al., 2016) are summarized as characteristic *large amount*. Lastly, we observe keywords "app category" (Vasa et al., 2012), "device" (Iacob and Harrison, 2013), "author name" (Martin et al., 2015), "user handle" (Di Sorbo et al., 2016), "vocabulary mismatch between user reviews and source code or issues reported in issue trackers" (Palomba et al., 2015). These words reflect information related to the app store and app development, so we summarize them as *domain knowledge*.

4.3. Discussed characteristics

After identifying these characteristics, we present comprehensive information for the discussed characteristics as follows. Please note that we leave out the *large amount* characteristic, which merely necessitates automated handling of app reviews but does not offer additional information for mining app reviews.

4.3.1. Review rating

When users review apps, they also provide star ratings to quantitatively measure their preferences for the apps.

- **Measurement of App Market Success**: Rating is often considered an indicator of app quality, used to measure the market success of an app (Linares-Vásquez et al., 2013; Guerrouj et al., 2015; Deka et al., 2017; Di Sorbo et al., 2021a). For instance, if errors occur, users are likely to quickly give poor ratings, resulting in a rapid decline in sales. Apart from code quality (Corral and Fronza, 2015), marketing strategy (Tian et al., 2015; Kübler et al., 2018) and UI design (Taba et al., 2014; Noei et al., 2017; Doosti et al., 2018) also have impacts on the overall rating of an app. However, using app rating to measure app market success is based on the premise that these applications are developed for a wide range of users.

Table 7
Review characteristics explicitly mentioned in the twenty most influential papers.

Papers	Review Characteristics
Vasa et al. (2012)	Star rating, review length, app category.
Carreño and Winbladh (2013)	Rating, sentiment.
Fu et al. (2013)	Rating, text comment, shorter in length, submitted from mobile devices on which typing is not so easy, timestamp, reviews are often specific to a particular version and vary over time, sentiment.
Iacob and Harrison (2013)	Posting date, rating, device, app version, style (short, unstructured and seldom obey grammar and punctuation rules), using unconventional syntax or sarcasm, a large number of reviews.
Pagano and Maalej (2013)	Length, ratings, helpfulness, feedback content (feedback type and pattern), feedback frequency.
Chen et al. (2014)	Text, rating, timestamp, the proportion of "informative" user reviews is relatively low, the volume of user reviews is simply too large.
Guzman and Maalej (2014)	Star rating, sentiment, short (less than 160 characters), the quality of the reviews varying widely.
Guzman et al. (2015b)	Title, comment, rating, number of words in the review, number of characters in the review, ratio of positive sentiment words, ratio of negative sentiment words, relatively low proportion of informative user reviews, unstructured nature of app reviews, free form, high amount of user reviews for popular apps.
Khalid et al. (2015)	Rating, comment.
Maalej and Nabil (2015)	Star rating, length, submission time, a bunch of useless, low quality reviews, reflecting positive and negative emotions.
Martin et al. (2015)	Rating, review body, author name.
Panichella et al. (2015)	Review comment, star rating, a large amount of received feedback, unstructured nature, varying quality.
Palomba et al. (2015)	Stars, notably shorter than issue descriptions, limited vocabulary, vocabulary mismatch between user reviews and source code or issues reported in issue trackers, free text, without predefined structure, describing informally bugs and desired features.
Gu and Kim (2015)	Rating, sentiment, the volume of user reviews is too large.
Phong et al. (2015)	Thousands of reviews each day, noisy (typos, acronyms, abbreviations, emoji icons), 60% of user reviews do not contain useful opinions, review topics affected by app releases.
Di Sorbo et al. (2016)	Title, review text, posting date, user handle, app version, star rating, review length, sentiment.
Maalej et al. (2016)	Star rating, length, submission time, free text, a bunch of useless, low-quality reviews, sentiment, hundreds of reviews submitted per day for popular apps.
Maalej et al. (2015)	Short, uninformative reviews for negative ratings.
McIlroy et al. (2016)	Title, date, rating, the unstructured and informal nature of reviews, large number of user reviews, free form.
Villarroel et al. (2016)	Rating, release, hundreds of reviews per day.

Table 8
Review characteristics and relevant keywords in the twenty most influential papers.

Review characteristics	Keywords
Review rating	Star rating (Vasa et al., 2012; Guzman and Maalej, 2014; Maalej and Nabil, 2015; Panichella et al., 2015; Di Sorbo et al., 2016; Maalej et al., 2016), rating (Carreño and Winbladh, 2013; Fu et al., 2013; Iacob and Harrison, 2013; Pagano and Maalej, 2013; Chen et al., 2014; Guzman et al., 2015b; Khalid et al., 2015; Martin et al., 2015; Gu and Kim, 2015; McIlroy et al., 2016; Villarroel et al., 2016), stars (Palomba et al., 2015).
Review length	Review length (Vasa et al., 2012; Di Sorbo et al., 2016), shorter in length (Fu et al., 2013), short (Iacob and Harrison, 2013), length (Pagano and Maalej, 2013; Guzman and Maalej, 2014; Maalej et al., 2016), number of words in the review (Guzman et al., 2015b), number of characters in the review (Guzman et al., 2015b), shorter than issue descriptions (Palomba et al., 2015), limited vocabulary (Palomba et al., 2015).
Textual content	Text comment (Fu et al., 2013), submitted from mobile devices on which typing is not so easy (Fu et al., 2013), style (Iacob and Harrison, 2013), unstructured (Iacob and Harrison, 2013), seldom obey grammar and punctuation rules (Iacob and Harrison, 2013), using unconventional syntax or sarcasm (Iacob and Harrison, 2013), helpfulness (Pagano and Maalej, 2013), feedback content (feedback type and pattern (Pagano and Maalej, 2013), relatively low proportion of "informative" user reviews (Chen et al., 2014; Guzman et al., 2015b), varying quality (Guzman and Maalej, 2014; Panichella et al., 2015), title (Guzman et al., 2015b; McIlroy et al., 2016), comment (Guzman et al., 2015b; Khalid et al., 2015), unstructured nature (Guzman et al., 2015b; Panichella et al., 2015; McIlroy et al., 2016), free form (Guzman et al., 2015b; McIlroy et al., 2016), a bunch of useless, low quality reviews (Maalej and Nabil, 2015; Maalej et al., 2016), review body (Martin et al., 2015), review comment (Panichella et al., 2015), free text (Palomba et al., 2015; Maalej et al., 2016), without predefined structure (Palomba et al., 2015), describing informally bugs and desired features (Palomba et al., 2015), noisy (typos, acronyms, abbreviations, emoji icons) (Phong et al., 2015), 60% of user reviews do not contain useful opinions (Phong et al., 2015), review text (Di Sorbo et al., 2016), short, uninformative reviews for negative ratings (Maalej et al., 2015), informal nature (McIlroy et al., 2016).
Domain knowledge	App category (Vasa et al., 2012), device (Iacob and Harrison, 2013), author name (Martin et al., 2015), vocabulary mismatch between user reviews and source code or issues reported in issue trackers (Palomba et al., 2015), user handle (Di Sorbo et al., 2016).
Dynamic nature	Timestamp (Fu et al., 2013; Chen et al., 2014), often specific to a particular version (Fu et al., 2013), varying over time (Fu et al., 2013), posting date (Iacob and Harrison, 2013; Di Sorbo et al., 2016; McIlroy et al., 2016), app version (Iacob and Harrison, 2013; Di Sorbo et al., 2016), feedback frequency (Pagano and Maalej, 2013), submission time (Maalej and Nabil, 2015; Maalej et al., 2016), review topics affected by app releases (Phong et al., 2015), release (Villarroel et al., 2016).
Review sentiment	Sentiment (Carreño and Winbladh, 2013; Fu et al., 2013; Guzman and Maalej, 2014; Gu and Kim, 2015; Di Sorbo et al., 2016; Maalej et al., 2016), ratio of positive sentiment words (Guzman et al., 2015b), ratio of negative sentiment words (Guzman et al., 2015b), reflecting positive and negative emotions (Maalej and Nabil, 2015), short, uninformative reviews for negative ratings (Maalej et al., 2015).
Large amount	A large number of reviews (Iacob and Harrison, 2013; McIlroy et al., 2016), large volume (Chen et al., 2014; Gu and Kim, 2015), high amount of user reviews for popular apps (Guzman et al., 2015b), a large amount of received feedback (Panichella et al., 2015), thousands of reviews each day (Phong et al., 2015), hundreds of reviews submitted per day for popular apps (Maalej et al., 2016; Villarroel et al., 2016).

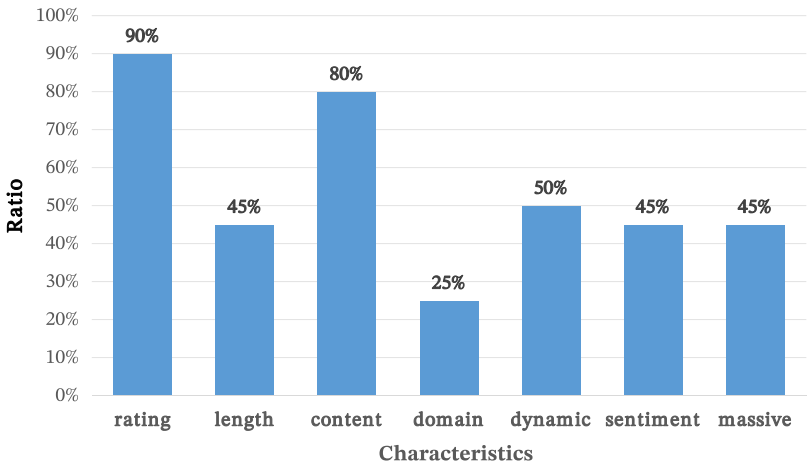

Fig. 10. The ratio of each characteristic mentioned in 20 most influential papers.

When interviewing five software companies, Pagano and Bruegge (2013) found that public ratings are considered less important by the product manager when the corresponding software is developed for a smaller group of professionals.

- **Correlation with Sentiment**: By presenting top words across all ratings and analyzing the distribution of a specific sentiment detection word set amongst the ratings, Hoon et al. (2012) drew an empirical inference that app rating is associated with the words expressing sentiment. This finding helps to determine whether the review text is consistent with the rating, thus filtering inconsistent reviews (Fu et al., 2013).
- **Correlation with User Intentions**: Feature requests are always associated with higher ratings while bug reports are always associated with lower ratings (Pagano and Maalej, 2013; Ha and Wagner, 2013; Di Sorbo et al., 2021a). Hence, the rating can be an important feature for classifying or prioritizing app reviews (Phong et al., 2015; Maalej and Nabil, 2015; Gao et al., 2018a).

4.3.2. Review length

In this part, we present the typical size of a review and the influencing factors of review length.

- **Typical Size of A Review**: Hoon and his colleagues (Hoon et al., 2012; Vasa et al., 2012) statistically analyzed 8.7 million user reviews from 17,330 top free and paid apps on the Apple App Store, finding that the median feedback length across all applications is 69 characters and the mean is 117, indicating that users tend to leave short feedback. Therefore, techniques that preprocess and analyze short text can be used to analyze app reviews. Pagano and Maalej (2013) attained similar statistics, observing a median of 61, a mean of 106.09, and 76.7% of reviews being within 140 characters. These subtle differences may be caused by different sample sizes. Although longer reviews are more likely to contain useful content because users spend more time providing them (Vasa et al., 2012), Pagano and Maalej (2013) found none of the extremely long reviews contain useful information.
- **Influencing Factors of Length**: App review length is found to be related to review rating, review sentiment, app category, and culture. Review with a poor rating is likely to be longer, which accords with the finding that users tend to write less the more they like an application and vice versa (Vasa et al., 2012; Pagano and Maalej, 2013). It is explained that users write longer reviews to request improvements. Hence, review length can be an important feature for classifying app reviews. Moreover, app category and culture (Guzman et al., 2018) also have impacts on review length. In particular, Vasa et al. (2012) observed that reviews for the category Game are significantly shorter. Guzman et al. (2018) found that users from a country with a higher level of Individualism and Indulgence tend to write longer app reviews while users from a country with a higher level of Power Distance tend to write shorter app reviews. Thus, the review length can be an important feature for classifying or prioritizing app reviews (Maalej and Nabil, 2015; Gao et al., 2018a).

4.3.3. Textual content

In this part, we present the characteristics of review content, concerning words, language style, and informativeness.

- **Limited Frequent Words**: Analyzing 8.7 million user reviews, Hoon et al. (2012) found that only a small set of words is used frequently in all reviews. In particular, only 5,200 words occur more than 1,000 times, and less than 3,600 words occur over 2,000 times. They suggested that this limited set of words can help us better train our tools. Moreover, previous studies also exploited processing techniques such as filtering out rare words and identifying collocations to extract information from app reviews.
- **Informal Language**: App reviews, as a type of text that users generate on social media, are informally written, thus prone to be less grammatically correct and contain typos, acronyms, abbreviations, slangs, emoji icons, etc. Hoon et al. (2012) identified 881,879 unique text expressions and found that 17% of them occur less than three times suggesting the existence of miss spelled words, user-constructed words (e.g., *intelliweather, loooved, g8*), sentences without punctuation and other expressions causing the low text quality. While a human reader may infer the meaning of these terms within the given context, computer-based analysis can encounter difficulties with such word types. Hence, common pre-processing techniques are needed as described in Section 5.4.1.
- **Polarities in Informativeness**: The amount of information contained in app reviews presents two extremes. On the one hand, previous studies (Vasa et al., 2012; Pagano and Maalej, 2013; Chen et al., 2014) report that about two-thirds of the app reviews are uninformative to help app researchers in terms of mobile app development and evolution, merely expressing sentiment. On the other hand, over 50% of the reviews contained multiple topics (Pagano and Maalej, 2013). A review often contains multiple sentences, and each sentence may express different intentions. Furthermore, a review sentence may even describe user experience on multiple app aspects. Fig. 11 presents a user review of TikTok in Google Play, reflecting three app aspects the user complains about, i.e., *favorite button*, *like button*, and *follow button*. Besides, low-star reviews are more informative since they inform developers about app aspects for improvement (Khalid et al., 2015; Srisopha et al., 2021). To

> *"There's been 3 glitches that I keep coming across. 1 the favorite button every time I tried to add something to the favorites it wouldn't. I tried everything uninstalling and reinstalling, restarting my phone over and over again, logging in and out, and more. Number 2 and 3 is the same issue but* with *the like button and follow button. It's so annoying and frustrating and makes the app near impossible to use."*

Fig. 11. A user review of TikTok in Google Play.

handle this characteristic, previous studies filter out uninformative reviews and choose suitable analytic levels, as presented in Section 5.4.1.

4.3.4. Domain knowledge

App reviews present the user experience with certain apps, concerning the functionality and non-functionality aspects of apps. Compared with professional technical artifacts (e.g., bug report, change log, and source code), app reviews are usually written by non-technical users without any domain knowledge of mobile app development (Ciurumelea et al., 2017). Besides, most users are not experienced in review writing. Analyzing 8.7 million iOS app reviews written by 5.5 million unique authors, Hoon et al. (2016) found that 71.5% of authors only write one review while 2.02% of authors write more than five reviews. In other words, 88.55% of the reviews are contributed by authors who write five or fewer reviews. As a result, app users may not be able to provide enough detail in precise technical language (Haering et al., 2021), thus providing no actionable information for app developers to maintain or improve apps. To mitigate this problem, app reviews can be jointly mined with other domain knowledge, e.g., device (Iacob and Harrison, 2013; Martin et al., 2015), app price (Martin et al., 2015), app category (Vasa et al., 2012), app description (Martin et al., 2015).

4.3.5. Dynamic nature

The dynamic nature of app reviews is reflected in both macro and micro perspectives. Macroscopically, app reviews are triggered by app updates and are written for specific app versions. Hence, different app versions result in reviews reflecting different issues and user experiences. Previous studies (Pagano and Maalej, 2013; Fu et al., 2013) observed that spikes in reviews are closely linked to new app updates. These spikes predominantly manifest as bursts of positive or negative comments. Concisely, the median time elapsed between submitting feedback and the latest app release is six days, and users tend to provide less feedback over time (Guzman et al., 2018). Among all the app reviews, 2.24% of them explicitly mention the app version, praising improvements brought by updates or complaining about the accompanying usability issues (Iacob et al., 2013). Besides, new users keep coming up to comment on apps. Microscopically, there are users updating their reviews for different app releases (Hoon et al., 2016) or after communicating with the app team (Hassan et al., 2018). This characteristic sheds light on integration with release notes and changelogs.

4.3.6. Review sentiment

The most common words in user reviews tend to convey emotion, with a disproportionately large number of words expressing negative sentiment (Hoon et al., 2012). However, Iacob et al. (2013) observed that users tend to leave positive feedback. In particular, among 3,279 reviews they manually coded, positive feedback accounted for 49.02% while negative feedback accounted for only 6.47%. One explanation is that a positive experience with an app motivates users to write reviews (Hoon et al., 2016).

- **Positive Sentiment**: Iacob et al. (2013) observed a correlation between review positivity and feature request, which implies that users would always look for a better app. This finding accords with the observation that feature requests are always associated with higher ratings (Pagano and Maalej, 2013; Ha and Wagner, 2013). Hence, sentiment can be an important feature for classifying app reviews.
- **Negative Sentiment**: Negative reviews always accompany reports of major bugs, while reports on minor bugs often being associated with positive feedback (Iacob et al., 2013). Hoon et al. (2012) analyzed top words across all ratings and found that words expressing sentiment are inconsistent with review ratings. Moreover, they observed that lower ratings (i.e., 1, 2, and 3) present a high concentration of user-perceived faults with the app. Observing top words such as *crashes* and *screen*, they suggested that phrase-based text extraction techniques could be used to elicit user pain points with the apps and prioritize different app issues to be addressed. Moreover, analyzing negative reviews via uni-gram is not adequate, because negation (e.g., do not, not worth) is commonly used to express dissatisfaction.

4.4. Trend of app review topics

In this part, we present the trend of app categories and app review topics, as well as the distribution of app review topics across different app categories. The data is obtained from all 167 papers.

4.4.1. App category trend

Analysis Process: To begin with, we provide an overview of the evolution of various app categories studied over time. Before analysis, we first determine a list of app categories. The segmentation of app categories by app distribution platforms is notably granular; for instance, both Google Play Store and Apple App Store provide over 25 app categories, many of which are similar. The presence of redundant categories further complicates analysis and comparison. Therefore, we adopt the list of app categories introduced by Huebner et al. (2020), where they grouped 34 Google Play Store app categories into eight clusters. Additionally, we map app categories from other app distribution platforms to this established list.

Findings: Fig. 12 illustrates the trend of each app category based on our repository. Over the period from 2012 to 2016, there was fluctuation in the frequency of studies conducted on each app category, with peaks occurring in 2013 and 2015. Starting in 2017, there was a stable increase in the study of all app categories, which aligns with the overall trend depicted in Fig. 6. Notably, the categories of *Productivity*, *Media Consumption*, *Utilitarian*, and *Social* garnered the most attention in research. *Media Consumption*, *Utilitarian*, *Social*, *Health, Nutrition, and Medical* categories demonstrated stable growth, with peaks occurring in 2020. Conversely, *Productivity* and *Other* experienced a decline in the number of studies in 2018 and 2019 compared to 2017, followed by a peak in 2019 and a subsequent gradual decline. The categories of *Game* and *Photography*, on the other hand, received comparatively less research attention, and the number of studies on these categories exhibited fluctuations from 2012 to 2022.

4.4.2. Review type trend

Analysis Process: Initially, we read all 167 papers to extract details regarding the taxonomy of review content and the associated categorization standards. For example, Lu and Liang (2017) categorized app reviews into Non-Functional Requirements (NFRs) such as reliability, usability, portability, and performance, Functional Requirements (FRs), and Others based on standard *User Requirements*. In this process, we identified a total of 68 papers with explicit categorizing standards, and 88 papers exhibited clear categorization of reviews, either through manually defined taxonomy or clearly defined review types. We analyzed and merged these standards, resulting in nine standards

Table 9
Examples for review categorization standards.

Categorization standards	Taxonomies
App Aspect	Advertising, User Interface, Pricing and Payments, Resource Usage, Device Compatibility, Connectivity, Privacy, Sign-up Experience, Tutorial, Audio, Video, Notification/Alerts, Translation and Internationalization, Location Services, Uninstall*, Update, Stability (Huebner et al., 2018).
Issues & Complaints	Functional Complaint, Crashing, User Interface, Feature Request, Additional Cost, Privacy and Ethical Issue, Network Problem, Compatibility Issue, Feature Removal, Response Time, Uninteresting Content, Update Issue, Resource Heavy, Unspecified (McIlroy et al., 2016).
Software Evolution	Praise, Feature Evaluation, Bug report, Feature request, Other (Shah et al., 2019).
Security & Privacy	Tracking & Spyware, Phishing, Unauthorized Charges, Unintended Data Disclosure, Targeted Ads, Spam, General (Hatamian et al., 2019).
Undesired Behaviors	Unfair Cancellation and Refund Policies, False Advertisements, Delusive Subscriptions, Cheating Systems, Inaccurate Information, Unfair Fees, No Service, Deletion of Reviews, Impersonation, Fraudulent-Looking Apps (Obie et al., 2022).
User Intention	Information Giving, Information Seeking, Feature Request, Problem Discovery, Other (Di Sorbo et al., 2016).
User Experience	Rating, User Experience, Requirements, Community (Pagano and Maalej, 2013).
User Requirements	NFRs (Reliability, Usability, Portability, and Performance), Functional Requirements (FRs), Others (Lu and Liang, 2017).
Other	Useless, Helpful (Gao et al., 2022a).

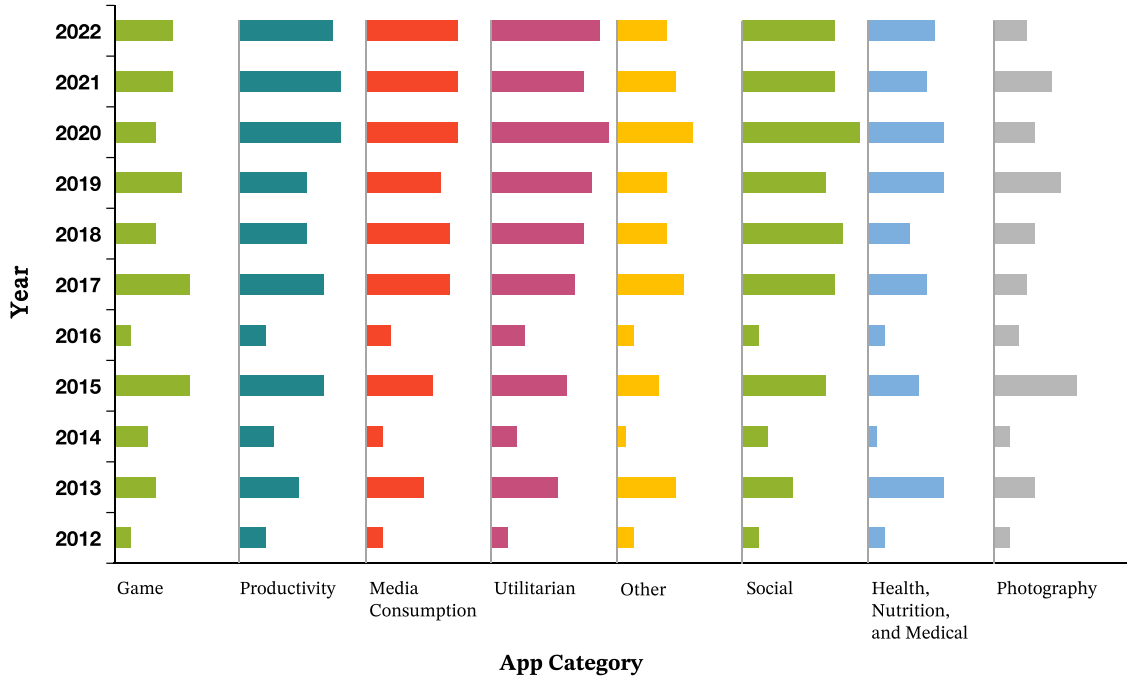

Fig. 12. Trend of app categories. Note: The list of app categories aligns with the framework introduced by Huebner et al. (2020).

for categorizing reviews (i.e., *App Aspect*, *Issue & Complaint*, *Software Evolution*, *Security & Privacy*, *Undesired Behavior*, *User Experience*, *User Requirement*, and *Other*). For each of these review categorization standards, we present an example taxonomy defined based on the standard, which is shown in Table 9.

Findings: As depicted in Fig. 13, the exploration of review content categorization started in 2013, featuring five distinct standards, i.e., *App Aspect*, *Issue & Complaint*, *Software Evolution*, *User Experience*, and *Other*. Notably, *Software Evolution* emerged as the predominant standard. The utilization frequencies of standards *App Aspect*, *Issue & Complaint*, and *Other* remained relatively steady from 2013 to 2022. In contrast, *User Experience* was the most used categorizing standard in 2013 but was rarely used until 2022. Beginning around 2017, there was a notable increase in attention towards standards *Security & Privacy*, *Undesired Behavior*, and *User Requirement*, with *User Requirement* particularly standing out. The year 2021 witnessed the most analysis of review categorization, with the utilization frequencies of standards *App Aspect* and *User Intention* unexpectedly exhibiting remarkable numbers.

In addition to showing the evolution of review categorization standards over time, we also present the distributions of different review categorization standards across different app categories, which is shown in Fig. 14. It is evident that *Software Evolution* stands out as the predominant review categorization standard in each app category. Concerning other review categorization standards, Game and Social apps exhibit a relatively even distribution of different categorization standards across various app categories. In the app categories Productivity and Utilitarian, *App Aspect* and *User Experience* are more frequently utilized standards. As for app categories Media Consumption and Other, *App Aspect* and *User Requirement* are more prevalent. In the app categories Health, Nutrition, and Medical, and Photography, *Undesired Behavior*, *Issue & Complaint*, and *Security & Privacy* are more commonly employed.

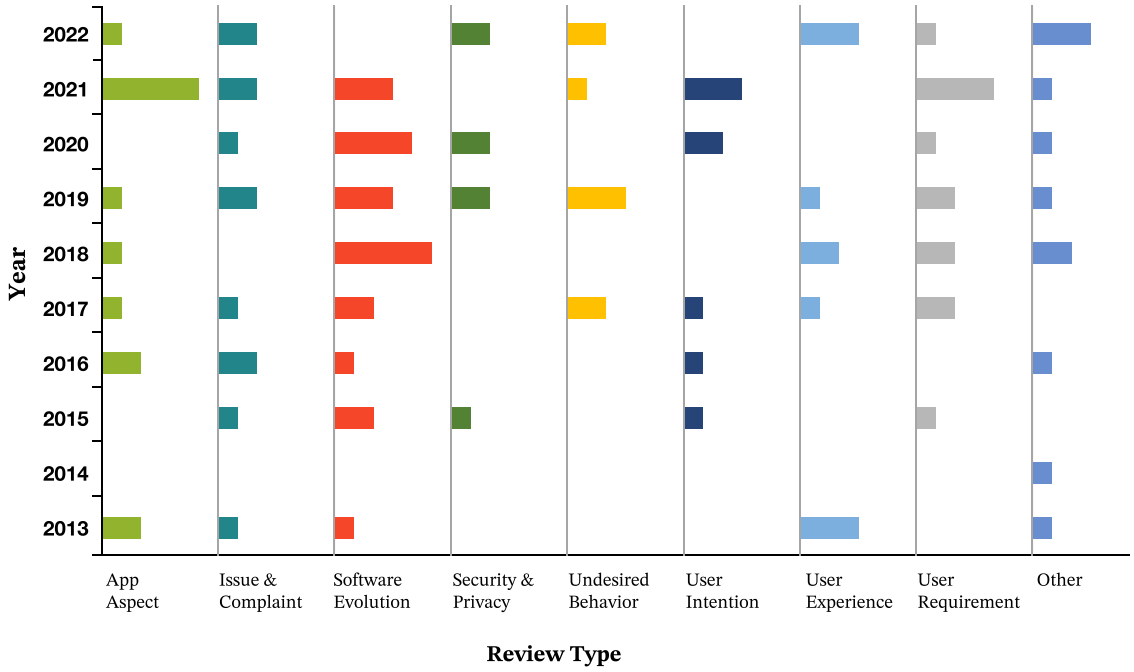

Fig. 13. Trend of review types.

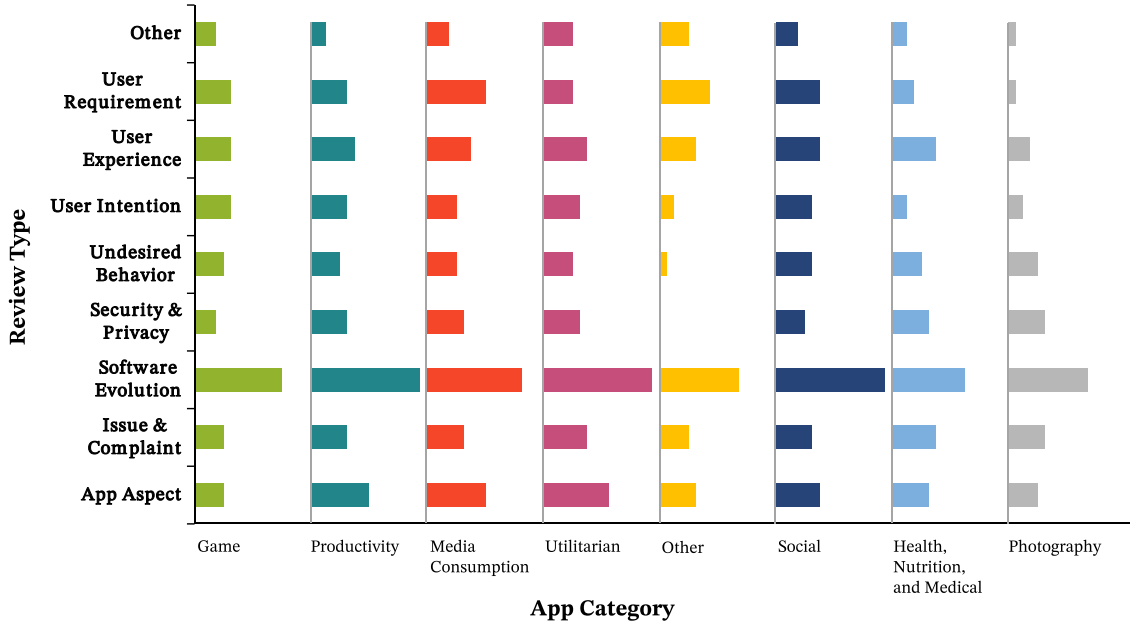

Fig. 14. Review types across different app categories.

Answer to RQ 2: User reviews are made of star ratings and short free text, which are written in an informal way, containing miss-spelled words and seldom obeying grammar rules. They are triggered by app updates and are written for specific app versions. App reviews present two extremes in terms of informativeness, either expressing sentiment or concerning multiple aspects of app development and maintenance.

5. Exploiting app review characteristics

In this section, we answer **RQ 3: How do previous studies handle the characteristics of app reviews and exploit them in the software ecosystem?** According to the user journey map, researchers explore and observe the characteristics of the review, and then effectively mine the information in the review according to the characteristics. When mining app reviews, researchers may wonder how to deal with these characteristics and how to apply these characteristics. Hence, we present our findings from these two folds.

Table 10
Handling and applications for app review rating.

Types	Techniques	Papers/References
Rating handling: constructing features/metrics	Average rating	Chen et al. (2014), Palomba et al. (2015), Huebner et al. (2020), McIlroy et al. (2017), Zhu et al. (2015), Zhang et al. (2020b,a), Scherr et al. (2017), Keertipati et al. (2016), Li et al. (2017), Gao et al. (2022a), Hassan et al. (2022), Dalpiaz and Parente (2019), Hu et al. (2019a), Huebner et al. (2018)
	Number of ratings	Kuehnhausen and Frost (2013), Groen et al. (2017)
	Rating-extremity	Gao et al. (2022a)
	Proportion of low, neutral, and positive ratings	Noei et al. (2018, 2019), Hu et al. (2019c,b)
	Ratings distribution type	Kuehnhausen and Frost (2013), Gao et al. (2018a)
App review selection	Selecting only app reviews with low star ratings	Khalid et al. (2015), Fu et al. (2013), Guo and Singh (2020), Khalid et al. (2016), Licorish et al. (2017), Keertipati et al. (2016), Eiband et al. (2019), Fereidouni et al. (2022), Tushev et al. (2022), Ebrahimi and Mahmoud (2022), McIlroy et al. (2016), Hu et al. (2019b)
Descriptive statistical analysis	Visualizing rating distribution	Vasa et al. (2012), Hoon et al. (2012), Guzman and Paredes Rojas (2019), Guzman et al. (2014), Rahman et al. (2017), Huebner et al. (2020), Hu et al. (2021), Scherr et al. (2017), Li et al. (2017), Zhu et al. (2014), Hassan et al. (2022), Wang et al., Hu et al. (2019a), Huebner et al. (2018), Kurtanović and Maalej (2018), Martens and Maalej (2019b), Hu et al. (2019b), Noei et al. (2017), Alqahtani and Orji (2020, 2019)
	Tracking rating changes	Gao et al. (2018a), Chen et al. (2021), Palomba et al. (2015), McIlroy et al. (2017), Hassan et al. (2018), Scherr et al. (2017), Li et al. (2017), Hassan et al. (2022)
Inferential statistical analysis	Correlation analysis	Martens and Maalej (2019a), Zhang et al. (2020b)
	Linear regression model	Fu et al. (2013), Licorish et al. (2017), Wu et al. (2021)
	Logistic regression model	Hassan et al. (2018), Nguyen et al. (2019)
	PMVD	Noei et al. (2021)
	Wilcoxon test	Martin et al. (2015)
	t-test	Alqahtani and Orji (2020)
	Kruskal–Wallis test	Hassan et al. (2022)
	Mann–Whitney U test	Hu et al. (2019a)
	Cliff's delta (δ)	Hu et al. (2019a)
Utilizing review rating as an input feature	Clustering feature	Noei et al. (2019), Uddin et al. (2020a)
	Classification feature	Alshangiti et al. (2022), Kurtanović and Maalej (2017), Scalabrino et al. (2019), Guzman et al. (2015b), Rahman et al. (2017), Srisopha et al. (2020b,a), Wang et al., Kurtanović and Maalej (2018)
	Prioritization feature	Chen et al. (2014), Phong et al. (2015), Gao et al. (2018a), Wei et al. (2017), Etaiwi et al. (2020), Man et al. (2016), Gao et al. (2015), Etaiwi et al. (2020), Keertipati et al. (2016), Gao et al. (2022a)
	Feature for prediction model	Zhang et al. (2020b,a)
	Feature for seq2seq model	Gao et al. (2019), Farooq et al. (2020)
Comparison	Comparing different app groups by rating	Chen et al. (2021), Huebner et al. (2020), Hassan et al. (2022)
Relationship with app updates	Detecting bad app updates	Hassan et al. (2020)
	Predicting ratings for specific app versions	Yao et al. (2017)

5.1. Rating

In this subsection, we present the handling and applications for review rating. The relevant practices are briefly shown in Table 10.

5.1.1. Handling rating

Since the rating is numeric, it can be directly used or be used to construct metrics. For example, Chen et al. (2014) directly used the value of star rating as a ranking feature. On the contrary, Licorish et al. (2017) calculated the average rating among all the review sentences to associate the rating with the app feature. Besides, there are some new features constructed based on review rating, e.g., rating-extremity (Gao et al., 2022a), proportion of low, neutral, and positive ratings (Noei et al., 2018, 2019), percentage of five-star ratings of reviews (Hu et al., 2019c,b), and ratings distribution type (Kuehnhausen and Frost, 2013; Gao et al., 2018a).

5.1.2. Application of rating

The applications of rating include app review selection, performing statistical analysis, using the review rating as an input feature, comparison, and relationship with app update.

App review selection. Low star rating is a frequently used indicator for selecting app reviews. App reviews with low star ratings can be used to identify common complaints conveyed in app reviews (Fu et al., 2013; Khalid et al., 2015, 2016), extract user stories (Guo and Singh, 2020), and recommend app features for software maintenance. However, there is no uniform standard for low star rating, and previous studies have set the threshold as three stars (Khalid et al., 2016), two stars (Fu et al., 2013; Khalid et al., 2015), and one star (Guo and Singh, 2020).

Performing statistical analysis. The applications of using review rating for statistical analysis can be divided into two categories, descriptive statistical analysis, and inferential statistical analysis. The

former focuses on describing and summarizing known information, while the latter focuses on finding patterns in the data to infer unknown information and then make decisions and predictions.

- **Descriptive statistical analysis.** Common descriptive statistical analysis includes visualizing rating distribution (Guzman et al., 2014; Rahman et al., 2017), and tracking rating changes (Hassan et al., 2018).
- **Inferential statistical analysis.** Review rating can be used as either an explanatory variable or an explained variable in inferential statistical models. For instance, Licorish et al. (2017) used rating as one of the explanatory variables of the multi-linear regression model to determine the problem severity level while Fu et al. (2013) used rating as the explained variable to identify informative words. Apart from the linear regression model, the logistic regression model is also commonly adopted. For instance, Hassan et al. (2018) used rating to prioritize the reviews to read and respond to, and Nguyen et al. (2019) used rating to predict security- and privacy-related app updates. Instead of using the explicit rating score, Wu et al. (2021) designed a rating-related feature (i.e., the counts of positive reviews) to identify key features via a multi-linear regression model. Besides, Noei et al. (2021) applied Proportional Marginal Variance Decomposition (PMVD) to identify key review topics. In addition, the hypothesis test is applied to rating to make decisions. For instance, Martin et al. (2015) used the Wilcoxon test to investigate app properties (e.g., rating) affected by varying dataset completeness.

Utilizing review rating as an input feature. Review rating can be used as a feature for clustering (Uddin et al., 2020a), classification (Kurtanović and Maalej, 2017; Scalabrino et al., 2019; Rahman et al., 2017), prioritization (Chen et al., 2014; Phong et al., 2015), and prediction models (Zhang et al., 2020b,a). Except for the explicit rating score, there are constructed rating features, e.g., the counts of negative and positive ratings (Phong et al., 2015; Groen et al., 2017). Moreover, previous studies (Gao et al., 2019; Farooq et al., 2020) incorporated review rating into the seq2seq models (e.g., RNN, LSTM) to automatically generate responses for user reviews.

Comparison. Review rating can serve as a metric to compare different app groups. For example, Hassan et al. (2022) compared the rating between peer apps and all the apps.

Relationship with app updates. Review rating can also be used to detect bad app updates (Hassan et al., 2020), and to predict ratings for specific app versions (Yao et al., 2017).

5.1.3. Take-home messages

Correlation with sentiment. By presenting top words across all ratings and analyzing the distribution of a specific sentiment detection word set amongst the ratings, Hoon et al. (2012) drew an empirical inference that app rating is associated with the words expressing sentiment. This finding helps to determine whether the review text is consistent with the rating, thus filtering inconsistent reviews (Fu et al., 2013). On the contrary, there are studies using rating to adjust or compute the sentiment scores (Gu and Kim, 2015; Gao et al., 2018b).

5.2. Length

In this subsection, we present the handling and applications for review length. The relevant practices are briefly shown in Table 11.

5.2.1. Handling length

We only find one practice for handling review length among all the papers we have surveyed. Oehri and Guzman (Oehri and Guzman, 2020) condensed the longer reviews to mitigate the length difference between review pairs thus facilitating calculating review similarity.

5.2.2. Application of length

The applications of length include app review selection, performing statistical analysis, using the review length as an input feature, comparison, model selection, and determining model input.

App review selection. Filtering out extremely short reviews is widely adopted by previous studies (Di Sorbo et al., 2016; Palomba et al., 2017; Zhou et al., 2021; Gao et al., 2018b; Farooq et al., 2020; Panthum and Senivongse, 2021; Haering et al., 2021; Fereidouni et al., 2022).

Performing statistical analysis. The applications of using review length for statistical analysis can also be divided into two categories, descriptive statistical analysis, and inferential statistical analysis. Hassan et al. (2018) used length as one of the explanatory variables of the logistic regression model to prioritize the reviews to read and respond to. Besides, the hypothesis test is applied to rating to make decisions. For instance, Martin et al. (2015) used the Wilcoxon test to investigate app properties (e.g., length) affected by varying dataset completeness.

Utilizing review length as an input feature. Review length can be used as a feature for classification (Panthum and Senivongse, 2021; Scalabrino et al., 2019; Kurtanović and Maalej, 2017; Rahman et al., 2017) and prioritization (Gao et al., 2018a; Man et al., 2016; Gao et al., 2015, 2018b; Di Sorbo et al., 2016). In addition to being an input feature of a traditional machine learning model, review length can also benefit the deep learning model (Gao et al., 2019).

Comparison. Review length can serve as a metric for comparing various app groups (McIlroy et al., 2016; Martens and Maalej, 2019b) and review categories (Kurtanović and Maalej, 2018).

Model selection. Wang et al. (2022a) selected a suitable pre-trained model according to the short text nature of app reviews.

Determining model input. Zhang et al. (2019) adopted the skip bi-gram collocations rather than bag of words for LDA to generate more meaningful topics. On the contrary, when performing review classification, Anam and Yeasin (2013) found that the performance of the Naive Bayes classifier is better when using unigrams as input compared to using bigrams.

5.2.3. Take-home messages

Hoon and his colleagues (Hoon et al., 2012; Vasa et al., 2012) statistically analyzed 8.7 million user reviews from 17,330 top free and paid apps on the Apple App Store, finding that the median feedback length across all applications is 69 characters and the mean is 117, indicating that users tend to leave short feedback. The short text nature of app reviews should be noticed when mining app reviews. On the one hand, short texts are more ambiguous since they have insufficient contextual information (Chen et al., 2019). On the other hand, common techniques designed for long text (e.g., paragraphs or documents) should be carefully applied for short reviews. For example, common topic models, e.g., LDA, and OLDA, are not suitable for short text like app reviews (Hadi and Fard, 2020; Wang et al., 2020). We present the tips for review length in Table 12.

Processing short reviews. Firstly, researchers can set the review length threshold for filtering out extremely short reviews, e.g., reviews with less than three words (Panthum and Senivongse, 2021), reviews with less than ten words (Haering et al., 2021). Such processing can also be applied to review sentence level, e.g., discarding all sentences with less than three words (Di Sorbo et al., 2016; Palomba et al., 2017; Zhou et al., 2021) or four words (Gao et al., 2018b). Apart from setting the length threshold, researchers can make reviews longer by concatenating the review title and review text to a single string (Vasa et al., 2012; Oehri and Guzman, 2020), concatenating comments from the same app together as a new document (Fu et al., 2013), or augmenting each review with several most similar words (Lu and Liang, 2017). Moreover, researchers can adopt representation methods that are suitable for short text. For instance, Wei et al. (2017) represented reviews via Microsoft Concept Graph, which can better understand

Table 11
Handling and applications for app review length.

Types	Techniques	Papers/References
Length handling	Condensing longer reviews	Oehri and Guzman (2020)
App review selection	Filtering out extremely short reviews	Hoon et al. (2013), Panthum and Senivongse (2021), Di Sorbo et al. (2016), Palomba et al. (2017), Zhou et al. (2021), Gao et al. (2018b), Farooq et al. (2020), Panthum and Senivongse (2021), Haering et al. (2021), Yang and Liang (2015), Carreño and Winbladh (2013), Hu et al. (2019c,b), Ebrahimi and Mahmoud (2022), Assi et al. (2021)
Inferential statistical analysis	Logistic regression model Wilcoxon test	Hassan et al. (2018) Martin et al. (2015)
Utilizing review length as an input feature	Classification feature	Alshangiti et al. (2022), Guzman et al. (2015b), Panthum and Senivongse (2021,?), Scalabrino et al. (2019), Kurtanović and Maalej (2017), Rahman et al. (2017), Srisopha et al. (2020b,a), Gao et al. (2022a), Maalej et al. (2016), Kurtanović and Maalej (2018), Martens and Maalej (2019b)
	Prioritization feature	Man et al. (2016), Gao et al. (2018a), Man et al. (2016), Gao et al. (2015, 2018b), Iacob and Harrison (2013), Di Sorbo et al. (2016), Gao et al. (2022a)
	Feature for seq2seq model	Gao et al. (2019)
Comparison	Comparing different app groups by length	McIlroy et al. (2016), Martens and Maalej (2019b)
	Comparing different review categories by length	Kurtanović and Maalej (2018)
Model selection	Selecting suitable pre-trained model	Wang et al. (2022a)
Determining model input	Adopting the skip bi-gram collocations rather than bag of words for LDA	Zhang et al. (2019)
	Naive Bayes classifier with unigram outperformed the one with bigram	Anam and Yeasin (2013)

Table 12
Take-home messages for review length.

Types	Techniques	Papers/References
Processing short reviews	Setting the review length threshold for filtering out extremely short reviews	Hoon et al. (2013), Panthum and Senivongse (2021), Di Sorbo et al. (2016), Palomba et al. (2017), Zhou et al. (2021), Gao et al. (2018b), Farooq et al. (2020), Panthum and Senivongse (2021), Haering et al. (2021), Yang and Liang (2015), Carreño and Winbladh (2013), Hu et al. (2019c,b), Ebrahimi and Mahmoud (2022), Assi et al. (2021)
	Concatenating the review title and review text to a single string	Vasa et al. (2012), Oehri and Guzman (2020), McIlroy et al. (2016), Yang and Liang (2015)
	Concatenating comments from the same app together as a new document	Fu et al. (2013)
	Augmenting each review with several most similar words	Lu and Liang (2017)
	Adopting representation methods suitable for short text	Wei et al. (2017), Zhou et al. (2021)
Applying topic modeling to short reviews	LDA (concatenating comments from the same app together as a new document)	Fu et al. (2013)
	BTM	Hadi and Fard (2020), Wang et al. (2020)
	BST	Gao et al. (2022a,b)
Adopting suitable metrics for short reviews	Asymmetric Dice coefficient	Palomba et al. (2017)

short text. Faced up with the sparse and high-dimensional word-review matrix, Zhou et al. (2021) used PCA to reduce the dimension.

Applying topic modeling to short reviews. Topic modeling allows researchers and developers to take a glance at the discussed topics of a large number of app reviews, which is a preliminary step for further analysis. However, traditional topic models such as Latent Dirichlet Allocation (LDA) and Probabilistic Latent Semantic Analysis (PLSA) exhibit suboptimal performance when confronted with short text (Hadi and Fard, 2020; Guzman and Maalej, 2014). To mitigate this problem, Fu et al. (2013) concatenated user reviews from the same app together to generate a long document. Besides, bi-gram, compared with unigram, can better represent the respective topic. Therefore, topic

Table 13
Handling for domain knowledge.

Types	Techniques	Papers/References
Constructing features	Using apps' historical ranking records to quantify the ranking patterns in terms of rising, maintaining, and recession phases	Zhu et al. (2015)
	User name-based features	Wang et al.
App review selection	Selecting reviews with greater than 50 total helpfulness votes	Besmer et al. (2020)
Topic modeling	Simultaneously modeling app descriptions and user reviews with topic models	Park et al. (2015)

Table 14
Applications for domain knowledge: combining with app metadata and review metadata.

Types	Techniques	Papers/References
App category	Visualizing the distribution of reviews by app categories	Hu et al. (2019b), Wang et al., Hatamian et al. (2019)
	Logistic regression model	Nguyen et al. (2019)
	Classification feature	Wang et al. (2022a), Jha and Mahmoud (2019)
	Prioritization feature	Hassan et al. (2018)
	Feature for seq2seq model	Gao et al. (2019), Farooq et al. (2020)
	Computing app similarity	Yao et al. (2017)
App ID/app name	Prioritization feature	Hassan et al. (2018)
	Retrieving the app-specific reviews and responses	Farooq et al. (2020)
App install	Visualizing the distribution of app installs	Hu et al. (2019b)
App ranking	Tracking ranking changes	Hassan et al. (2022)
	Comparing app ranking of different treatments	Hassan et al. (2022)
	Inferential statistical analysis: Wilcoxon test	Hassan et al. (2022)
	Ranking-based features for detecting ranking fraud	Zhu et al. (2015)
App description	Identifying app features	Wu et al. (2021), Wang et al. (2020), Uddin et al. (2020a)
	Using app descriptions as app-specific information to augment app reviews	Farooq et al. (2020), Gao et al. (2021)
	Matching user reviews with app features extracted from app descriptions	Wu et al. (2021)
	Using app descriptions to help the manual coding process	Licorish et al. (2017)
App market policy	Deriving a taxonomy of 26 undesired behaviors	Hu et al. (2021)
User name	Visualizing the distribution of users	Wang et al.
	Utilizing user name-based features for review removal detection	Wang et al.
	Utilizing self-similarity of reviewers to detect app review manipulation	Li et al. (2017)
	Identifying fraudulent reviewers and marking all their reviews as fake reviews	Hu et al. (2019c,b)
Helpfulness vote	Utilizing helpfulness vote to select high quality reviews	Harkous et al. (2022)
	Visualizing the distribution of helpfulness votes	Gao et al. (2022a), Fereidouni et al. (2022)
	Classification feature	Srisopha et al. (2020a)
	Ground truth for predicting review helpfulness	Gao et al. (2022a)

models adapted for short text are exploited, e.g., Biterm Topic Modeling (BTM) (Hadi and Fard, 2020; Wang et al., 2020), Biterm-based Sentiment-Topic Modeling (BST) (Gao et al., 2022a,b).

Adopting suitable metrics for short reviews. Palomba et al. (2017) employed the asymmetric Dice coefficient to measure the similarity between the app review cluster and the source code file since the former is notably shorter. Taking into account the minimum cardinality of the word sets, the asymmetric Dice coefficient provides a more effective means of measuring document similarity.

Although the short text nature of app reviews poses a challenge for mining reviews, there are researchers who focus on extremely short reviews. For instance, Hoon et al. (2013) retrieved reviews with up to five words to analyze the sentiment contained in app reviews.

5.3. Domain knowledge

In Section 4.3.4, we previously discussed that app reviews, often written by non-expert users, lack sufficient detail in precise technical language. To address this limitation, previous studies jointly mined app reviews together with other data containing domain knowledge (e.g., bug report, change log, and source code). In this subsection, we present the common handling and applications for domain knowledge. The relevant practices are briefly shown in Table 13, Table 14, and Table 15.

5.3.1. Handling domain knowledge

According to our survey, handling domain knowledge includes constructing features, app review selection, and topic modeling, as shown in Table 13. To be specific, Zhu et al. (2015) used apps' historical ranking records to quantify the ranking patterns in terms of the rising, maintaining, and recession phases. Wang et al. combined user name and app review to construct features, i.e., *% removed reviews per user*, *% ratings per user*. Besmer et al. (2020) filtered reviews with greater than 50 total helpfulness votes to analyze highly engaging privacy reviews. Park et al. (2015) simultaneously modeled app descriptions and user reviews with topic models to mitigate the lexicon gap between developers and users.

5.3.2. Applications of domain knowledge

As shown in Tables 14 and 15, previous studies jointly mined app reviews with various types of data, including app store information, bug/issue report, changelog/release note, static analysis information, source code, and non-SE information.

Combining with app store information. We summarize the app store information in two folds, i.e., app metadata (Fu et al., 2013) and review metadata (Pagano and Maalej, 2013).

- **Combining with app metadata.** According to our survey, the application practices related to app metadata (Fu et al., 2013) encompass app category, app ID/app name, app install, app ranking, and app description. Regarding app category, Wang et al. visualized the distribution of the percentage of removed reviews for app categories. Besides, Hu et al. (2019b) visualized the distribution of app categories and app installs. Nguyen et al. (2019) used the app category as an independent variable for logistic regression to predict security- and privacy-related app updates. Moreover, previous

Table 15
Applications for domain knowledge: combining with other sources of information.

Types	Techniques	Papers/References
Combining with bug/issue report	Linking app reviews and bug/issue reports	Haering et al. (2021), Palomba et al. (2015), Wang et al. (2022b), Noei et al. (2019)
	Linking app reviews and issues/pull requests	Hu et al. (2022)
	Using bug reports to construct the ground truth of the mappings from reviews to code	Yu et al. (2022)
	Using issues/pull requests to generate a dataset for <app review, source code method> pairs	Hu et al. (2022)
Combining with changelog/release note	Visualizing the changes of issues in app reviews along with versions and changelogs	Gao et al. (2022b)
	Matching app reviews to release notes	Wang et al. (2021)
	Linking app feature in reviews and release notes	Uddin et al. (2020b), Haggag (2022)
	Utilizing changelogs to improve app review classification	Wang et al. (201b8)
Combining with static analysis information	Linking app reviews and static analysis warnings	Khalid et al. (2016)
	Using app reviews to determine the priority level of warnings in static analyzer	Wei et al. (2017)
Combining with source code	Using app review to localize problematic source code	Ciurumelea et al. (2017), Palomba et al. (2017), Zhou et al. (2021), Zhang et al. (2021)
Combining with non-SE information	Linking app reviews and tweets	Yadav et al. (2020)
	Combining with microblogs to generate competitive event storyline	Ouyang et al. (2018)
	Combining with news articles to provide in-depth and broader contextualization	Bano et al. (2021)
	Using phrase template mined from Amazon product reviews to apply on app reviews	Vu et al. (2016)

studies utilized the app category as input features, i.e., classification feature (Wang et al., 2022a; Jha and Mahmoud, 2019), prioritization feature (Hassan et al., 2018), feature for seq2seq models (e.g., RNN, LSTM) (Gao et al., 2019; Farooq et al., 2020). Besides, the app category was used to construct an app similarity matrix (Yao et al., 2017). Regarding app ID/app name, it was used to prioritize the reviews with a logistic regression model (Hassan et al., 2018) and to retrieve the app-specific reviews and responses to facilitate response generation for user reviews (Farooq et al., 2020). Regarding app ranking, Hassan et al. (2022) tracked ranking changes and compared the app' ranking within its peer groups and globally to study competitor apps. Zhu et al. (2015) constructed ranking-based features for detecting ranking fraud. Among the various app metadata, app description is often jointly mined with app reviews. We summarize the reasons mentioned in the surveyed papers as: (1) app descriptions benefiting from a more formal form of writing (Noei et al., 2018), which makes it easier to extract information; (2) app descriptions mainly conveying app features (Park et al., 2015), which allows researchers to identify app features (Wu et al., 2021; Wang et al., 2020; Uddin et al., 2020a). Apart from directly extracting feature-describing words and phrases, researchers can simultaneously model app descriptions and user reviews with topic models to identify key app features (Park et al., 2015), use app description as app-specific information to augment app review (Farooq et al., 2020; Gao et al., 2021), match user review with app feature extracted from app description (Wu et al., 2021). Moreover, app descriptions can help the manual coding process (Licorish et al., 2017). Regarding app market policy, Hu et al. (2021) manually created a taxonomy of 26 undesired behaviors from the app market policies.

- **Combining with review metadata.** According to our survey, the application practices related to review metadata (Pagano and Maalej, 2013) encompass user name and helpfulness vote. Regarding the user name, Wang et al. visualized the distribution of users of removed reviews and construct user name-based features for review removal detection. Li et al. (2017) utilized the self-similarity of reviewers to detect app review manipulation. Regarding the helpfulness vote, its applications involve selecting high quality reviews (Harkous et al., 2022), helpfulness votes distribution visualization (Gao et al., 2022a; Fereidouni et al., 2022), acting as a classification feature (Srisopha et al., 2020a), acting as ground truth for predicting review helpfulness (Gao et al., 2022a).

Combining with bug/issue report. The bug report is often jointly mined with app reviews to aid the software maintenance process as it contains information on bugs. Haering et al. (2021) matched the informal app reviews reporting problems to the professional bug reports in issue trackers by using bug summaries. By contrast, Palomba et al. (2015) linked informative reviews and issue reports by using issue descriptions. Hu et al. (2022) linked app reviews and issues/pull requests, thus generating a dataset made of <app review, source code method> pairs. Similarly, Yu et al. (2022) exploited bug reports to construct the ground truth of the mappings from reviews to code.

Combining with changelog/release note. Changelogs and release notes reflect changes in apps. Compared with traditional desktop software, mobile apps are updated more frequently. Hence, it is essential to monitor and track the app changes. For instance, Gao et al. (2022b) visualized the changes of detailed issues in app reviews along with versions and changelogs. Wang et al. (2021) matched app reviews to release notes. Furthermore, Uddin et al. (2020b) linked app features in reviews and release notes to capture the changes of app features. Wang et al. (201b8) utilized changelogs to improve app review classification.

Combining with static analysis. Static analyzers are used to detect errors in Android applications. On the one hand, static analysis provides extra information for app review analysis. For instance, Khalid et al. (2016) conducted a case study to confirm the relationship between user complaints in app reviews and static analysis warnings by linking app reviews and static analysis warnings. Their study provides evidence that static analysis can help developers find solutions to some user complaints expressed in app reviews. On the other hand, app reviews are used to improve the results of the static analyzer. Since many false warnings are generated by static analyzers, app review can be used to determine the warning priority level (Wei et al., 2017).

Combining with source code. Using app review to localize problematic source code is a popular task (Ciurumelea et al., 2017; Palomba et al., 2017; Zhou et al., 2021; Zhang et al., 2021). The common processes are as follows: (1) identifying app reviews conveying complaints; (2) localizing problematic source code by calculating the semantic similarity between app reviews and source code. For the second step, instead of using the massive and noisy code directly, researchers can use the names of source code elements (e.g., fields, methods, and classes) (Palomba et al., 2017). The accuracy in code localization can be improved by integrating information from historical issue reports (Zhang et al., 2021), commit messages (Zhou et al., 2021), or

app components (e.g., GUI-related terms) (Wei et al., 2017). Besides, source code can be used to aid app review analysis. For instance, Palomba et al. (2015) augmented textual commits with the names of the classes being changed in the source code to link informative reviews and commits.

Combining with non-SE information. Yadav et al. (2020) linked app reviews and tweets to integrate semantics into app review analysis. Ouyang et al. (2018) adopted microblogs to generate competitive event storylines. Similarly, Bano et al. (2021) utilized news articles to provide in-depth and broader contextualization. Vu et al. (2016) applied phrase templates mined from Amazon product reviews on app reviews.

5.3.3. Take-home messages

Mitigating the difference between user-generated app reviews and professional structural technical documents. Different from technical documents (e.g., issue reports and source code), user reviews are contributed by individuals who may have no or less knowledge of mobile application development, thus lacking relevant domain knowledge. As a result, app users may not be able to provide enough detail in precise language (Haering et al., 2021), thus providing no actionable information for app developers to maintain or improve apps. To mitigate this problem, researchers exploit various information to bridge the gap, e.g., source code (Wei et al., 2017; Yu et al., 2022), Android app components such as activities or broadcast receivers (Wei et al., 2017; Yu et al., 2022), commits (Zhou et al., 2021), issue reports (Zhang et al., 2021), or app market policies (Hu et al., 2021).

5.4. Textual content

For the textual content, previous studies normally do preprocessing and then extracting features. The relevant practices are briefly shown in Tables 16 and 17.

5.4.1. Handling textual content: Text preprocessing

Since app reviews are in natural language, common natural language processing steps are necessary to remove redundant information (Maalej and Nabil, 2015), e.g., stopwords removal, lemmatization, and stemming. Moreover, as a type of software artifact (Morales-Ramirez et al., 2015), app review needs specific handling for its domain-specific content, e.g., customized stopwords removal, filtering out inconsistent or uninformative reviews. Generally, we introduce text preprocessing practices including noise removal, informal language correction, noise reduction, review-level filtering, and other practices.

Noise removal. For noise removal, we summarized three types of practices from the surveyed papers, i.e., filtering out non-English reviews, stopwords and common words removal, and uncommon words removal.

- **Filtering out non-English reviews.** Filtering out non-English reviews is a common practice. There are three ways in previous work. The first is only to collect reviews written in English when collecting reviews (Ha and Wagner, 2013; Nguyen et al., 2019; Wang et al., 2020). Google's Compact Language Detector can be used to realize this practice. The second is to filter English reviews after collecting reviews in any language (Srisopha et al., 2019; Noei et al., 2021; Panthum and Senivongse, 2021; Man et al., 2016; Farooq et al., 2020). In particular, researchers can simply remove all non-English characters (Wu et al., 2021) or remove app reviews that contain a significant amount/ratio of non-English words (Fu et al., 2013; Phong et al., 2015). The third is to translate non-English reviews into English (Yin and Pfahl, 2018). It should be noticed that app reviews in other languages can also follow the above practices. For example, Wu et al. (2021) filtered both Chinese and English feedback from the raw data and extended standard stopwords list of Chinese language.
- **Stopwords and common words removal.** The simplest way is to remove words in the standard stopwords list (Maalej and Nabil, 2015; Chen et al., 2014; Noei et al., 2021). However, app review has its own stopwords now that it is a specific type of text. To mitigate this problem, a lot of previous studies customized their own stopword list, either by adding more words common in app reviews into the standard list or by removing words significant for review mining from the standard list (Grano et al., 2018; Palomba et al., 2015; Panthum and Senivongse, 2021; Hu et al., 2021). For example, the names of the applications (Guzman and Maalej, 2014; Johann et al., 2017), the HTML tags (Oehri and Guzman, 2020; Fu et al., 2013; Besmer et al., 2020), the URLs (Oehri and Guzman, 2020; AlOmar et al., 2021), uninformative abbreviations (e.g., "ur") (Gao et al., 2018b; Zhou et al., 2021), emotional words (e.g., "bad" and "nice") (Gao et al., 2022b), everyday words (e.g., "good" and "much") (Gao et al., 2015) are manually added to the standard stopword list. Besides, there are researchers adopting more subjective approaches to identify common words, e.g., applying TF-IDF method (Wei et al., 2017), removing words that appear in 30% of app reviews (Park et al., 2015), building a dictionary of stop words based on RANKS NL (Wang et al., 2020). In addition, it is necessary to remove important words from the standard stopword list since there are stopwords relevant to user intentions (Maalej and Nabil, 2015). For instance, the terms "should" and "must" might indicate a feature request, "did", "while" and "before", and "now" a bug report.
- **Uncommon words removal.** Analyzing 8.7 million user reviews, Hoon et al. (2012) found that 17% of the words appear less than three times in these reviews. Later on, previous studies made some attempts to filter out infrequent words (Fu et al., 2013; Yao et al., 2017). For example, Park et al. (2015) removed words that appear in less than five reviews. Rahman et al. (2017) removed words that appear at most once after conducting lemmatization. Apart from removing rare words, there are researchers removing short words, e.g., words with less than three characters (Palomba et al., 2017; Park et al., 2015; Lu and Liang, 2017).
- **Removing punctuation and special characters.** Removing punctuation and special characters is a basic practice to remove noise in the text. Therefore, it is widely adopted in app review mining (Oehri and Guzman, 2020; Noei et al., 2021; Martin et al., 2015; Panthum and Senivongse, 2021; Ciurumelea et al., 2017; Zhou et al., 2021; Park et al., 2015). In particular, the emoji characters/icons are also filtered out in this process (Zhou et al., 2021; Obie et al., 2022; Tao et al., 2020).

Informal language correction. As informal text generated on social media, app reviews tend to contain typos, acronyms, and abbreviations, correcting these words can benefit mining app reviews in automatic ways (Phong et al., 2015). Phong et al. (2015) used a custom dictionary to correct misspelled words, acronyms, and abbreviations and Gao et al. (2018b, 2022b, 2019, 2021) followed this method. Grano et al. (2018) corrected mistakes via a systematic Information Retrieval (IR) preprocessing. Besides, there are online dictionaries and tools that can help this practice, e.g., Noslang,[23] (Panthum and Senivongse, 2021) gingerit library (Panthum and Senivongse, 2021), English vocabulary module of the JLanguageTool (Wei et al., 2017), English vocabulary of PYENCHANT library (Palomba et al., 2017; Zhang et al., 2021), neelindresh/ContractedText.py (Panthum and Senivongse, 2021). Instead of correcting the informal expressions, Man et al. (2016) directly filtered out these words.

Noise reduction. Common practices for noise reduction inlcude lowercasing (Oehri and Guzman, 2020; Fu et al., 2013; Chen et al., 2014), lemmatization (Maalej and Nabil, 2015; Guzman and Maalej,

[23] www.noslang.com

Table 16
Handling: preprocessing for app review content.

Types	Techniques	Papers/References
Noise removal	Filtering out non-English reviews	Srisopha et al. (2019), Phong et al. (2015), Gao et al. (2022b), Panthum and Senivongse (2021), Zhang et al. (2021), Wu et al. (2021), Hoon et al. (2013), Man et al. (2016), Ha and Wagner (2013), Nguyen et al. (2019), Wang et al. (2020), Farooq et al. (2020), Vu et al. (2016), Eiband et al. (2019), Yu et al. (2022), Fereidouni et al. (2022), Wang et al. (2022a), Noei et al. (2019), Assi et al. (2021), Tao et al. (2020)
	Stopwords and common words removal	Alshangiti et al. (2022), Haggag (2022), Guzman et al. (2014), Dhinakaran et al. (2018), Maalej and Nabil (2015), Chen et al. (2014), Martin et al. (2015), Noei et al. (2021), Lu and Liang (2017), Man et al. (2016), Gao et al. (2015, 2022b), Wei et al. (2017), Panichella et al. (2015), Wu et al. (2021), Palomba et al. (2015), Zhou et al. (2021), Zhang et al. (2021), Ciurumelea et al. (2017), AlOmar et al. (2021), Panthum and Senivongse (2021), Rahman et al. (2017), Zhu et al. (2015), Yadav et al. (2020), Wang et al. (201b8), Guzman et al. (2015b), Morales-Ramirez et al. (2017), Eiband et al. (2019), Khalajzadeh et al. (2022), Obie et al. (2022), Ebrahimi and Mahmoud (2022), Anam and Yeasin (2013), Yang and Liang (2015), Nayebi et al. (2018), Dalpiaz and Parente (2019), Wang et al. (2021), Maalej et al. (2016), Carreño and Winbladh (2013), Hu et al. (2019a), Malik et al. (2020), Jha and Mahmoud (2019), Noei et al. (2019), Jiang et al. (2014), Bhandari et al. (2013), Malgaonkar et al. (2022), Besmer et al. (2020), Bhandari et al. (2013), Zhang et al. (2019), Assi et al. (2021), Jha and Mahmoud (2018), Higashi et al. (2018), Mathews et al. (2021), Hatamian et al. (2019), Tao et al. (2020)
	Uncommon words removal	Fu et al. (2013), Rahman et al. (2017), Yao et al. (2017), Park et al. (2015), McIlroy et al. (2016), Jha and Mahmoud (2018)
	Removing punctuation and special characters	Hoon et al. (2012), Martin et al. (2015), AlOmar et al. (2021), Panthum and Senivongse (2021), Zhou et al. (2021), Eiband et al. (2019), Morales-Ramirez et al. (2017), Khalajzadeh et al. (2022), Obie et al. (2022), Bhandari et al. (2013), McIlroy et al. (2016), Bhandari et al. (2013), Yang and Liang (2015), Wang et al. (2021), Carreño and Winbladh (2013), Hu et al. (2019a), Malgaonkar et al. (2022)
Informal language correction	Correcting typos, acronyms, and abbreviations	Haggag (2022), Phong et al. (2015), Gao et al. (2018b, 2022b), Wei et al. (2017), Zhang et al. (2021), Gao et al. (2021), Grano et al. (2018), Scherr et al. (2017), Yu et al. (2022), Khalajzadeh et al. (2022), Noei et al. (2019), Shah et al. (2019), Tao et al. (2020)
Noise reduction	Lowercasing	Martin et al. (2015), Man et al. (2016), Gao et al. (2015, 2022b), AlOmar et al. (2021), Deocadez et al. (2017), Zhou et al. (2021), Zhang et al. (2021), Gao et al. (2019), Farooq et al. (2020), Gao et al. (2021), Fereidouni et al. (2022), Wang et al. (2022a), Khalajzadeh et al. (2022), Obie et al. (2022), Alshayban et al. (2020), Carreño and Winbladh (2013), Hu et al. (2019a), Jha and Mahmoud (2019), Malgaonkar et al. (2022)
	lemmatization in default or customized way	Martin et al. (2015), Guzman et al. (2014), Lu and Liang (2017), Man et al. (2016), Gao et al. (2015, 2022b), AlOmar et al. (2021), Zhou et al. (2021), Zhang et al. (2021), Gao et al. (2019, 2021), Panthum and Senivongse (2021), Rahman et al. (2017), Zhu et al. (2015), Dhinakaran et al. (2018), Scherr et al. (2017), Wang et al. (201b8), Morales-Ramirez et al. (2017), Eiband et al. (2019), Gao et al. (2022a), Wang et al. (2022a), Ebrahimi and Mahmoud (2022), Nayebi et al. (2018), Dalpiaz and Parente (2019), Wang et al. (2021), Maalej et al. (2016), Malik et al. (2020), Malgaonkar et al. (2022), Zhang et al. (2019), Assi et al. (2021), Tao et al. (2020)
	Stemming in default or customized way	Alshangiti et al. (2022), Haggag (2022), Phong et al. (2015), Lu and Liang (2017), Panthum and Senivongse (2021), Palomba et al. (2015), Wei et al. (2017), Panichella et al. (2015), Ciurumelea et al. (2017), Zhang et al. (2021), Yadav et al. (2020), Wang et al. (201b8), Guzman et al. (2015b), Vu et al. (2016), Deocadez et al. (2017), Khalajzadeh et al. (2022), McIlroy et al. (2016), Yang and Liang (2015), Alshayban et al. (2020), Maalej et al. (2016), Hu et al. (2019a), Jha and Mahmoud (2019), Noei et al. (2019), Besmer et al. (2020), Jha and Mahmoud (2018), Mathews et al. (2021), Hatamian et al. (2019)
	Unifying specific types of text	Gao et al. (2019), Wang et al. (2022a), Farooq et al. (2020), Fereidouni et al. (2022), Gao et al. (2018b, 2022b), Oehri and Guzman (2020)
Review-level filtering	Filtering out inconsistent reviews	Fu et al. (2013), Noei et al. (2021)
	Filtering out uninformative reviews	Chen et al. (2014), Noei et al. (2021), Man et al. (2016), Wei et al. (2017), Palomba et al. (2015), Haering et al. (2021), Zhou et al. (2021), Zhang et al. (2021), Hu et al. (2021), Tushev et al. (2022), Hu et al. (2022), Wang et al. (2021), Yu et al. (2022), Alshangiti et al. (2022), van Vliet et al. (2020), Noei et al. (2019), Malgaonkar et al. (2022), Assi et al. (2021)
	Filtering out/Filtering repeated reviews	Martin et al. (2015), Panthum and Senivongse (2021), Hu et al. (2019c,b)
Other review preprocessing practices	Singularization, resolving negations, resolving synonyms, and coreference resolution	Haggag (2022), Jiang et al. (2014), Palomba et al. (2017), Noei et al. (2018), Scalabrino et al. (2019), Carreño and Winbladh (2013), Noei et al. (2019), S¨anger et al. (2017)

2014; Guzman et al., 2015a), and stemming (Chen et al., 2014; Noei et al., 2021; Di Sorbo et al., 2016). Lemmatization and stemming aim to reduce a word to its simplest form, thus reducing redundant information. Lemmatization can transform words more accurately but at a slower speed (Phong et al., 2015; Maalej and Nabil, 2015). Besides, lemmatization can be combined with PoS tags to get better results (Man et al., 2016). On the contrary, stemming transforms words faster but usually leads to the over-stemming problem (e.g., "adding" to "ad") (Man et al., 2016). Phong et al. (2015) trained a tri-gram language model to fix the over-stemming problem. Apart from the standard approaches, researchers customized lemmatization and stemming by focusing on nouns and verbs to make them more effective (Man et al., 2016; Gao et al., 2019). Moreover, some researchers reduce noise by unifying specific types of text, replacing rich expressions with limited tokens. Common practices include replacing app names and user names with "<app>" and "<user>" (Gao et al., 2019, 2021; Wang et al., 2022a), greetings and signatures with "<salutation>" and "<signature>" (Farooq et al., 2020; Fereidouni et al., 2022), email

address with "<email>" (Gao et al., 2019, 2021; Farooq et al., 2020; Fereidouni et al., 2022), URL with "<url>" (Gao et al., 2021; Farooq et al., 2020; Fereidouni et al., 2022), numbers with "<number>" (Farooq et al., 2020; Fereidouni et al., 2022), digits with "<digit>" (Gao et al., 2018b, 2022b, 2019, 2021), support account identifiers with the software name (Oehri and Guzman, 2020).

Review level filtering. Review level filtering includes filtering out inconsistent reviews in terms of sentiment, uninformative reviews, unwanted reviews, and filtering wanted reviews. Fu et al. (2013) are the first to filter out inconsistent user reviews by comparing star ratings with sentiment scores calculated from a regression model. The classification method is commonly adopted to filter out uninformative reviews and there are some wildly used tools, e.g., AR-Miner (Chen et al., 2014; Noei et al., 2021; Palomba et al., 2015; Noei et al., 2018; Malgaonkar et al., 2022; Assi et al., 2021), ARdoc (Palomba et al., 2017; Zhou et al., 2021), Caspar (Guo and Singh, 2020; Tushev et al., 2022). To filter out unwanted reviews (i.e., the feature request user reviews) from the identified functionality-relevant reviews, Wang et al. (2020) combined sentiment analysis with tense detection. As for retrieving wanted reviews, SURF (Di Sorbo et al., 2016) is a popular tool since it classifies reviews from two perspectives, i.e., user intentions and review topics, and previous studies (Gao et al., 2018a; Wei et al., 2017; Zhang et al., 2021) adopted SURF. Besides, Kong et al. (2015) adopted sparse SVM to obtain security-related reviews, and Hu et al. (2021) used BTM to filter reviews concerning undesired behaviors. Lastly, duplicated/similar reviews are suggested to be removed since they are considered fraudulent reviews provided to promote app search rank (Martin et al., 2015; Panthum and Senivongse, 2021; Singh et al., 2022). In particular, Di Sorbo et al. (2016) computed the duplicate tokens rate to achieve this goal.

Other review preprocessing practices. Other review preprocessing practices include singularization (Palomba et al., 2017), resolving negations (Noei et al., 2018; Scalabrino et al., 2019), resolving synonyms (Noei et al., 2018; Scalabrino et al., 2019), and coreference resolution (Noei et al., 2021).

5.4.2. Application of textual content: Feature extraction.

We present the handling and application of review textual content from four perspectives following a natural language processing (NLP) handbook (Indurkhya and Damerau, 2010), i.e., tokenization, lexical analysis, syntactic analysis, and semantic analysis.

Tokenization. There are generally two practices to tokenize the review text. The basic one is to split the review text into words (i.e., word tokenization) (Fu et al., 2013; Chen et al., 2014; Johann et al., 2017). Apart from being used to analyze the vocabulary in app reviews (Hoon et al., 2012), this practice can generate a bag of words that are commonly used as features for subsequent analysis (Maalej and Nabil, 2015; Lu and Liang, 2017; Panthum and Senivongse, 2021). The second is to split a raw review into sentences, which is common in review classification or clustering (Chen et al., 2014; Noei et al., 2021; Di Sorbo et al., 2021b; Kurtanović and Maalej, 2017; Mekala et al., 2021; Gu and Kim, 2015; Johann et al., 2017; Di Sorbo et al., 2016). This practice is popular these years since a review contains multiple sentences that may express different intentions (Pagano and Maalej, 2013). Moreover, each review can be further split into subsentences to get a more fine-grained analysis target (Sun et al., 2021; Zhou et al., 2021). These two treatments are used for different purposes and can be used in combination. For example, to classify the Review Sentence, researchers can first conduct sentence tokenization to obtain the target of the analysis, and then conduct word tokenization to obtain a bag of words as the input of the classifiers.

Lexical analysis. At the lexical level, app review analysis involves the extraction of specific words and phrases. In this study, we introduce various lexical features, including Bag-of-Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF), verbs, nouns, adjectives, n-grams, keywords, and collocations.

- **BoW and TF-IDF:** BoW (Cen et al., 2015; Greenheld et al., 2018; Lu and Liang, 2017; Panthum and Senivongse, 2021; Panichella et al., 2015; Scherr et al., 2017; Deocadez et al., 2017; Maalej et al., 2016; Mathews et al., 2021; Jha and Mahmoud, 2019, 2018; Dhinakaran et al., 2018; Zhang et al., 2020a) and TF-IDF (Lu and Liang, 2017; Panthum and Senivongse, 2021; Ciurumelea et al., 2017; Yadav et al., 2020; Wang et al., 201b8; Guzman et al., 2015b; Wei et al., 2017; Gao et al., 2021, 2022a; Khalajzadeh et al., 2022; McIlroy et al., 2016; Yang and Liang, 2015; Maalej et al., 2016; Jha and Mahmoud, 2019; Malgaonkar et al., 2022; Jha and Mahmoud, 2018; Mathews et al., 2021; Hatamian et al., 2019) are two common techniques used in NLP for text representation and feature extraction. BoW simply leverages the frequency of a term in a document while TF-IDF also takes the importance of a word into consideration. Besides, Jha and Mahmoud (2018) introduced the application of Bag-of-Frames (BoF), which can derive lower-level representation and thus have good generalization ability.
- **Verb, noun, and adjective:** Informative words and phrases are frequently extracted as lexical features or keywords. Verbs, nouns, adjectives, and their phrases are considered related to app features (Zhang et al., 2019), and they can be extracted by using Part of Speech (POS) tagging. These extracted words can be further used to form collocations by applying the collocation finding algorithm, and resulting collocations are the phrases representing app features (Hadi and Fard, 2020; Guzman and Maalej, 2014; Guzman et al., 2015a, 2014).
- **N-gram:** N-grams are typical lexical features, and we present three types of n-grams used in app reviews. The first and most popular one is the word n-grams. Word n-gram features for a sentence encompass all n consecutive tokens within that sentence. For example, Shah et al. (2019) extracted 1 to 3 word n-grams for review classification. The second one is character n-grams. Character n-gram features for a sentence encompass all n consecutive letters within the tokens of that sentence. For example, Gu and Kim (2015) extracted 2 to 4 character n-grams for review classification. The third is the POS n-grams, which are represented as all n consecutive POS tags for a target sentence. For example, Kurtanović and Maalej (Kurtanović and Maalej, 2018) extracted 1 to 3 POS n-grams for review classification.
- **Keyword:** Previous studies commonly exploited predefined keywords to identify specific types of app reviews. For example, Chen et al. (2021) identified UI-related reviews in this way and then manually categorized these reviews. Besides, predefined keywords can be utilized to construct app review datasets, especially for categories with rare app reviews (Obie et al., 2022; Nema et al., 2022).
- **Collocation:** Gao et al. (2022b) discovered that the information extracted solely by n-grams is insufficient. Therefore, they recommended incorporating collocations as a supplement. Generally, collocations are identified by using the frequent itemset mining algorithm (Luna et al., 2019). To ensure the interpretability of the extracted collocations, researchers can employ metrics such as Pointwise Mutual Information (PMI) (Hadi and Fard, 2020), and likelihood ratio (Guzman and Maalej, 2014).

Syntactic analysis. At the syntactic level, app review analysis involves the parsing tree.

- **Parsing tree:** A parsing tree, also known as a parse tree or syntax tree, is a graphical representation that shows the hierarchical structure and syntactic relationships within a sentence based on a specific grammar or parsing algorithm. It breaks down a sentence into its constituent parts, such as phrases and sub-phrases, and represents their relationships. Gu and Kim (2015) identified tree nodes and concatenated the POS tags of the first five nodes as the flat text feature for a parsing tree. Moreover, Kurtanović and

Table 17
Application: Feature extraction from app review content.

Types	Techniques/Features	Papers/References
Tokenization	Word tokenization	Hoon et al. (2012), Dhinakaran et al. (2018), Fu et al. (2013), Chen et al. (2014), AlOmar et al. (2021), de Araújo and Marcacini (2021), Man et al. (2016), Wei et al. (2017), Uddin et al. (2020b), Zhang et al. (2021), Hu et al. (2021), Wu et al. (2021), Johann et al. (2017), Morales-Ramirez et al. (2017), Srisopha et al. (2020a), Eiband et al. (2019), Obie et al. (2022), Harkous et al. (2022), Alshayban et al. (2020), Dalpiaz and Parente (2019), Carreño and Winbladh (2013), Jha and Mahmoud (2019), S¨anger et al. (2017), Besmer et al. (2020), Assi et al. (2021), Higashi et al. (2018), Hatamian et al. (2019), Tao et al. (2020)
	Sentence tokenization	Mekala et al. (2021), Iacob et al. (2013), Chen et al. (2014), Lu and Liang (2017), Groen et al. (2017), de Araújo and Marcacini (2021), Panthum and Senivongse (2021), Chen et al. (2021), Zhou et al. (2021), Panichella et al. (2015), Wu et al. (2021), Man et al. (2016), Sun et al. (2021), Noei et al. (2021), Huebner et al. (2020), Di Sorbo et al. (2021b), Keertipati et al. (2016), Iacob and Harrison (2013), van Vliet et al. (2020), Yu et al. (2022), Zhang et al. (2020a), Wang et al. (2022a), Dalpiaz and Parente (2019), Carreño and Winbladh (2013), Di Sorbo et al. (2021a), Tao et al. (2020)
Lexical analysis	BoW	Cen et al. (2015), Greenheld et al. (2018), Lu and Liang (2017), Panthum and Senivongse (2021), Panichella et al. (2015), Scherr et al. (2017), Deocadez et al. (2017), Maalej et al. (2016), Mathews et al. (2021), Jha and Mahmoud (2019, 2018), Dhinakaran et al. (2018), Zhang et al. (2020a)
	TF-IDF	Alshangiti et al. (2022), Lu and Liang (2017), Panthum and Senivongse (2021), Ciurumelea et al. (2017), Yadav et al. (2020), Wang et al. (201b8), Guzman et al. (2015b), Wei et al. (2017), Gao et al. (2021, 2022a), Khalajzadeh et al. (2022), McIlroy et al. (2016), Yang and Liang (2015), Bhandari et al. (2013), Maalej et al. (2016), Jha and Mahmoud (2019), Yu et al. (2022), Zhang et al. (2020a), Malgaonkar et al. (2022), Jha and Mahmoud (2018), Mathews et al. (2021), Hatamian et al. (2019)
	Verb, noun, and adjective	Phong et al. (2015), Guo and Singh (2020), Guzman et al. (2014), Hadi and Fard (2020), Guzman and Maalej (2014), Haering et al. (2021), Guzman et al. (2015a), Rahman et al. (2017), Zhang et al. (2021), Zhu et al. (2015), Su'a et al. (2017), Morales-Ramirez et al. (2017), Vu et al. (2016), Dalpiaz and Parente (2019), Malik et al. (2020), Jiang et al. (2014), Yu et al. (2022), Malgaonkar et al. (2022), Zhang et al. (2019), Tao et al. (2020)
	N-gram	Gu and Kim (2015), Kurtanović and Maalej (2018), AlOmar et al. (2021), Gao et al. (2015), Ciurumelea et al. (2017), Yadav et al. (2020), Scherr et al. (2017), Keertipati et al. (2016), Eiband et al. (2019), Yu et al. (2022), Nema et al. (2022), Khalajzadeh et al. (2022), Ebrahimi and Mahmoud (2022), Maalej et al. (2016), Kurtanović and Maalej (2018), Noei et al. (2019), Shah et al. (2019)
	Keyword	Chen et al. (2021), Haggag (2022), Hu et al. (2021), Gao et al. (2019), Khalid et al. (2016), Obie et al. (2022)
	Collocation	Guzman et al. (2014, 2015b), Guzman and Maalej (2014), Morales-Ramirez et al. (2017), Su'a et al. (2017), Dalpiaz and Parente (2019), Hadi and Fard (2020), Ebrahimi and Mahmoud (2022), Tao et al. (2020), Vu et al. (2016)
Syntactic analysis	Parsing tree	Gu and Kim (2015), Kurtanović and Maalej (2018,?), Shah et al. (2019), Yu et al. (2022)
Semantic analysis	Deontic	Licorish et al. (2017), Keertipati et al. (2016), Morales-Ramirez et al. (2017,?)
	Synonym	Di Sorbo et al. (2016), Martin et al. (2015), Wang et al. (2020), Yin and Pfahl (2018), Kong et al. (2015), Higashi et al. (2018)
	App feature/aspect	Guo and Singh (2020), Guzman et al. (2014, 2015b), Gao et al. (2015, 2022b), Ciurumelea et al. (2017), Morales-Ramirez et al. (2017), Wang et al. (2022a), Dalpiaz and Parente (2019), Malik et al. (2020), Huebner et al. (2018), Dąbrowski et al. (2019), S¨anger et al. (2017), Jiang et al. (2014), Zhang et al. (2019), Assi et al. (2021), Tao et al. (2020)
	Topic	Alshangiti et al. (2022), Huebner et al. (2020), Palomba et al. (2017), Zhang et al. (2021), Gao et al. (2022b), Yadav et al. (2020), Morales-Ramirez et al. (2017), Zhu et al. (2014), Gao et al. (2022a), Hassan et al. (2022), Tushev et al. (2022), Chen et al. (2015)
	Intention of the writer	Di Sorbo et al. (2021a), Gao et al. (2018a), Morales-Ramirez et al. (2017), Srisopha et al. (2020b,a)
	Semantic dependence graph	Gu and Kim (2015), Di Sorbo et al. (2021b), Panichella et al. (2015), Wang et al. (2020), Shah et al. (2019), Tao et al. (2020), Yu et al. (2022)
	Linguistic rule	Iacob and Harrison (2013), Di Sorbo et al. (2021b), Groen et al. (2017), Hu et al. (2021), Panichella et al. (2015), Tao et al. (2020)

Maalej (Kurtanović and Maalej, 2018) retrieved sentence syntax sub-trees count and sentence syntax tree height as features for review classification.

Semantic analysis. At the semantic level, app review analysis involves the deontic, synonyms, app features/aspects, topics, intentions of writers, the semantic dependence graph, and linguistic rules.

- **Deontic:** Maalej and Nabil (Maalej and Nabil, 2015) found that some stopwords can be relevant for the review classification. For instance, the terms "should" and "must" might indicate a feature request. Later, Licorish et al. (2017) introduced deontic terms to analyze app reviews. They noted that the modal verbs (e.g., must and should) can help to identify obligations, and negative modal verbs (e.g., does not, cannot, and must not) are related to prohibitions. Thus, they use the negative modal verbs to complement the default negative words. Keertipati et al. (2016) counted deontic terms for review prioritization.
- **Synonym:** Synonyms can help enrich domain knowledge. Previous studies (Di Sorbo et al., 2016; Wang et al., 2020) used WordNet to generate synonyms for all the extracted keywords. Kong et al. (2015) expanded the keywords list for security features by

considering synonyms, as well as antonyms and mismatch words. Besides, Guzman and Maalej (Guzman and Maalej, 2014) grouped collocations whose pairs of words are synonyms and misspellings by using Wordnet.

- **App feature/aspect:** App reviews provide abundant descriptions of app features and relevant user perception. Typical practices in prior studies involve extracting app features (Huebner et al., 2018; Tao et al., 2020; S[ii]anger et al., 2017), generating <*feature, opinion*> pairs (Malik et al., 2020), using app features as input to topic models (Guzman and Maalej, 2014), and prioritizing the app features (Keertipati et al., 2016).
- **Topic:** App reviews can be grouped into different topics by topic modeling. Typical practices in prior studies involve analyzing the distribution of topics (Huebner et al., 2020; Zhu et al., 2014), topic ranking (Gao et al., 2022b), using topics as input for review classification and prioritization (Yadav et al., 2020; Morales-Ramirez et al., 2017), and using topics to locate the source code of user-perceived app issues (Palomba et al., 2017; Zhang et al., 2021).
- **Intention of the writer:** App reviews reflect the intentions of users, which can be detected by tools, e.g., ARdoc (Srisopha et al., 2020b,a), SURF (Gao et al., 2018a). Apart from categorizing reviews based on user intention, intention can also serve as a classification feature (Srisopha et al., 2020b). Furthermore, Morales-Ramirez et al. (2017) utilized intention to quantify the severity perceived by the user.
- **Semantic dependence graph:** A Semantic Dependence Graph (SDG) is a representation that captures the semantic relationships between words or concepts in a sentence or text. SDG focuses on the meaning or semantic roles of words and their connections. Gu and Kim (2015) traversed SDG nodes and concatenated edges and POS tags as the flat text feature for an SDG. Besides, they introduced *trunk word* (i.e., the word at the root of a semantic dependence graph). In addition, previous studies extract keywords (Gu and Kim, 2015) or identify reviews containing predefined patterns (Di Sorbo et al., 2021b) from the SDG based on predefined linguistic templates. Moreover, previous studies extract the verb phrases and noun phrases by using the typed dependency relations (Wang et al., 2020; Yu et al., 2022).
- **Linguistic rule:** Linguistic rules can provide both semantic and syntactic information (Di Sorbo et al., 2021b) and it is widely adopted. For instance, Iacob and Harrison (2013) utilized linguistic rules to retrieve sentences related to feature requests. In addition, previous studies (Tao et al., 2020; Guo and Singh, 2020) extracted <*aspect, opinion*> pairs via predefined linguistic rules.

5.4.3. Take-home messages

Table 17 outlines the prevalent textual features and techniques categorized into four levels: Tokenization, Lexical analysis, Syntactic analysis, and Semantic analysis. These levels encompass distinct information and can complement each other, making them commonly employed in combination for app review analysis, such as review classification.

In machine learning methods, word tokenization serves as the foundational step, followed by the calculation of Bag of Words (BoW) and TF-IDF. Then, verbs, nouns, and adjectives can be extracted by PoS tagging, and n-grams, considered richer in information compared to BoW, can be generated. Lexical features like BoW, TF-IDF, verbs, nouns, and adjectives often act as input features for classification and can be employed collectively. N-grams and collocation are instrumental in app feature extraction, while keywords facilitate classification through keyword matching.

Parsing tree analysis typically involves acquiring PoS tags, which serve as input features. In terms of Semantic analysis, being a higher-level concept, these features often serve as the output/results of app review analysis. For example, developers and researchers adopted various ways to extract app features, including topic modeling, pattern-based methods, and large language models.

5.5. Dynamic nature

The dynamic nature of app reviews is reflected in both macro and micro perspectives. At the macro level, app reviews are often prompted by app updates and are written for specific app versions (Pagano and Maalej, 2013; Fu et al., 2013). On the micro level, users update their reviews for different app releases (Hoon et al., 2016) or after communicating with the app team (Hassan et al., 2018). As shown in Table 18, we present the handling, applications, and tips for dynamic app review submissions.

5.5.1. Handling dynamic nature

App reviews are triggered by app updates and are written for specific app versions. Previous studies (Pagano and Maalej, 2013; Fu et al., 2013) observed that spikes in reviews are closely related to new releases of an app. To tackle such characteristics, researchers can segregate app reviews by releases (Guzman and Paredes Rojas, 2019; Fu et al., 2013; Hadi and Fard, 2020; Noei et al., 2021; Gao et al., 2015, 2018b; Noei et al., 2018; Uddin et al., 2020b; Gao et al., 2022b; Nguyen et al., 2019). Besides, various features are constructed, e.g., review burstiness, review freshness, TimeOfDay, DayOfWeek, IsWeekEnd, and TimeAfterRelease (Li et al., 2017; Srisopha et al., 2020a; Gao et al., 2022a).

5.5.2. Application of dynamic nature

Applications of the dynamic nature of app reviews include generating time series data, conducting trend analysis, comparison, and exploiting submission dates.

Generating time series data. Numerical features such as app rankings, ratings, review sentiment, and review topics can be used to generate time series data (Zhu et al., 2014; Ouyang et al., 2018).

Trend analysis. Trend analysis provides information on how app reviews change over time. For instance, Fu et al. (2013) visualized the lifespan of an app by plotting time series of reviews. Gao et al. (2022b) visualized detailed issues identified from user reviews along with app versions.

Comparison. Li et al. (2017) compared the standard deviation of review submission time for targeted and random apps, as well as the burstiness before and after promotion.

Exploiting submission date. Submission date can quantitatively indicate the dynamics of app reviews and thus can be used to prioritize reviews (Wei et al., 2017; Hassan et al., 2018).

5.5.3. Take-home messages

Choosing approaches that can model dynamic information. The spikes of app reviews are triggered by app updates. To dynamically capture what users write about in the app reviews, researchers can apply dynamic topic models (Gao et al., 2015) and online learning (Hadi and Fard, 2020; Gao et al., 2018b, 2022b) for analysis rather than the traditional topic models (e.g., LDA, PLSA).

Jointly mining artifacts that contain dynamic information about apps. Previous studies (Pagano and Maalej, 2013; Fu et al., 2013) observed that spikes in reviews are closely related to new releases of an app. Since apps keep evolving over the versions, researchers can jointly employ artifacts that reflect app changes (e.g., release notes (Noei et al., 2021, 2018; Scalabrino et al., 2019; Hassan et al., 2020; Uddin et al., 2020b), app changelogs (Gao et al., 2015, 2018b, 2022b)) to mine app reviews.

5.6. Sentiment

As shown in Tables 19 and 20, we present the handling and applications for app review sentiment.

Table 18
Handling, applications, and take-home messages for dynamic nature of app review.

Types	Techniques	Papers/References
Handling dynamic submissions	Segregating app reviews by releases	Guzman and Paredes Rojas (2019), Fu et al. (2013), Hadi and Fard (2020), Noei et al. (2021), Palomba et al. (2015), Gao et al. (2015, 2018b), Noei et al. (2018), Uddin et al. (2020b), Gao et al. (2022b), Nguyen et al. (2019), Di Sorbo et al. (2021a)
Handling: Constructing features	Review burstiness	Li et al. (2017)
	TimeOfDay, DayOfWeek, IsWeekEnd, and TimeAfterRelease	Srisopha et al. (2020a)
	Review freshness	Gao et al. (2022a)
Applications: Generating time series data	App rankings, ratings, review sentiment, review topics	Zhu et al. (2014), Ouyang et al. (2018), Zhang et al. (2020b,a), Singh et al. (2022)
Applications: Trend analysis	Visualizing user reviews (amount) along with app versions/time series	Fu et al. (2013), Phong et al. (2015), Guzman et al. (2014), Scherr et al. (2017), Gao et al. (2022b), Martens and Maalej (2019b)
	Deriving time series patterns	Chen et al. (2014), Gao et al. (2018a)
	Tracking issues reported in reviews overtime	Vu et al. (2016)
Applications: Comparison	Comparing different apps	Li et al. (2017)
	Comparing different treatments	Li et al. (2017)
Applications: Exploiting submission date	Classification	Srisopha et al. (2020a)
	Prioritization	Etaiwi et al. (2020), Wei et al. (2017), Hassan et al. (2018), Gao et al. (2022a)
Take-home messages	Choosing approaches that can model dynamic information	Gao et al. (2015), Hadi and Fard (2020), Gao et al. (2018b, 2022b)
	Jointly mining artifacts that contain dynamic information about apps	Noei et al. (2021, 2018), Scalabrino et al. (2019), Hassan et al. (2020), Uddin et al. (2020b), Gao et al. (2015, 2018b, 2022b)

5.6.1. Handling sentiment

Handling sentiment includes review text filtering, sentiment word extraction, deriving sentiment scores, deriving sentiment polarities, constructing features, generating aspect-sentiment pairs, and analyzing emoticons.

Review text filtering. Review text filtering generally includes text extraction and text removal. Sentiment provides researchers with a new dimension of thinking and processing in terms of text filtering. For instance, Sun et al. (2021) extracted sentences likely to express sentiments via pattern matching. Fu et al. (2013) filtered out non-negative words. Wang et al. (2020) filtered out user reviews describing non-existed features from the identified functionality-relevant reviews by combining sentiment analysis with tense detection. Noei et al. (2021) filtered out inconsistent user reviews by comparing star ratings with sentiment scores. Besides, selecting negative reviews is common practice (Fu et al., 2013; Eiband et al., 2019; Ikram and Kaafar, 2017; Su'a et al., 2017; Tao et al., 2020).

Sentiment words extraction. There are studies conducting sentiment word extraction. The most convenient method is to exploit available dictionaries and lists (Licorish et al., 2017). Besides, Jiang et al. (2014) designed linguistic rules to extract sentiment words and generate aspect-sentiment pairs. Fu et al. (2013) used negative words as input for topic modeling.

Deriving sentiment score. Sentiment is not provided explicitly but is implied in the textual reviews. To quantify sentiment, researchers need to derive the sentiment score. In previous studies, researchers either adopted available sentiment analysis tools (Maalej and Nabil, 2015; Oehri and Guzman, 2020; Noei et al., 2021; Guzman and Maalej, 2014; Guzman et al., 2015a, 2014; Gu and Kim, 2015; Luiz et al., 2018) or designed the sentiment score themselves (Gao et al., 2015, 2018b; Panichella et al., 2015). The commonly used sentiment analysis tools include SentiStrength (Thelwall et al., 2010), Deeply Moving,[24] and SACI (Rocha et al., 2015). Besides, the customized approaches to derive sentiment scores vary a lot, including designing metrics based on rating and length (Gao et al., 2015, 2018b), and classifying review sentiment with machine learning classifiers (Panichella et al., 2015).

Deriving sentiment polarity. Deriving sentiment polarity is a more coarse-grained practice compared with deriving sentiment score. It is commonly based on deriving sentiment score. For instance, Gao et al. (2022a) computed the total positive score minus the total negative score of the review text while Wang et al. (2022a) compared a 1.5 negative score with a positive score. Besides, Anam and Yeasin (2013) adopted Naive Bayes classifier and Jiang et al. (2014) exploited propagation algorithm-based polarity assignment method.

Constructing feature. Various sentiment features are constructed, e.g., sentiment-word-num, rating-extremity, and sentiment similarity (Gao et al., 2022a; Ouyang et al., 2018), discrepancies (between the rating and the review's sentiment) (Kuehnhausen and Frost, 2013), ratio of negative, and positive sentiment words in the review (Guzman et al., 2015b; Srisopha et al., 2020b,a), proportion of negative, positive, and neutral reviews (Noei et al., 2018; Rahman et al., 2017).

Generating aspect-sentiment pair. Aspect-sentiment analysis falls under the umbrella of opinion mining, wherein the objective is to discern the sentiment linked to particular aspects or features of a product or topic. This technique is widely employed in app review analysis (Gu and Kim, 2015; Vu et al., 2016; Dąbrowski et al., 2019; Jiang et al., 2014). In particular, Gu and Kim (2015) employed Deeply Moving to assign sentiment polarities for target aspects and generated aspect-sentiment pairs. Similarly, Tao et al. (2020) obtained <misbehavior-aspect-opinion> by analyzing SDG and extracting corresponding words based on predefined semantic patterns.

Analyzing emoticon. Scherr et al. (2017) classified emoticons into different sentiment polarities. Similarly, Srisopha et al. (2020b) derived fine-grained sentiment scores for emojis.

5.6.2. Application of sentiment

Applications of sentiment include performing statistical analysis, using sentiment as an input feature, model building, and comparison.

Performing statistical analysis. The applications of using review sentiment score for statistical analysis can also be divided into two categories, descriptive statistical analysis, and inferential statistical analysis. Guzman et al. (2014) visualized sentiment distribution. Besides, Zhang et al. (2020b) conducted a correlation analysis between review sentiment and the number of app downloads. Hassan et al. (2018) used sentiment score as one of the explanatory variables of the

[24] Deeply Moving: https://nlp.stanford.edu/sentiment/

Table 19
Handling for app review sentiment.

Types	Techniques	Papers/References
Review text filtering	Selecting negative reviews	Fu et al. (2013), Wang et al. (2020), Eiband et al. (2019), Ikram and Kaafar (2017), Su'a et al. (2017), Tao et al. (2020)
Sentiment words extraction	Predefined linguistic rules	Jiang et al. (2014)
	Negative weights of words from a linear regression	Fu et al. (2013)
	LIWC dictionary and lists in previous studies	Licorish et al. (2017)
Deriving sentiment score	Manual annotation	Besmer et al. (2020)
	SentiStrength	Oehri and Guzman (2020), Noei et al. (2021), Guzman and Maalej (2014), Wang et al. (2020), Guzman et al. (2015a, 2014), Maalej et al. (2016)
	SentiStrength-SE	Noei et al. (2018), Wang et al. (2022a)
	Deeply Moving	Gu and Kim (2015)
	Google Cloud Natural Language API	Eiband et al. (2019)
	SACI	Luiz et al. (2018)
	Stanford CoreNLP	Nayebi et al. (2018)
	VADER	Huebner et al. (2020), Nayebi et al. (2018), Jha and Mahmoud (2019), Huebner et al. (2018), Tao et al. (2020), Hatamian et al. (2019), Malgaonkar et al. (2022)
	SnowNLP	Ouyang et al. (2018), Zhang et al. (2020b,a)
	Lexicon-based method with AFINN-165 dataset	Besmer et al. (2020)
	Customized sentiment scores	Gao et al. (2015, 2018b), Panichella et al. (2015), Kuehnhausen and Frost (2013), Gao et al. (2022a)
Deriving sentiment polarity	Manual annotation	Guzman and Paredes Rojas (2019), Yu et al. (2022)
	Computing the total positive score minus the total negative score of the review text	Gao et al. (2022a)
	Comparing 1.5 negative score with positive score	Wang et al. (2022a)
	Naive Bayes classifier	Anam and Yeasin (2013)
	Propagation algorithm-based polarity assignment method	Jiang et al. (2014)
Constructing feature	Sentiment-word-num, and rating-extremity	Gao et al. (2022a)
	Sentiment similarity	Gao et al. (2022a), Ouyang et al. (2018)
	Discrepancies (between the rating and the review's sentiment)	Kuehnhausen and Frost (2013)
	Ratio of negative, and positive sentiment words in the review	Guzman et al. (2015b), Srisopha et al. (2020b,a)
	Proportion of negative, positive, and neutral reviews	Noei et al. (2018), Rahman et al. (2017)
Generating aspect-sentiment pair	Assigning sentiment to key phrases (e.g., app aspects, app features)	Gu and Kim (2015), Vu et al. (2016), Dąbrowski et al. (2019), Jiang et al. (2014), Tao et al. (2020)
	Analyzing the co-occurrence of negative words and app features	Keertipati et al. (2016)
	Analyzing the prevalence of dominant emotions for each review theme	Harkous et al. (2022)
Analyzing emoticon	Classifying emoticons	Scherr et al. (2017)
	Deriving emoji sentiment scores	Srisopha et al. (2020b)

logistic regression model to prioritize the reviews to read and respond to.

Utilizing sentiment as an input feature. Sentiment score can be used as a feature for clustering (Uddin et al., 2020a), classification (Maalej and Nabil, 2015; Kurtanović and Maalej, 2017; Hoon et al., 2013; Panthum and Senivongse, 2021), prioritization (Gao et al., 2015, 2018b; Yin and Pfahl, 2018; Uddin et al., 2020b), and prediction models (Zhang et al., 2020b,a). Moreover, previous studies (Gao et al., 2019) incorporated review sentiment score into the seq2seq models (e.g., RNN, LSTM) to automatically generate responses for user reviews.

Model building. Sentiment can be jointly mined with review content in topic modeling, e.g., the Biterm-based Sentiment-Topic Modeling (BST) (Gao et al., 2022a), the Aspect and Sentiment Unification Model (ASUM) (Carreño and Winbladh, 2013). Besides, Nema et al. (2022) fine-tuned the BERT model on a sentiment dataset to leverage the sentiment polarities in app reviews, thereby enhancing the classifier's performance.

Comparison. Sentiment is a common metric for comparison. Nayebi et al. (2018) compared sentiment polarities between reviews and

Table 20
Applications for app review sentiment.

Types	Techniques	Papers/References
Performing statistical analysis	Visualizing sentiment distribution (with rating)	Guzman et al. (2014), Huebner et al. (2018), Besmer et al. (2020)
	Correlation analysis	Zhang et al. (2020b)
	Logistic regression model	Hassan et al. (2018)
Utilizing sentiment as an input feature	Clustering feature	Uddin et al. (2020a)
	Classification feature	Maalej and Nabil (2015), Kurtanović and Maalej (2017), Hoon et al. (2013), Rahman et al. (2017), Panthum and Senivongse (2021), Srisopha et al. (2020a), Wang et al. (2022a), Kurtanović and Maalej (2018)
	Prioritization feature	Gao et al. (2015, 2018b), Yin and Pfahl (2018), Uddin et al. (2020b), Keertipati et al. (2016), Vu et al. (2016), Gao et al. (2022a)
	Feature for prediction model	Zhang et al. (2020b,a)
	Feature for seq2seq model	Gao et al. (2019)
Model building	Jointly mining review content and sentiment in topic modeling	Gao et al. (2022a), Carreño and Winbladh (2013)
	Fine-tuning the pre-trained model on a sentiment dataset	Nema et al. (2022)
Comparison	Comparing sentiment polarities	Nayebi et al. (2018)
	Comparing sentiment scores	Besmer et al. (2020)

tweets. Besmer (Besmer et al., 2020) compared sentiment scores between privacy-related and non-privacy-related reviews

5.6.3. Take-home messages

Correlation with rating.

Hoon et al. (2012) presented the top words across all ratings and examined the distribution of a specific sentiment detection word set among the ratings. Their study provided empirical evidence supporting the association between app ratings and the words expressing sentiment. This finding helps to determine whether the review text is consistent with the rating, thus filtering inconsistent reviews (Fu et al., 2013). On the contrary, Gu and Kim (2015) used rating to adjust the sentiment scores of extracted app aspects.

Answer to RQ 3: Numeric characteristics (i.e., *review rating*, *review length*, and *review sentiment*) are commonly used to construct features, select app reviews, perform statistical analysis, and act as features for different tasks. Regarding *domain knowledge*, previous studies jointly mined app reviews with SE and non-SE information. For the *textual content*, previous studies normally do preprocessing and then extract features from various linguistic levels. For the *dynamic nature* of app reviews, the most common handling is to segregate app reviews by releases and use submission date to generate time series.

6. Publicly available tools

In this section, we answer **RQ 4: How available and what are the limitations of tools proposed in previous studies?** To foster further research in the field and advance existing tools, we conduct a manual inspection and summarization of open-source tools presented in all 167 papers, ensuring that the provided links remain accessible. This compilation serves to inform subsequent researchers about available open-source tools, facilitating their selection for research and replication based on the specific components outlined in our summaries. Additionally, we discuss the limitations associated with these tools. In the end, we discuss open-source tools involving developers in the evaluation process and present an industrial view of user feedback and automatic tools.

We compile a list of publicly accessible tools identified in 167 papers, as shown in Table 21. In addition to extracting replication package links from each paper, we conduct searches using tool names to uncover potential open-source links. Furthermore, authors may provide comprehensive lists of tools they have designed on their homepages. In total, we have identified 37 open-source tools, with 34 originating from valid links in papers, one tool (i.e., RE-SWOT (Dalpiaz and Parente, 2019)) discovered through tool name searches, and two tools (i.e., ARdoc (Panichella et al., 2015), CrossMiner (Man et al., 2016)) identified through inspection of authors' homepages.

Classification: There are nine papers providing open-source tools to automatically classify app reviews, which can help developers save a lot of effort in finding desired types of user feedback. Before conducting classification, researchers need pre-defined taxonomies, either by manual analyses of small samples, or by referring to existing taxonomies. These tools can be used to classify the reviews based on different categorizing standards such as user intention (ARdoc (Panichella et al., 2015), URM (Di Sorbo et al., 2016)), user requirements (MARC 3.0 (Jha and Mahmoud, 2019)), software evolution (NEON (Di Sorbo et al., 2021b)), undesired behaviors (CHAMP (Hu et al., 2021), Dolos (Rahman et al., 2021), Obie et al. (2022)), helpful and useless (Mekala et al. (2021)), human-centric issues (Khalajzadeh et al. (2022)).

Feature Extraction: There are four papers (SAFE (Johann et al., 2017), RE-SWOT (Dalpiaz and Parente, 2019), KEFE (Wu et al., 2021), FeatCompare (Assi et al., 2021)) providing open-source tools to automatically extract app features from app reviews. App features reflect the function of an app. Developers can obtain users' opinions on the app's functions from app reviews (Johann et al., 2017; Wu et al., 2021), and can also compare the functions of their own app with those of competing apps (Dalpiaz and Parente, 2019; Assi et al., 2021), thereby gaining users' favor in the app market. Three tools ((SAFE (Johann et al., 2017), RE-SWOT (Dalpiaz and Parente, 2019), KEFE (Wu et al., 2021)) adopt pattern matching-based methods to extract app features, which require pre-defined linguistic patterns. Typically, the identification of these linguistic patterns is a manual process, which is labor-intensive and tedious. Di Sorbo et al. (2021b) introduced NEON to identify recurring linguistic patterns automatically, but the patterns extracted automatically still necessitate manual revision, as only 35% of them are deemed useful. Additionally, FeatCompare (Assi et al., 2021) employed ABAE (He et al., 2017), an unsupervised neural attention model to extract app features. FeatCompare also introduced a distinction between global features (app features shared among all types of apps) and local features (specific features shared among similar apps), which can help better extract app features.

Matching App Reviews with Other Artifacts: There are three papers providing open-source tools (URR (Ciurumelea et al., 2017), CHA-

Table 21
Publicly available tools proposed in previous studies.

Tools	Functionalities	Components	Artifacts	Years	Venues
AR-miner (Chen et al., 2014)	Summarizing app reviews by visualizing the most informative reviews	Filtering + Grouping + Ranking + Visualization	App reviews	2014	ICSE
ARdoc (Panichella et al., 2015)	Classifying app reviews for software maintenance and evolution	Taxonomy for Software Maintenance and Evolution + Text Analysis + Natural Language Processing + Sentiment Analysis + Learning Classifiers	App reviews	2015	ICSME
CRISTAL (Palomba et al., 2015)	Tracing app reviews onto app changes for better evolution	Extracting Issues and Commits + Detecting Links (Linking Informative Reviews and Issues + Linking Informative Reviews and Commits + Linking Issues and Commits + Filtering Links)	App reviews, commit notes and issue reports	2015	ICSME
CrossMiner (Man et al., 2016)	Summarizing cross-platform app issues from app reviews	Clean Review Extraction + Keywords Generation (Training Model + Extracting Keywords + Ranking Reviews) + Visualization	App reviews	2016	ISSRE
SURF (Di Sorbo et al., 2016)	Summarizing app reviews by visualizing the most informative reviews and recommending software changes	Intention Classification + Topics Classification + Sentence Scoring and Extraction + Summary Generation	App reviews	2016	FSE
URM (Di Sorbo et al., 2016)	Classifying app reviews based on users' intentions and the review topics	Intention Classification + Topics Classification	App reviews	2016	FSE
CHANGEADVISOR (Palomba et al., 2017)	Localizing source code to be changed based on app reviews	User Feedback Classification + Input Preprocessing + User Feedback Clustering + Recommending Source Code Changes	App reviews, source code	2017	ICSE
SAFE (Johann et al., 2017)	Extracting app features via pre-defined text patterns	Identifying Patterns+ Automated Feature Extraction + Automated Feature Matching	App reviews, app descriptions	2017	RE
URR (Ciurumelea et al., 2017)	Classifying app reviews and localizing source code to be changed	Classifying Reviews + Linking Reviews with Source Code	App reviews, source code	2017	SANER
IDEA (Gao et al., 2018b)	Summarizing app reviews for detecting emerging app issues	Phrase Extraction + Emerging Topic Detection + Topic Interpretation + Visualization	App reviews	2018	ICSE
INFAR (Gao et al., 2018a)	Summarizing app reviews with trend analysis and three insights (i.e., salient topics, abnormal topics, and correlated topics)	Insight Extraction + Text Template Definition + User Review Prioritization	App reviews	2018	FSE
MARC 2.0 (Jha and Mahmoud, 2018)	Summarizing app reviews by leveraging frame semantics	App Review Classification + Review Summarization	App reviews	2018	EMSE
CLAP (Scalabrino et al., 2019)	Summarizing app reviews for app release planning	Categorizing User Reviews + Clustering Related Reviews + Prioritizing Review Clusters	App reviews	2019	TSE
MARC 3.0 (Jha and Mahmoud, 2019)	Identifying non-functional requirements from app reviews	Manual Classification + Automated Classification + Dictionary Based Classification	App reviews	2019	EMSE
NEON (Di Sorbo et al., 2021b)	Identifying recurrent language structures in text	Pattern Mining + Classification	App reviews/development emails/bug descriptions on GitHub	2019	TSE
RE-SWOT (Dalpiaz and Parente, 2019)	Generating and visualizing user requirements from app reviews with SWOT model	Identify Features and Transform Ratings + Calculate FPS per Feature + Generate RE-SWOT Matrix + Generate Requirements + Visualization	App reviews	2019	REFSQ
RRGen (Gao et al., 2019)	Automating app review response generation	RNN Encoder–Decoder Model + Attention Mechanism	App reviews, developer responses	2019	ASE
Caspar (Guo and Singh, 2020)	Summarizing user stories of problems from app reviews	Extracting Events + Synthesizing Event Pairs (Event encoding, Classification, Synthesis) + Inferring Events (USE+SVM, Bi-LSTM network, Clustering)	App reviews	2020	ICSE

(*continued on next page*)

Table 21 (*continued*).

Tools	Functionalities	Components	Artifacts	Years	Venues
Srisopha et al. Srisopha et al. (2020a)	Predicting developer responses for app reviews	Feature Selection (Rating + Length + Vote + Time + Style + Sentiment + Intention)	App reviews, developer responses	2020	ESEM
CHAMP (Hu et al., 2021)	Identifying app reviews revealing undesired behaviors against app market policies via semantic rules	Training Dataset Labelling (Topic Modeling and Topic Labelling) + Automated Semantic Rule Extraction (Representative Keywords Extraction + Semantic Rule Generation) + Behavior Identification	App reviews, app market policies	2021	ICSE
CoRe (Gao et al., 2021)	Automating app review response generation	Attentional Encoder–decoder Model + Pointer-generator Model	App reviews, app descriptions, developer responses	2021	TOSEM
Dolos (Rahman et al., 2021)	Attributing fraudulent behaviors to crowdsourcing site profiles by using app reviews	Fraud Worker Profile Collection + Fraud Component Detection + Component Attribution	App reviews, crowdsourcing websites	2021	TKDE
FeatCompare (Assi et al., 2021)	Comparing app features with competitive apps via app reviews	Data Preprocessor + Global–local Sensitive Feature Extractor + Rating Aggregator	App reviews	2021	EMSE
KEFE (Wu et al., 2021)	Identifying app features highly correlated to app ratings from app reviews	Feature Extraction + User Review Matching + Regression Analysis	App reviews, app descriptions	2021	ICSE
Mekala et al. Mekala et al. (2021)	Classifying app reviews as helpful or useless in review level and sentence level	Traditional Machine Learning Implementations/ Deep Learning Implementations	App reviews	2021	RE
Where2Change (Zhang et al., 2021)	Localizing source code to be changed based on app reviews and issue reports	Selecting Change Request-Related User Feedback + Clustering Change-Related User Feedback + Building a Link between Feedback Clusters and Issue Reports + Change Request Localization Using Enriched Cluster of User Feedback	App reviews, source code, issue reports	2021	TSE
Ebrahimi and Mahmoud (Ebrahimi and Mahmoud, 2022)	Summarizing privacy issues from app reviews	Privacy Term Extraction + Privacy Review Summarization	App reviews	2022	ASE
Hark (Harkous et al., 2022)	Summarizing privacy issues from app reviews	Privacy Feedback Classifier + Issue Generation + Theme Creation (Issue Grouping + Theme Title Creation) + Improving Navigability (Emotions + Feedback Quality)	App reviews	2022	SP
HMK-RVM (Alshangiti et al., 2022)	Summarizing app reviews by leveraging the hierarchical relationships between app review labels	Hierarchical User Review Classification + Multi-Kernel Relevance Vector Machines	App reviews	2022	FSE
Khalajzadeh et al. Khalajzadeh et al. (2022)	Classifying human-centric issues in app reviews and issue comments	Categories of End-User Human-Centric Issues + Automated Classification of End-User Human-Centric Issues (Machine Learning/Deep Learning)	App reviews, issue comments in GitHub	2022	TSE
Malgaonkar et al. Malgaonkar et al. (2022)	Prioritizing app reviews	Regression-based prioritization technique (Entropy + Frequency + TF-IDF + Sentiment analysis)	App reviews	2022	IST
MERIT (Gao et al., 2022b)	Summarizing app reviews for detecting emerging app issues	Data Preparation (Preprocessing and Filtering + Polarity Word Preparation) + AOBST (Biterm Sentiment-Topic Model + Adaptive Online Joint Sentiment-Topic Tracing) + Emerging Issue Detection (Emerging Topic Identification + Automatic Topic Interpretation +Interpreting Topics with Phrases/Sentences) + Interpreting Topics with Phrases	App reviews	2022	TSE
Obie et al. Obie et al. (2022)	Identifying app dishonest behaviors from app reviews	Automatic Classification of Honesty Violations + Categories of Honesty Violations	App reviews	2022	MSR
PPrior (Fereidouni et al., 2022)	Prioritizing app issues from app reviews	Self-Supervised Training + Contrastive Training + Radius Neighbor Classification	App reviews	2022	Big Data

(*continued on next page*)

Table 21 (*continued*).

Tools	Functionalities	Components	Artifacts	Years	Venues
SIRA (Wang et al., 2022a)	Summarizing app reviews for presenting problematic app features	Problematic Feature Extraction + Problematic Feature Clustering + Visualization	App reviews	2022	ICSE
SOLAR (Gao et al., 2022a)	Summarizing app reviews by prioritizing the most informative reviews	Review Helpfulness Prediction + Biterm-Based Sentiment-Topic Modeling + Multifactor Topic and Review Ranking	App reviews	2022	TR
Wang et al. Wang et al. (2022b)	Matching app reviews and bug reports	Noun and Verb Selection + Pretrained model	App reviews, bug reports	2022	QRS

NGEADVISOR (Palomba et al., 2017), Where2Change (Zhang et al., 2021)) to exploit app reviews to localize source code to be changed. Upon analyzing app reviews to understand users' voice, developers often find the need for adjustments. This involves modifying the source code to enhance the app's functions and services. Tools that utilize app reviews to pinpoint source code can be valuable in assisting developers with these modifications. Typically, this type of tool calculates the similarity between app reviews and source code text and matches the most similar ones. To bridge the gap between user-generated app reviews (expressed in natural language) and the technical language of source code created by developers, the process is initiated by clustering app reviews, with each class representing a distinct function. Following this, source code components such as class names and method names are extracted. Ultimately, the similarity between these source code components and the clusters of app reviews is computed. In addition, Where2Change (Zhang et al., 2021) improved app review clusters by integrating detailed information from historical issue reports, surpassing the performance of CHANGEADVISOR (Palomba et al., 2017). Besides, CRISTAL (Palomba et al., 2015) detected the links between app reviews and source code changes (i.e., issue reports, and commits) to aid release planning. Wang et al. (2022b) highlighted that matching app reviews with bug reports can assist app developers in promptly identifying new bugs based on user feedback.

Review Response: Automated response tools prove valuable for popular apps handling a substantial volume of daily reviews. These tools enable developers to respond promptly, enhancing user satisfaction and potentially boosting the app's star rating (McIlroy et al., 2017). Gao et al. provided two open-source tools (RRGen (Gao et al., 2019), CoRe (Gao et al., 2021)) to automatically generate developer responses for app reviews, both utilizing neural machine translation models. RRGen relies on app review and developer response pairs, while CoRe additionally incorporates app descriptions. Moreover, Srisopha et al. (2020a) explored the relationship between app review features and developer responses, contributing insights to the development of effective review response tools.

Summarization: The objective of review summarization is to provide app review information in a concise summary. Review summarization can be conducted from various perspectives, including app maintenance and evolution (SURF (Di Sorbo et al., 2016), MARC 2.0 (Jha and Mahmoud, 2018), CLAP (Scalabrino et al., 2019), SOLAR (Gao et al., 2022a), HMK-RVM (Alshangiti et al., 2022), Malgaonkar et al. (2022)), app issues (CrossMiner (Man et al., 2016), IDEA (Gao et al., 2018b), MERIT (Gao et al., 2022b), PPrior (Fereidouni et al., 2022), SIRA (Wang et al., 2022a)), user requirements (CLAP (Scalabrino et al., 2019)), trend (INFAR (Gao et al., 2018a), IDEA (Gao et al., 2018b), MERIT (Gao et al., 2022b)), user stories (Caspar (Guo and Singh, 2020)), security and privacy (Hark (Harkous et al., 2022), Ebrahimi and Mahmoud (Ebrahimi and Mahmoud, 2022)). Commonly, review summarization involves classification, clustering, feature extraction, ranking, and visualization to display the most informative reviews.

Limitations: We list the limitations from the view of review characteristics we summarized in this SLR. For the characteristic ***Textual Content***, in the process of mining app reviews, previous studies tend to mix user opinions and app aspects (Gu and Kim, 2015) or extract app features without distinguishing user requirements, potentially leading to confusion for developers. We suggest analyzing app reviews in a structural way. For the characteristic ***Dynamic Nature***, previous studies analyze app reviews on a static level; however, the dynamic nature of user feedback makes it more important to analyze app reviews across different time periods, e.g., different app versions. For the characteristic ***Sentiment***, previous studies in the realm of app reviews commonly utilize sentiment analysis tools with default settings. It has been verified that sentiment analysis tools need customization for SE data, as the default setting may decrease accuracy (Lin et al., 2018).

Involving developers in evaluation process: Among 37 open-source tools, 15 engage developers for evaluation. This includes two tools for classification (Di Sorbo et al., 2016; Khalajzadeh et al., 2022), two for feature extraction (Dalpiaz and Parente, 2019; Assi et al., 2021), three for source code localization (Ciurumelea et al., 2017; Palomba et al., 2017; Zhang et al., 2021), two for review response (Gao et al., 2019, 2021), and six for summarization (Chen et al., 2014; Di Sorbo et al., 2016; Gao et al., 2018a; Scalabrino et al., 2019; Gao et al., 2022a; Wang et al., 2022a). Within these, two papers use developers' annotations as ground truth (Chen et al., 2014; Zhang et al., 2021), one evaluates classification correctness with developers (Di Sorbo et al., 2016), three assess the quality of generated content (Di Sorbo et al., 2016; Gao et al., 2019, 2021). The metrics proposed to measure content quality include "content adequacy", "conciseness", "expressiveness", "grammatical fluency", "relevance", and "accuracy". Besides, eight papers gauge the usefulness of the tools with developers (Khalajzadeh et al., 2022; Dalpiaz and Parente, 2019; Assi et al., 2021; Ciurumelea et al., 2017; Palomba et al., 2017; Gao et al., 2018a; Scalabrino et al., 2019; Wang et al., 2022a), and one involves developers in manually evaluating the consistency between prioritized reviews and changelogs, as well as informativeness (Gao et al., 2022a).

Industrial view of user feedback and automatic tools: According to the survey of Van Oordt and Guzman (Van Oordt and Guzman, 2021), user feedback is vital for rapid app evolution due to its dynamic and fast-paced nature. Practitioners collect user feedback mainly to identify new requirements, prioritize features, and address bugs and common customer complaints. Among practitioners, 12 out of 16 obtain feedback from review platforms, while users often reach out to support, sales, or marketing departments. Feedback is obtained through multiple channels. Even though the feedback quality is low, review platforms receive the largest number of feedback. After filtering noise, app reviews provide valuable information for practitioners. While 71% of participants are aware of automated tools for user feedback analysis, 83% still prefer manual analysis due to the complex nature of the information. The absence of an easy-to-use, general-purpose tool for analyzing multi-channel feedback is noted, with current automated tools focusing on classification, clustering, and functional-level sentiment analysis.

Answer to RQ 4: Among 167 papers, a total of 48 papers provide replication links, with 12 links being invalid and 36 serving as valid tool links. These 36 tool links offer access to 37 tools, with 90% of them coded in Python, and one providing an out-of-the-box tool. These tools are developed for review classification, app feature extraction, matching app reviews with other artifacts, review response, and summarization. These tools are mostly analyzing static app reviews without considering app updates. Besides, They adopt sentiment analysis tools with default settings, which may decrease accuracy. In addition, we discuss open-source tools involving developers in the evaluation process and present an industrial view of user feedback and automatic tools.

7. Implications and future directions

In this part, we present existing practices in app review analysis and propose future directions as listed in Table 22. Besides, we discuss the impact of LLMs on app review analysis and present implications of our SLR for SE community.

7.1. Existing practices

Research on app reviews commenced in 2012, and since then, the number of studies in this domain has consistently increased. These studies provide valuable experience regarding collecting app reviews, and handling and exploiting common review characteristics. In this subsection, we discuss practices belonging to all nine areas observed in our SLR.

7.1.1. App distribution platform selection

The initial publications on app reviews emerged in 2012, primarily concentrating on the Apple App Store. Subsequently, there was a gradual decline in the number of papers studying the Apple App Store. In the subsequent year (2013), research expanded to include papers on the Google Play Store, with over 70% of the publications involving apps on this platform. Since 2014, there has been an emergence of papers examining multiple app stores. Among the surveyed papers, about 29.01% employed multiple app distribution platforms to facilitate the generalization or comparison of results across various app markets, whereas 67.90% concentrated on a single target app market. Scraping reviews from app distribution platforms is quite convenient via various official APIs or third-party APIs, with the majority of previous studies (93.18%) collecting app reviews in this way. However, there are constraints associated with scraping data from app stores, including limitations on the number of reviews that can be crawled, permissions to download only the reviews of the latest app version, and restrictions allowing the crawling of only the most recent review from each user for each app. Martin et al. (2017) proved that the incomplete review dataset would lead to biased results. To address these limitations, previous studies need to collect as many reviews as possible, and two solutions exist for this purpose. The first involves turning to third-party app monitoring platforms, while the second requires keeping track of app reviews from studied apps.

7.1.2. Mobile app selection

We observed a highly right-skewed distribution in the number of apps collected in previous studies. In 60.13% of the papers, the number of selected apps is less than 100, and 51.58% of these papers do not exceed ten. Specifically, during the process of mobile app selection, existing studies take into account factors like app popularity, app category, and research purpose. Researchers tend to choose popular apps due to their higher review counts, providing more information for extraction. Regarding app categories, there are two contrasting approaches. Typically, the selection of apps considers diverse categories. However, some studies specifically focus on certain app categories, such as the Productivity category. Additionally, apps can be chosen based on research objectives. For instance, in studies involving cross-platform apps, researchers select apps that receive feedback on all app distribution platforms they are investigating (Oehri and Guzman, 2020).

7.1.3. App review collection

In this area, we provide an overview of the app review population size and the sampling strategies employed in previous studies. The app review population size is evenly distributed among 1k~10k, 10k~100k, 100k~1M, 1M~10M, accounting for 18.99%, 21.52%, 20.89%, and 21.52% respectively. A total of 60 papers proceed to extract much smaller subsets from the collected app review populations using techniques such as simple random sampling, stratified sampling, or key-point investigation. The primary purpose of sampling a subset is to conduct manual labeling for evaluating proposed automated tools. Another purpose is for the authors to undertake manual analysis, such as categorizing reviews based on app aspects. We find that 83.33% of papers sample less than 10,000 reviews, and 55.00% of papers sample less than 3,000 reviews.

7.1.4. Rating

When users review apps, they assign star ratings as a quantitative measure of their preferences for apps. The handling and applications for review rating include using ratings to construct features/metrics, select app reviews, conduct descriptive statistical analysis and inferential statistical analysis, directly employing ratings as input features for machine learning and deep learning methods, comparing different app groups based on ratings, and exploring the relationship between ratings and app updates.

7.1.5. Length

Hoon and his colleagues (Hoon et al., 2012; Vasa et al., 2012) discovered that the median feedback length across all applications is 69 characters, with a mean of 117, suggesting that users typically provide short feedback. Besides, app review length has been observed to be associated with review rating, sentiment, app category, and cultural factors. The handling and applications for review length include condensing lengthier reviews, utilizing review length as a criterion for selecting app reviews, conducting inferential statistical analysis, employing review length directly as an input feature for machine learning and deep learning methods, and comparing various app/review groups based on review length. Additionally, the short nature of app reviews is taken into consideration when selecting appropriate models and determining model inputs.

7.1.6. Domain knowledge

App reviews, often contributed by users with limited expertise, lack specific technical details, thereby reducing their utility for developers. To enhance the informativeness of app reviews, it is advisable to engage in joint mining with domain knowledge, incorporating elements such as device information, app price, and app category. The handling and applications for domain knowledge include constructing features, selecting app reviews, topic modeling, combining with app store information, bug/issue reports, changelogs/release notes, static analysis information, source code, and non-SE information.

7.1.7. Textual content

The characteristics of review content encompass restricted vocabulary usage, an informal language style, and polarity in informativeness. The handling and applications for review textual content involve noise removal, noise reduction, informal language correction, review-level filtering, tokenization, lexical analysis, syntactic analysis, and semantic analysis.

Table 22
Existing practices and future directions for areas proposed in our SLR (“EP” stands for existing practice, and “FD” stands for future direction).

Types	Existing practices/future directions
App Distribution Platform Selection	EP: Selecting one app distribution platform EP: Selecting multiple app distribution platforms EP: Using app monitoring platforms EP: Keeping tracking app distribution platforms FD: Studying mini-program platforms FD: Studying differences between reviews on app monitoring platforms and reviews on official app distribution platforms
Mobile App Selection	EP: App selection according to app popularity EP: App selection according to app category EP: App selection according to research purpose FD: Studying competing apps with similar app features
App Review Collection	EP: Collected app review population EP: Simple random sampling EP: Stratified sampling EP: Key-point investigation FD: Applying other sampling strategies FD: Building large-scale datasets with fine-grained labels FD: Studying differences between old-dated reviews and recent reviews
Rating	EP: Constructing features/metrics EP: App review selection EP: Descriptive statistical analysis EP: Inferential statistical analysis EP: Utilizing review rating as an input feature EP: Comparison EP: Relationship with app update
Length	EP: Condensing longer reviews EP: App review selection EP: Inferential statistical analysis EP: Utilizing review length as an input feature EP: Comparison EP: Model selection EP: Determining model input
Domain knowledge	EP: Constructing features EP: App review selection EP: Topic modeling EP: Combining with app store information EP: Combining with bug/issue reports EP: Combining with changelog/release notes EP: Combining with static analysis information EP: Combining with source code EP: Combining with non-SE information FD: Combining with multimodal data FD: Exploiting the domain knowledge commonly perceived by users in app reviews FD: Focusing on app-specific questions
Textual content	EP: Noise removal EP: Informal language correction EP: Noise reduction EP: Review-level filtering EP: Tokenization EP: Lexical analysis EP: Syntactic analysis EP: Semantic analysis FD: Adjusting analysis level for app reviews FD: Exploring more text features
Dynamic nature	EP: Handling dynamic submissions EP: Constructing features EP: Generating time series data EP: Trend analysis EP: Comparison EP: Exploiting submission date FD: Generating time series numerical features
Sentiment	EP: Review text filtering EP: Sentiment word extraction EP: Deriving sentiment scores EP: Deriving sentiment polarities EP: Constructing features EP: Generating aspect-sentiment pairs EP: Analyzing emoticons FD: Developing customized sentiment tools for app reviews

7.1.8. Dynamic nature

App reviews exhibit a dynamic nature both at the macro and micro levels. Macroscopically, reviews are influenced by app updates, reflecting varying issues and user experiences across different app versions. Microscopically, users update reviews for different app releases or after communication with the app teams. The handling and applications for review dynamic nature involve handling dynamic submissions, constructing features based on time info, generating time series data, performing trend analysis, making time-based comparisons, and leveraging submission dates.

7.1.9. Sentiment

The most common words in user reviews tend to convey emotion, with a notably large number of words expressing negative sentiment. The handling and applications for review sentiment include review text filtering based on sentiment, sentiment word extraction, deriving sentiment scores and polarities, constructing features based on sentiment, generating aspect-sentiment pairs, and analyzing emoticons.

7.2. Discussion

In this part, we explore the impact of *LLMs* on app review analysis and present implications of our SLR for SE community.

7.2.1. Discussion on LLMs

Since 2021, LLMs have been increasingly utilized in app review analysis. We have identified nine papers that leverage LLMs for various purposes, including review classification (Mekala et al., 2021; Khalajzadeh et al., 2022; Harkous et al., 2022), feature extraction (de Araújo and Marcacini, 2021; Wang et al., 2022a), review prioritization (Fereidouni et al., 2022), matching bug reports with app reviews (Haering et al., 2021; Wang et al., 2022b), and linking app reviews with source code (Hu et al., 2022). Commonly employed models include BERT, T5, and their derivatives. The use of large language models reduces the need for extensive manual data processing, offering a convenient and efficient approach. Integrating LLMs into app review analysis enables the design of more effective automated tools to aid developers.

Notably, previous research highlights the significant performance improvement achievable with large language models. For instance, Khalajzadeh et al. (2022) compared traditional machine learning methods with large language models for classifying human-centered issues, revealing that the BERT classifier outperformed the machine learning model with the best performance. Furthermore, LLMs demonstrate a robust understanding of text without excessive noise removal, a limitation in traditional machine learning models that heavily rely on carefully designed features and high-quality data.

In the context of LLMs, characteristics mentioned in our systematic literature review continue to provide valuable insights for researchers and developers. For instance, Haering et al. (2021), in their work on matching bug reports with app reviews, utilized DistilBERT to represent app reviews and bug reports. Their emphasis on nouns, excluding other parts of speech tokens, was driven by the recognition that nouns often describe components and app features. Notably, their experiments uncovered that certain words, like verbs, introduced bias and hindered performance. This underscores the importance of emphasizing the information within nouns, enabling improvements in model performance.

7.2.2. Implications for SE community

The impact of our SLR in the software engineering community is substantial, benefiting both developers and app platforms. For developers, the SLR facilitates more efficient use of reviews by allowing them to focus on reviews of interest and swiftly comprehend user-reported problems and needs. Simultaneously, for app platforms, the SLR provides guidance in identifying fraudulent reviews and undesirable behaviors. This aids in standardizing platform management, enabling effective control and removal of misleading reviews, ultimately earning user support.

Additionally, apart from app reviews, our SLR can offer insights into other types of comment data, like those attached to bug reports and comments on GitHub issues. This aids developers and researchers in identifying relevant practices outlined in our SLR. Drawing insights from existing work, we present a set of review characteristics, as well as handling and processing techniques tailored for developers and researchers. These techniques, coupled with common characteristics to consider, aim to enhance the effective utilization of user reviews in various contexts.

7.3. Future directions

In this subsection, we present future research directions among all nine areas observed in our SLR.

7.3.1. App distribution platform selection

Future Direction 1: Studying mini-program platforms. Since 2017, the introduction of mini-programs by WeChat has sparked a trend embraced by other platforms like Alipay, Baidu, and TikTok. Mini programs offer a quicker and cost-effective alternative to standard apps, especially favorable for smaller developer teams. Unlike traditional apps, they eliminate the need for installation wait times, optimizing the overall user experience. This trend is expected to shape the future of app development. Taking WeChat's mini-programs as an illustration, the platform offers textual user reviews, ratings, and insightful details such as the number of total ratings and textual reviews, and star rating distribution. These user reviews can be sorted by star rating or time, and receive helpfulness votes from other users. Hence, research is essential to examine the alignment of reviews on mini-program platforms with those on conventional app distribution platforms. This investigation aims to uncover potential disparities and pinpoint distinctive topics specific to mini-program platforms, such as compatibility.

Future Direction 2: Studying differences between reviews on app monitoring platforms and reviews on official app distribution platforms. To overcome limitations in scraping data from app stores (e.g., restrictions on the number of reviews that can be crawled), one solution is to resort to third-party app monitoring platforms. However, to the best of our knowledge, there are no studies on the consistency and reliability of data extracted from these third-party app monitoring platforms.

7.3.2. Mobile app selection

Future Direction 3: Studying competing apps with similar app features. Existing studies consider factors such as app popularity, app category, and research purpose during the process of mobile app selection. However, these standards do not take into account the user's perspective. For instance, when a user seeks a translation app, falling under categories like Productivity and Utilities, they typically employ keyword searches like "translation" instead of browsing entire categories Productivity and Utilities. We suggest future studies to analyze user opinions on these competing apps with similar functions, offering practical insights for developers to excel in intense app competition, where users often compare similar apps rather than exploring entire categories.

7.3.3. App review collection

Future Direction 4: Applying other sampling strategies. Previous research has utilized techniques like simple random sampling, stratified sampling, and key-point investigation to obtain statistically significant samples from populations. We suggest future investigations explore alternative sampling strategies, such as systematic sampling and snowball sampling. Additionally, researchers could examine

Table 23
Main differences between our study and previous surveys.

Dimensions	Our study	Genc-Nayebi and Abran (2017)	Tavakoli et al. (2018)	Noei and Lyons (2019)	Dąbrowski et al. (2022a)	Lin et al. (2022)
Time Period	2012–2022	2011–2015	2011–2017	2012–2019	2010–2020	not specified
No. Papers	167	24	34	21	182	185
Paper Demographics	✓		✓		✓	
App Distribution Platform Selection (RQ 1.1)	✓			✓		
Mobile App Selection (RQ 1.2)	✓					
App Review Collection (RQ 1.3)	✓					
Discussed Characteristics (RQ 2)	✓					
App Categories Trend (RQ 2)	✓					
Review Type Trend (RQ 2)	✓					
Exploiting App Review Characteristics (RQ 3)	✓			✓	✓	✓
Publicly Available Tools for App Review Analysis (RQ 4)	37 tools		9 tools		16 tools	4 tools

potential differences in review content based on different sampling strategies and assess the impact of these strategies on the app review characteristics outlined in our SLR.

While there are a large number of previous studies on app review analysis, the use of diverse datasets and analysis perspectives poses challenges for comparison. On the one hand, app reviews offer rich information, enabling researchers to analyze from various views based on their interests (e.g., software evolution, user complaints, undesired behaviors of apps). On the other hand, this diversity also complicates tool comparisons. For instance, review classification, a fundamental task, has seen various proposed methods, making comparisons challenging when the categories differ. The variety of classifications further limits subsequent researchers from directly applying existing tools, necessitating the initiation of work on app reviews with tasks like classification or clustering. Consequently, existing automatic classification tools may not effectively assist developers and researchers in reducing their workload. As for the availability and replicability of datasets and tools used in these 167 papers, out of 61 papers sharing open source links to datasets, 14 are invalid, leaving 47 as valid links to datasets. Regarding open-source tools, a total of 48 papers provide replication links, with 12 links being invalid and 36 serving as valid tool links. These 36 tool links offer access to 37 tools, with 90% of them coded in Python, and one providing an out-of-the-box tool (Gao et al., 2018a). Despite the variety, only eight of these tools were utilized as baselines for comparison. Therefore, we propose two future research directions for datasets as follows:

Future Direction 5: Building large-scale datasets with fine-grained labels. Large-scale datasets with fine-grained labels (so that researchers can merge the fine-grained labels as they want) need to be built.

Future Direction 6: Studying differences between outdated reviews and recent reviews. Most of the data is too old and not continuously updated. For example, the dataset in Guzman and Maalej (2014) is the most frequently adopted dataset among 167 papers and this dataset was collected in 2014. We suggest that empirical studies are needed to see if there are any differences between the old and new app review datasets. Besides, keeping updating data is also practical.

As shown in Sections 5.1 and 5.2, the numerical characteristics of app review (i.e., Rating and Length) have been thoroughly studied. However, there remains a need for further research into other characteristics that have not been fully explored yet. Hence, we provide suggestions for future research directions as follows:

7.3.4. Domain knowledge

Future Direction 7: Combining with multimodal data. Apart from the conventional app store metadata, certain information remains unexplored in conjunction with app review analysis. For example, Apple App Store provides "app preview" (i.e., screenshots of an app interface) to present the user interface and main app functions for users. So far, app reviews have been jointly mined with text and numerical data, but they have not been integrated with other types of data, such as pictures and videos. Therefore, we believe this integration will provide new insights for app review analysis. Besides, we suggest researchers explore multimodal data fusion techniques for jointly mining app reviews with other sources of information. This suggestion arises from the fact that existing research is limited to matching different data via calculating similarity and subsequently merging these similar pairs together in the form of raw text.

Future Direction 8: Exploiting the domain knowledge commonly perceived by users in app reviews. This suggestion is inspired by previous study (Yu et al., 2022). In their work, they utilized *Visible Label Information* in reviews, such as widgets (e.g., "reply button"). Besides, they consider app function-related verb phrases related to *Activities* in *AndroidManifest.xml* file, Class/Method Names in source code.

Future Direction 9: Focusing on app-specific questions. Regarding the context for the usage of domain knowledge, researchers should focus on app-specific questions that differ between mobile apps and traditional software. For example, it is worth paying special attention to user feedback associated with mobile app usage issues. Given the challenges of operating on smaller screens, users may encounter difficulties with navigation, touch controls, and interacting with small icons and buttons. From the perspective of app performance, optimizing program loading time and responsiveness becomes imperative, given that mobile users frequently switch between different apps simultaneously. Moreover, the trend to integrate applets and extensions into mobile apps further reinforces these worries. Compatibility issues arise since applets and extensions must cater to various devices, operating systems, and application versions. Besides, applets and extensions can significantly impact an application's performance, leading to slower load times, increased resource consumption, and decreased responsiveness. Security is another vital aspect to address, as these components might access sensitive user data or system resources, requiring careful validation and scrutiny. Lastly, ensuring that each extension harmoniously complements the overall application design and functionality without disrupting user experience is essential.

7.3.5. Textual content

Future Direction 10: Adjusting analysis level for app reviews. Existing research either analyzes the entire review as a whole or splits it into individual sentences for analysis. However, these two analysis levels have limitations. The former mixes multiple app aspects together for analysis, while the latter misses the user usage scenarios in the context. We recommend splitting the review according to different app aspects and analyzing the review with contextual information.

Future Direction 11: Exploring more text features. For instance, readability serves as a metric to gauge review quality, while words in

Table 24
List of the surveyed papers.

Study ID	Title	Year	Reference
1	A preliminary analysis of mobile app user reviews	2012	Vasa et al. (2012)
2	A preliminary analysis of vocabulary in mobile app user reviews	2012	Hoon et al. (2012)
3	Managing the enterprise business intelligence app store: Sentiment analysis supported requirements engineering	2012	Goul et al. (2012)
4	User feedback in the appstore: An empirical study	2013	Pagano and Maalej (2013)
5	What are you complaining about?: A study of online reviews of mobile applications	2013	Iacob et al. (2013)
6	Do Android users write about electric sheep? Examining consumer reviews in Google Play	2013	Ha and Wagner (2013)
7	Why people hate your app: making sense of user feedback in a mobile app store	2013	Fu et al. (2013)
8	Awesome!: Conveying satisfaction on the app store	2013	Hoon et al. (2013)
9	Trusting smartphone Apps? To install or not to install, that is the question	2013	Kuehnhausen and Frost (2013)
10	Retrieving and analyzing mobile apps feature requests from online reviews	2013	Iacob and Harrison (2013)
11	Accessibility in smartphone applications: What do we learn from reviews?	2013	Anam and Yeasin (2013)
12	Analysis of user comments: An approach for software requirements evolution	2013	Carreño and Winbladh (2013)
13	Serendipitous recommendation for mobile apps using item-item similarity graph	2013	Bhandari et al. (2013)
14	AR-miner: Mining informative reviews for developers from mobile app marketplace	2014	Chen et al. (2014)
15	How do users like this feature? A fine grained sentiment analysis of app reviews	2014	Guzman and Maalej (2014)
16	FAVe: Visualizing user feedback for software evolution	2014	Guzman et al. (2014)
17	Prioritizing the devices to test your app on: A case study of Android game apps	2014	Khalid et al. (2014)
18	Popularity modeling for mobile apps: A sequential approach	2014	Zhu et al. (2014)
19	For user-driven software evolution: Requirements elicitation derived from mining online reviews	2014	Jiang et al. (2014)
20	What do mobile app users complain about?	2015	Khalid et al. (2015)
21	Mining user opinions in mobile app reviews: A keyword-based approach	2015	Phong et al. (2015)
22	Retrieving diverse opinions from app reviews	2015	Guzman et al. (2015a)
23	What parts of your apps are loved by users?	2015	Gu and Kim (2015)
24	The app sampling problem for app store mining	2015	Martin et al. (2015)
25	PAID: Prioritizing app issues for developers by tracking user reviews over versions	2015	Gao et al. (2015)
26	User reviews matter! Tracking crowdsourced reviews to support evolution of successful apps	2015	Palomba et al. (2015)
27	Generative models for mining latent aspects and their ratings from short reviews	2015	Li et al. (2015)
28	SimApp: A framework for detecting similar mobile applications by online kernel learning	2015	Chen et al. (2015)
29	How can I improve my app? Classifying user reviews for software maintenance and evolution	2015	Panichella et al. (2015)
30	AUTOREB: Automatically understanding the review-to-behavior fidelity in Android applications	2015	Kong et al. (2015)
31	Discovery of ranking fraud for mobile apps	2015	Zhu et al. (2015)
32	Leveraging user reviews to improve accuracy for mobile app retrieval	2015	Park et al. (2015)
33	Ensemble methods for app review classification: An approach for software evolution	2015	Guzman et al. (2015b)
34	Identification and classification of requirements from app user reviews	2015	Yang and Liang (2015)
35	On user rationale in software engineering	2015	Kurtanović and Maalej (2018)
36	Mobile app security risk assessment: A crowdsourcing ranking approach from user comments	2015	Cen et al. (2015)
37	It feels different from real life: Users' opinions of mobile applications for mental health	2015	de Alva et al. (2015)
38	What would users change in my app? Summarizing app reviews for recommending software changes	2016	Di Sorbo et al. (2016)
39	Experience report: Understanding cross-platform app issues from user reviews	2016	Man et al. (2016)
40	Examining the relationship between FindBugs warnings and app ratings	2016	Khalid et al. (2016)
41	Approaches for prioritizing feature improvements extracted from app reviews	2016	Keertipati et al. (2016)
42	Phrase-based extraction of user opinions in mobile app reviews	2016	Vu et al. (2016)
43	Analyzing and automatically labelling the types of user issues that are raised in mobile app reviews	2016	McIlroy et al. (2016)
44	On the automatic classification of app reviews	2016	Maalej et al. (2016)
45	SAFE: A simple approach for feature extraction from app descriptions and app reviews	2017	Johann et al. (2017)
46	Automatic classification of non-functional requirements from augmented app user reviews	2017	Lu and Liang (2017)
47	Users — The hidden software product quality experts?: A study on how app users report quality aspects in online reviews	2017	Groen et al. (2017)
48	OASIS: prioritizing static analysis warnings for Android apps based on app user reviews	2017	Wei et al. (2017)
49	Attributes that predict which features to fix: Lessons for app store mining	2017	Licorish et al. (2017)
50	Analyzing reviews and code of mobile apps for better release planning	2017	Ciurumelea et al. (2017)
51	Recommending and localizing change requests for mobile apps based on user reviews	2017	Palomba et al. (2017)
52	Is it worth responding to reviews? Studying the top free apps in Google Play	2017	McIlroy et al. (2017)
53	Search rank fraud and malware detection in Google Play	2017	Rahman et al. (2017)
54	Version-aware rating prediction for mobile app recommendation	2017	Yao et al. (2017)
55	An automated feedback-based approach to support mobile app development	2017	Scherr et al. (2017)
56	Crowdsourced app review manipulation	2017	Li et al. (2017)
57	Exploiting user feedback in tool-supported multi-criteria requirements prioritization	2017	Morales-Ramirez et al. (2017)
58	Preliminary study on applying semi-supervised learning to app store analysis	2017	Deocadez et al. (2017)
59	A first look at mobile ad-blocking apps	2017	Ikram and Kaafar (2017)
60	QuickReview: A novel data-driven mobile user interface for reporting problematic app features	2017	Su'a et al. (2017)
61	A study of the relation of mobile device attributes with the user-perceived quality of Android apps	2017	Noei et al. (2017)
62	Fine-grained opinion mining from mobile app reviews with word embedding features	2017	S¨anger et al. (2017)
63	Mobile apps for mood tracking: An analysis of features and user reviews	2017	Caldeira et al. (2017)
64	INFAR: insight extraction from app reviews	2018	Gao et al. (2018a)
65	App review analysis via active learning: Reducing supervision effort without compromising classification accuracy	2018	Dhinakaran et al. (2018)
66	What aspects do non-functional requirements in app user reviews describe? An exploratory and comparative study	2018	Wang et al. (2018a)
67	Exploring the integration of user feedback in automated testing of Android applications	2018	Grano et al. (2018)
68	The OIRE method - Overview and initial validation	2018	Yin and Pfahl (2018)
69	Online app review analysis for identifying emerging issues	2018	Gao et al. (2018b)

(*continued on next page*)

Table 24 (*continued*).

Study ID	Title	Year	Reference
70	Winning the app production rally	2018	Noei et al. (2018)
71	Studying the dialogue between users and developers of free apps in the Google Play Store	2018	Hassan et al. (2018)
72	Automating developers' responses to app reviews	2018	Greenheld et al. (2018)
73	Can app changelogs improve requirements classification from app reviews? An exploratory study	2018	Wang et al. (201b8)
74	Sentiment analysis for software engineering: How far can we go?	2018	Lin et al. (2018)
75	App store mining is not enough for app improvement	2018	Nayebi et al. (2018)
76	What people like in mobile finance apps: An analysis of user reviews	2018	Huebner et al. (2018)
77	Modeling user concerns in the app store: A case study on the rise and fall of Yik Yak	2018	Williams and Mahmoud (2018)
78	Using frame semantics for classifying and summarizing application store reviews	2018	Jha and Mahmoud (2018)
79	Improvement of user review classification using keyword expansion	2018	Higashi et al. (2018)
80	Competitivebike: Competitive analysis and popularity prediction of bike-sharing apps using multi-source data	2018	Ouyang et al. (2018)
81	Gender and user feedback: An exploratory study	2019	Guzman and Paredes Rojas (2019)
82	Same app, different countries: A preliminary user reviews study on most downloaded iOS apps	2019	Srisopha et al. (2019)
83	Too many user-reviews! What should app developers look at first?	2019	Noei et al. (2021)
84	Exploiting natural language structures in software informal documentation	2019	Di Sorbo et al. (2021b)
85	Listening to the crowd for the release planning of mobile apps	2019	Scalabrino et al. (2019)
86	Short text, large effect: Measuring the impact of user reviews on Android app security & privacy	2019	Nguyen et al. (2019)
87	Automating app review response generation	2019	Gao et al. (2019)
88	When people and algorithms meet: User-reported problems in intelligent everyday applications	2019	Eiband et al. (2019)
89	An empirical study of game reviews on the Steam platform	2019	Lin et al. (2019)
90	RE-SWOT: From user feedback to requirements via competitor analysis	2019	Dalpiaz and Parente (2019)
91	Studying the consistency of star ratings and reviews of popular free hybrid Android and iOS apps	2019	Hu et al. (2019a)
92	Mining non-functional requirements from App store reviews	2019	Jha and Mahmoud (2019)
93	Release early, release often, and watch your users' emotions: Lessons from emotional patterns	2019	Martens and Maalej (2019a)
94	Towards understanding and detecting fake reviews in app stores	2019	Martens and Maalej (2019b)
95	Finding and analyzing app reviews related to specific features: A research preview	2019	Dąbrowski et al. (2019)
96	Towards prioritizing user-related issue reports of mobile applications	2019	Noei et al. (2019)
97	Software feature refinement prioritization based on online user review mining	2019	Zhang et al. (2019)
98	Simplifying the classification of app reviews using only lexical features	2019	Shah et al. (2019)
99	Dating with scambots: Understanding the ecosystem of fraudulent dating applications	2019	Hu et al. (2019c)
100	Want to earn a few extra bucks? A first look at money-making apps	2019	Hu et al. (2019b)
101	Revealing the unrevealed: Mining smartphone users privacy perception on app markets	2019	Hatamian et al. (2019)
102	Usability issues in mental health applications	2019	Alqahtani and Orji (2019)
103	Same same but different: Finding similar user feedback across multiple platforms and languages	2020	Oehri and Guzman (2020)
104	AOBTM: Adaptive online biterm topic modeling for version sensitive short-texts analysis	2020	Hadi and Fard (2020)
105	Caspar: Extracting and synthesizing user stories of problems from app reviews	2020	Guo and Singh (2020)
106	Order in chaos: Prioritizing mobile app reviews using consensus algorithms	2020	Etaiwi et al. (2020)
107	Studying bad updates of top free-to-download apps in the Google Play Store	2020	Hassan et al. (2020)
108	Feature recommendation by mining updates and user feedback from competitor apps	2020	Uddin et al. (2020b)
109	App competition matters: How to identify your competitor apps?	2020	Uddin et al. (2020a)
110	What people focus on when reviewing your app-An analysis across app categories	2020	Huebner et al. (2020)
111	SmartPI: Understanding permission implications of Android apps from user reviews	2020	Wang et al. (2020)
112	App-aware response synthesis for user reviews	2020	Farooq et al. (2020)
113	A semantic-based framework for analyzing app users' feedback	2020	Yadav et al. (2020)
114	How features in iOS app store reviews can predict developer responses	2020	Srisopha et al. (2020b)
115	Learning features that predict developer responses for iOS app store reviews	2020	Srisopha et al. (2020a)
116	The use of pretrained model for matching app reviews and bug reports	2020	Wang et al. (2022b)
117	App popularity prediction by incorporating time-varying hierarchical interactions	2020	Zhang et al. (2020b)
118	Accessibility issues in Android apps: State of affairs, sentiments, and ways forward	2020	Alshayban et al. (2020)
119	Mining user opinions to support requirement engineering: An empirical study	2020	Dąbrowski et al. (2020)
120	Identifying and classifying user requirements in online feedback via crowdsourcing	2020	van Vliet et al. (2020)
121	Comparing mobile apps by identifying 'Hot' features	2020	Malik et al. (2020)
122	Investigating user perceptions of mobile app privacy: An analysis of user-submitted app reviews	2020	Besmer et al. (2020)
123	Identifying security issues for mobile applications based on user review summarization	2020	Tao et al. (2020)
124	Insights from user reviews to improve mental health apps	2020	Alqahtani and Orji (2020)
125	Classifying user requirements from online feedback in small dataset environments using deep learning	2021	Mekala et al. (2021)
126	Exploiting the unique expression for improved sentiment analysis in software engineering text	2021	Sun et al. (2021)
127	Finding the needle in a haystack: On the automatic identification of accessibility user reviews	2021	AlOmar et al. (2021)
128	RE-BERT: automatic extraction of software requirements from app reviews using BERT language model	2021	de Araújo and Marcacini (2021)
129	Generating functional requirements based on classification of mobile application user reviews	2021	Panthum and Senivongse (2021)
130	How should I improve the UI of my app? a study of user reviews of popular apps in the Google Play	2021	Chen et al. (2021)
131	Identifying key features from app user reviews	2021	Wu et al. (2021)
132	Automatically matching bug reports with related app reviews	2021	Haering et al. (2021)
133	User review-based change file localization for mobile applications	2021	Zhou et al. (2021)
134	Where2Change: Change request localization for app reviews	2021	Zhang et al. (2021)
135	CHAMP: Characterizing undesired app behaviors from user comments based on market policies	2021	Hu et al. (2021)
136	Automating app review response generation based on contextual knowledge	2021	Gao et al. (2021)
137	Towards de-anonymization of Google Play search rank fraud	2021	Rahman et al. (2021)
138	An exploratory analysis of human-centric issues in parking solutions using surveys and mobile parking app reviews	2021	Li et al. (2021)
139	The role of user reviews in app updates: A preliminary investigation on app release notes	2021	Wang et al. (2021)
140	Investigating the criticality of user-reported issues through their relations with app rating	2021	Di Sorbo et al. (2021a)
141	Featcompare: Feature comparison for competing mobile apps leveraging user reviews	2021	Assi et al. (2021)
142	Ah-cid: A tool to automatically detect human-centric issues in app reviews	2021	Mathews et al. (2021)

(*continued on next page*)

Table 24 (*continued*).

Study ID	Title	Year	Reference
143	The rise and fall of covid-19 contact-tracing apps: When NFRs collide with pandemic	2021	Bano et al. (2021)
144	Emerging app issue identification via online joint sentiment-topic tracing	2022	Gao et al. (2022b)
145	Multisource heterogeneous user-generated contents-driven interactive estimation of distribution algorithms for personalized search	2022	Bao et al. (2022)
146	Towards automatically localizing function errors in mobile apps with user reviews	2022	Yu et al. (2022)
147	Listening to users' voice: Automatic summarization of helpful app reviews	2022	Gao et al. (2022a)
148	Analyzing user perspectives on mobile app privacy at scale	2022	Nema et al. (2022)
149	Proactive prioritization of app issues via contrastive learning	2022	Fereidouni et al. (2022)
150	Where is your app frustrating users?	2022	Wang et al. (2022a)
151	On the importance of performing app analysis within peer groups	2022	Hassan et al. (2022)
152	Mining user feedback for software engineering: Use cases and reference architecture	2022	Dąbrowski et al. (2022b)
153	Domain-specific analysis of mobile app reviews using keyword-assisted topic models	2022	Tushev et al. (2022)
154	Supporting developers in addressing human-centric issues in mobile apps	2022	Khalajzadeh et al. (2022)
155	On the violation of honesty in mobile apps: Automated detection and categories	2022	Obie et al. (2022)
156	Which app is going to die? A framework for app survival prediction with multitask learning	2022	Zhang et al. (2020a)
157	Hark: A deep learning system for navigating privacy feedback at scale	2022	Harkous et al. (2022)
158	Demystifying "removed reviews" in iOS app store	2022	Wang et al.
159	Unsupervised summarization of privacy concerns in mobile application reviews	2022	Ebrahimi and Mahmoud (2022)
160	Hierarchical Bayesian multi-kernel learning for integrated classification and summarization of app reviews	2022	Alshangiti et al. (2022)
161	Better identifying and addressing diverse issues in mhealth and emerging apps using user reviews	2022	Haggag (2022)
162	User perspectives and ethical experiences of apps for depression: A qualitative analysis of user reviews	2022	Bowie-DaBreo et al. (2022)
163	AccessiText: automated detection of text accessibility issues in Android apps	2022	Alshayban and Malek (2022)
164	Erasing labor with labor: Dark patterns and lockstep behaviors on Google Play	2022	Singh et al. (2022)
165	Lighting up supervised learning in user review-based code localization: dataset and benchmark	2022	Hu et al. (2022)
166	Prioritizing user concerns in app reviews—A study of requests for new features, enhancements and bug fixes	2022	Malgaonkar et al. (2022)
167	An empirical study on release notes patterns of popular apps in the Google Play Store	2022	Yang et al. (2022)

all capital letters might highlight specific app aspects or emotions. Numerical values may offer device and version information, and reviews with different numbers/proportions of spelling errors convey different levels of information. Furthermore, we suggest that more metrics and features from linguistics can be introduced.

7.3.6. Dynamic nature

Future Direction 12: Generating time series numerical features. Existing research only uses simple dynamic information, such as the post time of app review or the corresponding release version. We suggest that time series analysis methods can be used to construct deeper features on the daily/per app release numerical feature (e.g., rating, quantity), such as *difference* and *growth rate*. Researchers can also study time series patterns e.g., clustering similar patterns, which is beneficial for monitoring the app's health from the user's perspective.

7.3.7. Sentiment

Future Direction 13: Developing customized sentiment tools for app reviews. Existing research exploits out-of-the-box sentiment analysis tools, but the results of previous work (Lin et al., 2018) show that this is not applicable to SE data. Furthermore, the performance of the customized tool Sentistrength-SE (Islam and Zibran, 2018), when applied to app reviews, is even worse than that of SentiStrength (Thelwall et al., 2010), NLTK,[25] and Stanford CoreNLP (Manning et al., 2014). Therefore, more reliable sentiment analysis for app reviews should be developed. For example, the review rating or other indicators can be used as response variables to learn the sentiment score of each word to obtain more accurate sentiment information, which can also help to expand the sentiment lexicon in the literature.

8. Related work

This review is not the initial attempt to synthesize knowledge from the literature analyzing app reviews from the perspective of software engineering ((Dąbrowski et al., 2022a; Genc-Nayebi and Abran, 2017; Tavakoli et al., 2018; Noei and Lyons, 2019; Lin et al., 2022)). Our SLR, however, differs substantially from previous studies in scope of the literature surveyed and the perspective of our analysis. Following the work of Dąbrowski et al. (2022a), we show the differences between our study and previous works in Table 23. We compare the following dimensions: time period covered, number of papers surveyed, and the topics we focus on, i.e., Paper Demographics, App Distribution Platform Selection (RQ 1.1), Mobile App Selection (RQ 1.2), App Review Collection (RQ 1.3), Discussed Characteristics (RQ 2), App Categories Trend (RQ 2), Review Type Trend (RQ 2), Exploiting App Review Characteristics (RQ 3), and Publicly Available Tools for App Review Analysis (RQ 4).

While Noei and Lyons (2019) also wrote about the methods and limitations of collecting application reviews, they only delved into influential studies on Google Play user reviews. In contrast, our discussion encompasses various app distribution platforms, including the influential Apple App Store. We conducted a comparison of comments collected from these platforms, providing a statistical analysis of platform selection in previous papers.

Moreover, while several papers explore the characteristics of app reviews, our focus is distinct as we consider a broader range of app review papers. Besides, we conduct a comprehensive investigation, systematically identifying key characteristics. In the SLR of Noei and Lyons (2019), the focus was solely on characteristic *Textual Content*. Lin et al. (2022) concentrated on characteristic *Sentiment*, introducing various existing sentiment analysis tools across diverse software artifacts (e.g., app reviews, bug reports). Dąbrowski et al. (2022a) also discussed characteristic *Textual Content* but emphasized mining techniques, while our goal is to present relevant characteristics and provide guidance on their utilization by summarizing common practices from literature. When analyzing tool availability, we identified more tools compared to other related SLRs. Besides, we have outlined specific components to facilitate researchers in obtaining detailed information more efficiently.

9. Conclusion

Through a systematic search, we have identified 167 relevant studies, which are thoroughly examined to answer our research questions. The results of the study show that people apply review mining to many activities of the software ecosystem. Besides, a list of review characteristics is summarized via a key-point investigation. Moreover, the literature review presents common handling and application for

[25] NLTK: https://www.nltk.org/

each review characteristic. Armed with this knowledge, researchers are able to envision the challenges of app review mining and adopt appropriate methods to mine desired information from app reviews. Lastly, we discuss the future research directions.

CRediT authorship contribution statement

Xiaohui Wang: Investigation, Methodology, Validation, Writing – original draft. **Tao Zhang:** Validation, Supervision, Writing – review & editing, Funding acquisition, Resources. **Youshuai Tan:** Data curation, Investigation, Writing – review & editing. **Weiyi Shang:** Supervision, Writing – review & editing. **Yao Li:** Resources, Writing – review & editing.

Declaration of competing interest

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

Data availability

No data was used for the research described in the article.

Acknowledgment

This work was supported by the Macao Science and Technology Development Fund (FDCT), Macau SAR (Grant ID: 0014/2022/A).

References

Abou Khalil, Z., Zacchiroli, S., 2022. Software artifact mining in software engineering conferences: A meta-analysis. In: ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. pp. 227–237.

Ali, M., Joorabchi, M.E., Mesbah, A., 2017. Same app, different app stores: A comparative study. In: IEEE/ACM International Conference on Mobile Software Engineering and Systems. pp. 79–90.

AlOmar, E.A., Aljedaani, W., Tamjeed, M., Mkaouer, M.W., El-Glaly, Y.N., 2021. Finding the needle in a haystack: On the automatic identification of accessibility user reviews. In: ACM Conference on Human Factors in Computing Systems. pp. 1–15.

Alqahtani, F., Orji, R., 2019. Usability issues in mental health applications. In: Conference on User Modeling, Adaptation and Personalization. pp. 343–348.

Alqahtani, F., Orji, R., 2020. Insights from user reviews to improve mental health apps. Health Inform. J. 26 (3), 2042–2066.

Alshangiti, M., Shi, W., Lima, E., Liu, X., Yu, Q., 2022. Hierarchical Bayesian multi-kernel learning for integrated classification and summarization of app reviews. In: ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering. pp. 558–569.

Alshayban, A., Ahmed, I., Malek, S., 2020. Accessibility issues in android apps: State of affairs, sentiments, and ways forward. In: ACM/IEEE International Conference on Software Engineering. pp. 1323–1334.

Alshayban, A., Malek, S., 2022. AccessiText: automated detection of text accessibility issues in android apps. In: ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering. pp. 984–995.

Anam, A.I., Yeasin, M., 2013. Accessibility in smartphone applications: what do we learn from reviews? In: International ACM SIGACCESS Conference on Computers and Accessibility. pp. 1–2.

Assi, M., Hassan, S., Tian, Y., Zou, Y., 2021. FeatCompare: Feature comparison for competing mobile apps leveraging user reviews. Empir. Softw. Eng. 26, 1–38.

Bano, M., Arora, C., Zowghi, D., Ferrari, A., 2021. The rise and fall of covid-19 contact-tracing apps: when nfrs collide with pandemic. In: IEEE International Requirements Engineering Conference. pp. 106–116.

Bao, L., Sun, X., Gong, D., Zhang, Y., 2022. Multisource heterogeneous user-generated contents-driven interactive estimation of distribution algorithms for personalized search. IEEE Trans. Evol. Comput. 26 (5), 844–858.

Besmer, A.R., Watson, J., Banks, M.S., 2020. Investigating user perceptions of mobile app privacy: An analysis of user-submitted app reviews. Int. J. Inf. Secur. Privacy 14 (4), 74–91.

Bhandari, U., Sugiyama, K., Datta, A., Jindal, R., 2013. Serendipitous recommendation for mobile apps using item-item similarity graph. In: Asia Information Retrieval Societies Conference. pp. 440–451.

Bowie-DaBreo, D., Sas, C., Iles-Smith, H., S¨unram-Lea, S., 2022. User perspectives and ethical experiences of apps for depression: A qualitative analysis of user reviews. In: ACM Conference on Human Factors in Computing Systems. pp. 1–24.

Caldeira, C., Chen, Y., Chan, L., Pham, V., Chen, Y., Zheng, K., 2017. Mobile apps for mood tracking: an analysis of features and user reviews. In: American Medical Informatics Annual Fall Symposium, vol. 2017, p. 495.

Carreño, L.V.G., Winbladh, K., 2013. Analysis of user comments: an approach for software requirements evolution. In: International Conference on Software Engineering. pp. 582–591.

Cen, L., Kong, D., Jin, H., Si, L., 2015. Mobile app security risk assessment: A crowdsourcing ranking approach from user comments. In: SIAM International Conference on Data Mining. pp. 658–666.

Chen, Q., Chen, C., Hassan, S., Xing, Z., Xia, X., Hassan, A.E., 2021. How should I improve the UI of my app? A study of user reviews of popular apps in the google play. ACM Trans. Softw. Eng. Methodol. 30 (3), 1–38.

Chen, N., Hoi, S.C., Li, S., Xiao, X., 2015. SimApp: A framework for detecting similar mobile applications by online kernel learning. In: ACM International Conference on Web Search and Data Mining. pp. 305–314.

Chen, J., Hu, Y., Liu, J., Xiao, Y., Jiang, H., 2019. Deep short text classification with knowledge powered attention. In: AAAI Conference on Artificial Intelligence, vol. 33, (01), pp. 6252–6259.

Chen, N., Lin, J., Hoi, S.C., Xiao, X., Zhang, B., 2014. AR-miner: Mining informative reviews for developers from mobile app marketplace. In: International Conference on Software Engineering. pp. 767–778.

Ciurumelea, A., Schaufelbühl, A., Panichella, S., Gall, H.C., 2017. Analyzing reviews and code of mobile apps for better release planning. In: IEEE International Conference on Software Analysis, Evolution and Reengineering. IEEE, pp. 91–102.

Clarke, S., 2014. Your digital journey is being mapped by your customers. MIT Sloan Manageme. Rev..

Corral, L., Fronza, I., 2015. Better code for better apps: A study on source code quality and market success of android applications. In: ACM International Conference on Mobile Software Engineering and Systems. pp. 22–32.

Dąbrowski, J., Letier, E., Perini, A., Susi, A., 2019. Finding and analyzing app reviews related to specific features: A research preview. In: International Conference on Requirements Engineering: Foundation for Software Quality. pp. 183–189.

Dąbrowski, J., Letier, E., Perini, A., Susi, A., 2020. Mining user opinions to support requirement engineering: an empirical study. In: International Conference on Advanced Information Systems Engineering. pp. 401–416.

Dąbrowski, J., Letier, E., Perini, A., Susi, A., 2022a. Analysing app reviews for software engineering: A systematic literature review. Empir. Softw. Eng. 27 (2), 43.

Dąbrowski, J., Letier, E., Perini, A., Susi, A., 2022b. Mining user feedback for software engineering: Use cases and reference architecture. In: IEEE International Requirements Engineering Conference. pp. 114–126.

Dalpiaz, F., Parente, M., 2019. RE-SWOT: from user feedback to requirements via competitor analysis. In: International Conference on Requirements Engineering: Foundation for Software Quality. pp. 55–70.

de Alva, F.E.M., Wadley, G., Lederman, R., 2015. It feels different from real life: users' opinions of mobile applications for mental health. In: Australian Computer Human Interaction Conference. pp. 598–602.

de Araújo, A.F., Marcacini, R.M., 2021. RE-BERT: Automatic extraction of software requirements from app reviews using BERT language model. In: Annual ACM Symposium on Applied Computing. pp. 1321–1327.

Deka, B., Huang, Z., Franzen, C., Hibschman, J., Afergan, D., Li, Y., Nichols, J., Kumar, R., 2017. Rico: A mobile app dataset for building data-driven design applications. In: Annual ACM Symposium on User Interface Software and Technology. pp. 845–854.

Deocadez, R., Harrison, R., Rodriguez, D., 2017. Preliminary study on applying semi-supervised learning to app store analysis. In: International Conference on Evaluation and Assessment in Software Engineering. pp. 320–323.

Dhinakaran, V.T., Pulle, R., Ajmeri, N., Murukannaiah, P.K., 2018. App review analysis via active learning: Reducing supervision effort without compromising classification accuracy. In: IEEE International Requirements Engineering Conference. pp. 170–181.

Di Sorbo, A., Grano, G., Aaron Visaggio, C., Panichella, S., 2021a. Investigating the criticality of user-reported issues through their relations with app rating. J. Softw.: Evol. Process 33 (3), e2316.

Di Sorbo, A., Panichella, S., Alexandru, C.V., Shimagaki, J., Visaggio, C.A., Canfora, G., Gall, H.C., 2016. What would users change in my app? Summarizing app reviews for recommending software changes. In: ACM SIGSOFT International Symposium on Foundations of Software Engineering. pp. 499–510.

Di Sorbo, A., Panichella, S., Visaggio, C.A., Di Penta, M., Canfora, G., Gall, H.C., 2021b. Exploiting natural language structures in software informal documentation. IEEE Trans. Softw. Eng. 47 (8), 1587–1604.

Doosti, B., Dong, T., Deka, B., Nichols, J., 2018. A computational method for evaluating UI patterns. arXiv preprint arXiv:1807.04191.

dos Santos, R.P., Werner, C.M.L., 2010. Revisiting the concept of components in software engineering from a software ecosystem perspective. In: Proceedings of the Fourth European Conference on Software Architecture: Companion Volume. pp. 135–142.

Ebrahimi, F., Mahmoud, A., 2022. Unsupervised summarization of privacy concerns in mobile application reviews. In: IEEE/ACM International Conference on Automated Software Engineering. pp. 1–12.

Eiband, M., V¨olkel, S.T., Buschek, D., Cook, S., Hussmann, H., 2019. When people and algorithms meet: User-reported problems in intelligent everyday applications. In: International Conference on Intelligent User Interfaces. pp. 96–106.

Eler, M.M., Orlandin, L., Oliveira, A.D.A., 2019. Do android app users care about accessibility? an analysis of user reviews on the google play store. In: Brazilian Symposium on Human Factors in Computing Systems. pp. 1–11.

Etaiwi, L., Hamel, S., Guéhéneuc, Y.-G., Flageol, W., Morales, R., 2020. Order in chaos: Prioritizing mobile app reviews using consensus algorithms. In: IEEE Annual Computers, Software, and Applications Conference. pp. 912–920.

Farooq, U., Siddique, A., Jamour, F., Zhao, Z., Hristidis, V., 2020. App-aware response synthesis for user reviews. In: IEEE International Conference on Big Data. pp. 699–708.

Fereidouni, M., Mosharrof, A., Farooq, U., Siddique, A., 2022. Proactive prioritization of app issues via contrastive learning. In: IEEE International Conference on Big Data. pp. 535–544.

Fu, B., Lin, J., Li, L., Faloutsos, C., Hong, J., Sadeh, N., 2013. Why people hate your app: Making sense of user feedback in a mobile app store. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1276–1284.

Gao, C., Li, Y., Qi, S., Liu, Y., Wang, X., Zheng, Z., Liao, Q., 2022a. Listening to users' voice: Automatic summarization of helpful app reviews. IEEE Trans. Reliab..

Gao, C., Wang, B., He, P., Zhu, J., Zhou, Y., Lyu, M.R., 2015. PAID: Prioritizing app issues for developers by tracking user reviews over versions. In: IEEE International Symposium on Software Reliability Engineering. pp. 35–45.

Gao, C., Zeng, J., Lo, D., Lin, C.-Y., Lyu, M.R., King, I., 2018a. INFAR: Insight extraction from app reviews. In: ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. pp. 904–907.

Gao, C., Zeng, J., Lyu, M.R., King, I., 2018b. Online app review analysis for identifying emerging issues. In: International Conference on Software Engineering. pp. 48–58.

Gao, C., Zeng, J., Wen, Z., Lo, D., Xia, X., King, I., Lyu, M.R., 2022b. Emerging app issue identification via online joint sentiment-topic tracing. IEEE Trans. Softw. Eng. 48 (8), 3025–3043.

Gao, C., Zeng, J., Xia, X., Lo, D., Lyu, M.R., King, I., 2019. Automating app review response generation. In: IEEE/ACM International Conference on Automated Software Engineering. pp. 163–175.

Gao, C., Zhou, W., Xia, X., Lo, D., Xie, Q., Lyu, M.R., 2021. Automating app review response generation based on contextual knowledge. ACM Trans. Softw. Eng. Methodol. 31 (1), 1–36.

Geiger, F.X., Malavolta, I., 2018. Datasets of android applications: A literature review. arXiv preprint arXiv:1809.10069.

Genc-Nayebi, N., Abran, A., 2017. A systematic literature review: Opinion mining studies from mobile app store user reviews. J. Syst. Softw. 125, 207–219.

Ghose, A., Ipeirotis, P.G., 2010. Estimating the helpfulness and economic impact of product reviews: Mining text and reviewer characteristics. IEEE Trans. Knowl. Data Eng. 23 (10), 1498–1512.

Goul, M., Marjanovic, O., Baxley, S., Vizecky, K., 2012. Managing the enterprise business intelligence app store: Sentiment analysis supported requirements engineering. In: Hawaii International Conference on System Sciences. pp. 4168–4177.

Grano, G., Ciurumelea, A., Panichella, S., Palomba, F., Gall, H.C., 2018. Exploring the integration of user feedback in automated testing of android applications. In: IEEE International Conference on Software Analysis, Evolution and Reengineering. pp. 72–83.

Greenheld, G., Savarimuthu, B.T.R., Licorish, S.A., 2018. Automating developers' responses to app reviews. In: Australasian Software Engineering Conference. pp. 66–70.

Groen, E.C., Kopczyńska, S., Hauer, M.P., Krafft, T.D., Doerr, J., 2017. Users—the hidden software product quality experts?: A study on how app users report quality aspects in online reviews. In: IEEE International Requirements Engineering Conference. pp. 80–89.

Gu, X., Kim, S., 2015. "What parts of your apps are Loved by users?". In: IEEE/ACM International Conference on Automated Software Engineering. pp. 760–770.

Guerrouj, L., Azad, S., Rigby, P.C., 2015. The influence of app churn on app success and StackOverflow discussions. In: IEEE International Conference on Software Analysis, Evolution, and Reengineering. pp. 321–330.

Guo, H., Singh, M.P., 2020. Caspar: Extracting and synthesizing user stories of problems from app reviews. In: IEEE/ACM International Conference on Software Engineering. pp. 628–640.

Guzman, E., Aly, O., Bruegge, B., 2015a. Retrieving diverse opinions from app reviews. In: ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. pp. 1–10.

Guzman, E., Bhuvanagiri, P., Bruegge, B., 2014. FAVe: Visualizing user feedback for software evolution. In: IEEE Working Conference on Software Visualization. pp. 167–171.

Guzman, E., El-Haliby, M., Bruegge, B., 2015b. Ensemble methods for app review classification: An approach for software evolution. In: IEEE/ACM International Conference on Automated Software Engineering. pp. 771–776.

Guzman, E., Maalej, W., 2014. How do users like this feature? A fine grained sentiment analysis of app reviews. In: IEEE International Requirements Engineering Conference. pp. 153–162.

Guzman, E., Oliveira, L., Steiner, Y., Wagner, L.C., Glinz, M., 2018. User feedback in the app store: A cross-cultural study. In: IEEE/ACM International Conference on Software Engineering: Software Engineering in Society. pp. 13–22.

Guzman, E., Paredes Rojas, A., 2019. Gender and user feedback: An exploratory study. In: IEEE International Requirements Engineering Conference. pp. 381–385.

Ha, E., Wagner, D., 2013. Do android users write about electric sheep? Examining consumer reviews in google play. In: IEEE Consumer Communications and Networking Conference. pp. 149–157.

Hadi, M.A., Fard, F.H., 2020. AOBTM: Adaptive online biterm topic modeling for version sensitive short-texts analysis. In: IEEE International Conference on Software Maintenance and Evolution. pp. 593–604.

Haering, M., Stanik, C., Maalej, W., 2021. Automatically matching bug reports with related app reviews. In: IEEE/ACM International Conference on Software Engineering. pp. 970–981.

Haggag, O., 2022. Better identifying and addressing diverse issues in mhealth and emerging apps using user reviews. In: International Conference on Evaluation and Assessment in Software Engineering 2022. pp. 329–335.

Harkous, H., Peddinti, S.T., Khandelwal, R., Srivastava, A., Taft, N., 2022. Hark: A deep learning system for navigating privacy feedback at scale. In: IEEE Symposium on Security and Privacy. pp. 2469–2486.

Hassan, S., Bezemer, C.P., Hassan, A.E., 2020. Studying bad updates of top free-to-download apps in the google play store. IEEE Trans. Softw. Eng. 46 (7), 773–793.

Hassan, S., Li, H., Hassan, A.E., 2022. On the importance of performing app analysis within peer groups. In: IEEE International Conference on Software Analysis, Evolution and Reengineering. pp. 890–901.

Hassan, S., Tantithamthavorn, C., Bezemer, C.P., Hassan, A.E., 2018. Studying the dialogue between users and developers of free apps in the google play store. Empir. Softw. Eng. 23 (3), 1275–1312.

Hatamian, M., Serna, J., Rannenberg, K., 2019. Revealing the unrevealed: Mining smartphone users privacy perception on app markets. Comput. Secur. 83, 332–353.

He, R., Lee, W.S., Ng, H.T., Dahlmeier, D., 2017. An unsupervised neural attention model for aspect extraction. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 388–397.

Higashi, K., Nakagawa, H., Tsuchiya, T., 2018. Improvement of user review classification using keyword expansion. In: International Conference on Software Engineering and Knowledge Engineering. pp. 125–124.

Hoon, L., Stojmenović, M., Vasa, R., Farrell, G., 2016. Spreading word: Author frequency of app user reviews. In: Australian Conference on Computer-Human Interaction. pp. 643–645.

Hoon, L., Vasa, R., Martino, G.Y., Schneider, J.G., Mouzakis, K., 2013. Awesome! conveying satisfaction on the app store. In: Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration. pp. 229–232.

Hoon, L., Vasa, R., Schneider, J.G., Mouzakis, K., 2012. A preliminary analysis of vocabulary in mobile app user reviews. In: Australian Computer-Human Interaction Conference. pp. 245–248.

Hu, X., Guo, Y., Lu, J., Zhu, Z., Li, C., Ge, J., Huang, L., Luo, B., 2022. Lighting up supervised learning in user review-based code localization: dataset and benchmark. In: ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering. pp. 533–545.

Hu, H., Wang, S., Bezemer, C.P., Hassan, A.E., 2019a. Studying the consistency of star ratings and reviews of popular free hybrid android and iOS apps. Empir. Softw. Eng. 24, 7–32.

Hu, Y., Wang, H., Ji, T., Xiao, X., Luo, X., Gao, P., Guo, Y., 2021. Champ: Characterizing undesired app behaviors from user comments based on market policies. In: IEEE/ACM International Conference on Software Engineering. pp. 933–945.

Hu, Y., Wang, H., Li, L., Guo, Y., Xu, G., He, R., 2019b. Want to earn a few extra bucks? a first look at money-making apps. In: IEEE International Conference on Software Analysis, Evolution and Reengineering. pp. 332–343.

Hu, Y., Wang, H., Zhou, Y., Guo, Y., Li, L., Luo, B., Xu, F., 2019c. Dating with scambots: Understanding the ecosystem of fraudulent dating applications. IEEE Trans. Dependable Secure Comput. 18 (3), 1033–1050.

Huebner, J., Frey, R.M., Ammendola, C., Fleisch, E., Ilic, A., 2018. What people like in mobile finance apps: An analysis of user reviews. In: International Conference on Mobile and Ubiquitous Multimedia. pp. 293–304.

Huebner, J., Girardello, A., Sliz, O., Fleisch, E., Ilic, A., 2020. What people focus on when reviewing your app-an analysis across app categories. IEEE Softw. 38 (3), 96–105.

Iacob, C., Harrison, R., 2013. Retrieving and analyzing mobile apps feature requests from online reviews. In: Working Conference on Mining Software Repositories. pp. 41–44.

Iacob, C., Veerappa, V., Harrison, R., 2013. What are you complaining about?: A study of online reviews of mobile applications. In: International BCS Human Computer Interaction Conference. pp. 1–6.

Ikram, M., Kaafar, M.A., 2017. A first look at mobile ad-blocking apps. In: IEEE International Symposium on Network Computing and Applications. pp. 1–8.

Indurkhya, N., Damerau, F.J., 2010. Handbook of Natural Language Processing, vol. 2, CRC Press.

Islam, M.R., Zibran, M.F., 2018. SentiStrength-SE: Exploiting domain specificity for improved sentiment analysis in software engineering text. J. Syst. Softw. 145, 125–146.

Jansen, S., 2020. A focus area maturity model for software ecosystem governance. Inf. Softw. Technol. 118, 106219.
Jha, N., Mahmoud, A., 2018. Using frame semantics for classifying and summarizing application store reviews. Empir. Softw. Eng. 23, 3734–3767.
Jha, N., Mahmoud, A., 2019. Mining non-functional requirements from app store reviews. Empir. Softw. Eng. 24, 3659–3695.
Jiang, W., Ruan, H., Zhang, L., Lew, P., Jiang, J., 2014. For user-driven software evolution: Requirements elicitation derived from mining online reviews. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 584–595.
Johann, T., Stanik, C., Alizadeh B., A.M., Maalej, W., 2017. SAFE: A simple approach for feature extraction from app descriptions and app reviews. In: IEEE International Requirements Engineering Conference. pp. 21–30.
Keele, S., et al., 2007. Guidelines for Performing Systematic Literature Reviews in Software Engineering. Technical Report, Technical report, ver. 2.3 ebse technical report. ebse.
Keertipati, S., Savarimuthu, B.T.R., Licorish, S.A., 2016. Approaches for prioritizing feature improvements extracted from app reviews. In: International Conference on Evaluation and Assessment in Software Engineering. pp. 1–6.
Khalajzadeh, H., Shahin, M., Obie, H.O., Agrawal, P., Grundy, J., 2022. Supporting developers in addressing human-centric issues in mobile apps. IEEE Trans. Softw. Eng..
Khalid, H., 2013. On identifying user complaints of iOS apps. In: 2013 35th International Conference on Software Engineering. pp. 1474–1476.
Khalid, H., Nagappan, M., Hassan, A.E., 2016. Examining the relationship between FindBugs warnings and app ratings. IEEE Softw. 33 (4), 34–39.
Khalid, H., Nagappan, M., Shihab, E., Hassan, A.E., 2014. Prioritizing the devices to test your app on: A case study of android game apps. In: ACM SIGSOFT International Symposium on Foundations of Software Engineering. pp. 610–620.
Khalid, H., Shihab, E., Nagappan, M., Hassan, A.E., 2015. What do mobile app users complain about? IEEE Softw. 32 (3), 70–77.
Kong, D., Cen, L., Jin, H., 2015. Autoreb: Automatically understanding the review-to-behavior fidelity in android applications. In: ACM SIGSAC Conference on Computer and Communications Security. pp. 530–541.
Kübler, R., Pauwels, K., Yildirim, G., Fandrich, T., 2018. App popularity: Where in the world are consumers most sensitive to price and user ratings? J. Market. 82 (5), 20–44.
Kuehnhausen, M., Frost, V.S., 2013. Trusting smartphone apps? To install or not to install, that is the question. In: IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support. pp. 30–37.
Kurtanović, Z., Maalej, W., 2017. Mining user rationale from software reviews. In: IEEE International Requirements Engineering Conference. pp. 61–70.
Kurtanović, Z., Maalej, W., 2018. On user rationale in software engineering. Requir. Eng. 23, 357–379.
Lemon, K.N., Verhoef, P.C., 2016. Understanding customer experience throughout the customer journey. J. Market. 80 (6), 69–96.
Li, S., Caverlee, J., Niu, W., Kaghazgaran, P., 2017. Crowdsourced app review manipulation. In: International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1137–1140.
Li, H., Lin, R., Hong, R., Ge, Y., 2015. Generative models for mining latent aspects and their ratings from short reviews. In: IEEE International Conference on Data Mining. pp. 241–250.
Li, R., Obie, H.O., Khalajzadeh, H., 2021. An exploratory analysis of human-centric issues in parking solutions using surveys and mobile parking app reviews. In: Australian Conference on Human-Computer Interaction. pp. 26–37.
Licorish, S.A., Savarimuthu, B.T.R., Keertipati, S., 2017. Attributes that predict which features to fix: Lessons for app store mining. In: International Conference on Evaluation and Assessment in Software Engineering. pp. 108–117.
Lin, D., Bezemer, C.P., Zou, Y., Hassan, A.E., 2019. An empirical study of game reviews on the steam platform. Empir. Softw. Eng. 24, 170–207.
Lin, B., Cassee, N., Serebrenik, A., Bavota, G., Novielli, N., Lanza, M., 2022. Opinion mining for software development: A systematic literature review. ACM Trans. Softw. Eng. Methodol. 31 (3), 1–41.
Lin, B., Zampetti, F., Bavota, G., Di Penta, M., Lanza, M., Oliveto, R., 2018. Sentiment analysis for software engineering: How far can we go? In: International Conference on Software Engineering. pp. 94–104.
Linares-Vásquez, M., Bavota, G., Bernal-Cárdenas, C., Di Penta, M., Oliveto, R., Poshyvanyk, D., 2013. Api change and fault proneness: A threat to the success of android apps. In: Joint Meeting on Foundations of Software Engineering. pp. 477–487.
Liu, P., Li, L., Zhao, Y., Sun, X., Grundy, J., 2020. Androzoooopen: Collecting large-scale open source android apps for the research community. In: International Conference on Mining Software Repositories. pp. 548–552.
Liu, M., Wu, C., Zhao, X.-N., Lin, C.Y., Wang, X.L., 2015. App relationship calculation: An iterative process. IEEE Trans. Knowl. Data Eng. 27 (8), 2049–2063.
Lu, M., Liang, P., 2017. Automatic classification of non-functional requirements from augmented app user reviews. In: International Conference on Evaluation and Assessment in Software Engineering. pp. 344–353.
Luiz, W., Viegas, F., Alencar, R., Mourão, F., Salles, T., Carvalho, D., Gonçalves, M.A., Rocha, L., 2018. A feature-oriented sentiment rating for mobile app reviews. In: World Wide Web Conference. pp. 1909–1918.
Luna, J.M., Fournier-Viger, P., Ventura, S., 2019. Frequent itemset mining: A 25 years review. Wiley Interdiscip. Rev.: Data Min. Knowl. Discov. 9 (6), e1329.
Maalej, W., Kurtanović, Z., Nabil, H., Stanik, C., 2016. On the automatic classification of app reviews. Requir. Eng. 21, 311–331.
Maalej, W., Nabil, H., 2015. Bug report, feature request, or simply praise? On automatically classifying app reviews. In: IEEE International Requirements Engineering Conference. pp. 116–125.
Maalej, W., Nayebi, M., Johann, T., Ruhe, G., 2015. Toward data-driven requirements engineering. IEEE Softw. 33 (1), 48–54.
Malgaonkar, S., Licorish, S.A., Savarimuthu, B.T.R., 2022. Prioritizing user concerns in app reviews–A study of requests for new features, enhancements and bug fixes. Inf. Softw. Technol. 144, 106798.
Malik, H., Shakshuki, E.M., Yoo, W.S., 2020. Comparing mobile apps by identifying 'hot'features. Future Gener. Comput. Syst. 107, 659–669.
Man, Y., Gao, C., Lyu, M.R., Jiang, J., 2016. Experience report: Understanding cross-platform app issues from user reviews. In: IEEE International Symposium on Software Reliability Engineering. pp. 138–149.
Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J.R., Bethard, S., McClosky, D., 2014. The stanford CoreNLP natural language processing toolkit. In: Annual Meeting of the Association for Computational Linguistics: System Demonstrations. pp. 55–60.
Martens, D., Maalej, W., 2019a. Release early, release often, and watch your users' emotions: Lessons from emotional patterns. IEEE Softw. 36 (5), 32–37.
Martens, D., Maalej, W., 2019b. Towards understanding and detecting fake reviews in app stores. Empir. Softw. Eng. 24 (6), 3316–3355.
Martin, W., Harman, M., Jia, Y., Sarro, F., Zhang, Y., 2015. The app sampling problem for app store mining. In: IEEE/ACM Working Conference on Mining Software Repositories. pp. 123–133.
Martin, W., Sarro, F., Jia, Y., Zhang, Y., Harman, M., 2017. A survey of app store analysis for software engineering. IEEE Trans. Softw. Eng. 43 (9), 817–847.
Mathews, C., Ye, K., Grozdanovski, J., Marinelli, M., Zhong, K., Khalajzadeh, H., Obie, H., Grundy, J., 2021. AH-CID: A tool to automatically detect human-centric issues in app. In: International Conference on Software Technologies. pp. 386–397.
McIlroy, S., Ali, N., Khalid, H., E. Hassan, A., 2016. Analyzing and automatically labelling the types of user issues that are raised in mobile app reviews. Empir. Softw. Eng. 21, 1067–1106.
McIlroy, S., Shang, W., Ali, N., Hassan, A.E., 2017. Is it worth responding to reviews? studying the top free apps in google play. IEEE Softw. 34 (3), 64–71.
Mekala, R.R., Irfan, A., Groen, E.C., Porter, A., Lindvall, M., 2021. Classifying user requirements from online feedback in small dataset environments using deep learning. In: IEEE International Requirements Engineering Conference. pp. 139–149.
Morales-Ramirez, I., Munante, D., Kifetew, F., Perini, A., Susi, A., Siena, A., 2017. Exploiting user feedback in tool-supported multi-criteria requirements prioritization. In: IEEE International Requirements Engineering Conference. pp. 424–429.
Morales-Ramirez, I., Perini, A., Guizzardi, R.S., 2015. An ontology of online user feedback in software engineering. Appl. Ontol. 10 (3–4), 297–330.
Nayebi, M., Cho, H., Ruhe, G., 2018. App store mining is not enough for app improvement. Empir. Softw. Eng. 23, 2764–2794.
Nema, P., Anthonysamy, P., Taft, N., Peddinti, S.T., 2022. Analyzing user perspectives on mobile app privacy at scale. In: International Conference on Software Engineering. pp. 112–124.
Nguyen, D.C., Derr, E., Backes, M., Bugiel, S., 2019. Short text, large effect: Measuring the impact of user reviews on android app security & privacy. In: IEEE Symposium on Security and Privacy. pp. 555–569.
Noei, E., Da Costa, D.A., Zou, Y., 2018. Winning the app production rally. In: ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. pp. 283–294.
Noei, E., Lyons, K., 2019. A survey of utilizing user-reviews posted on google play store. In: Annual International Conference on Computer Science and Software Engineering. pp. 54–63.
Noei, E., Syer, M.D., Zou, Y., Hassan, A.E., Keivanloo, I., 2017. A study of the relation of mobile device attributes with the user-perceived quality of android apps. Empir. Softw. Eng. 22 (6), 3088–3116.
Noei, E., Zhang, F., Wang, S., Zou, Y., 2019. Towards prioritizing user-related issue reports of mobile applications. Empir. Softw. Eng. 24, 1964–1996.
Noei, E., Zhang, F., Zou, Y., 2021. Too many user-reviews! what should app developers look at first? IEEE Trans. Softw. Eng. 47 (2), 367–378.
Obie, H.O., Ilekura, I., Du, H., Shahin, M., Grundy, J., Li, L., Whittle, J., Turhan, B., 2022. On the violation of honesty in mobile apps: automated detection and categories. In: International Conference on Mining Software Repositories. pp. 321–332.
Oehri, E., Guzman, E., 2020. Same same but different: Finding similar user feedback across multiple platforms and languages. In: IEEE International Requirements Engineering Conference. pp. 44–54.
Ouyang, Y., Guo, B., Lu, X., Han, Q., Guo, T., Yu, Z., 2018. Competitivebike: Competitive analysis and popularity prediction of bike-sharing apps using multi-source data. IEEE Trans. Mob. Comput. 18 (8), 1760–1773.
Pagano, D., Bruegge, B., 2013. User involvement in software evolution practice: A case study. In: International Conference on Software Engineering. pp. 953–962.
Pagano, D., Maalej, W., 2013. User feedback in the appstore: An empirical study. In: IEEE International Requirements Engineering Conference. pp. 125–134.

Palomba, F., Linares-Vásquez, M., Bavota, G., Oliveto, R., Di Penta, M., Poshyvanyk, D., De Lucia, A., 2015. User reviews matter! tracking crowdsourced reviews to support evolution of successful apps. In: IEEE International Conference on Software Maintenance and Evolution. pp. 291–300.

Palomba, F., Salza, P., Ciurumelea, A., Panichella, S., Gall, H., Ferrucci, F., De Lucia, A., 2017. Recommending and localizing change requests for mobile apps based on user reviews. In: IEEE/ACM International Conference on Software Engineering. pp. 106–117.

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: IEEE International Conference on Software Maintenance and Evolution. pp. 281–290.

Panthum, T., Senivongse, T., 2021. Generating functional requirements based on classification of mobile application user reviews. In: IEEE/ACIS International Conference on Software Engineering Research, Management and Applications. pp. 15–20.

Park, D.H., Liu, M., Zhai, C., Wang, H., 2015. Leveraging user reviews to improve accuracy for mobile app retrieval. In: International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 533–542.

Petz, G., Karpowicz, M., Fürschuß, H., Auinger, A., Střítesk̀y, V., Holzinger, A., 2013. Opinion mining on the web 2.0–characteristics of user generated content and their impacts. In: International Workshop on Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data. pp. 35–46.

Phong, M.V., Nguyen, T.T., Pham, H.V., Nguyen, T.T., 2015. Mining user opinions in mobile app reviews: A keyword-based approach. In: IEEE/ACM International Conference on Automated Software Engineering. pp. 749–759.

Rahman, M., Hernandez, N., Carbunar, B., Chau, D.H., 2021. Towards de-anonymization of google play search rank fraud. IEEE Trans. Knowl. Data Eng. 33 (11), 3648–3661.

Rahman, M., Rahman, M., Carbunar, B., Chau, D.H., 2017. Search rank fraud and malware detection in google play. IEEE Trans. Knowl. Data Eng. 29 (6), 1329–1342.

Rocha, L., Mourão, F., Silveira, T., Chaves, R., Sá, G., Teixeira, F., Vieira, R., Ferreira, R., 2015. SACI: Sentiment analysis by collective inspection on social media content. J. Web Semant. 34, 27–39.

S¨anger, M., Leser, U., Klinger, R., 2017. Fine-grained opinion mining from mobile app reviews with word embedding features. In: International Conference on Applications of Natural Language to Information Systems. pp. 3–14.

Sayagh, M., Kerzazi, N., Adams, B., Petrillo, F., 2018. Software configuration engineering in practice interviews, survey, and systematic literature review. IEEE Trans. Softw. Eng. 46 (6), 646–673.

Scalabrino, S., Bavota, G., Russo, B., Di Penta, M., Oliveto, R., 2019. Listening to the crowd for the release planning of mobile apps. IEEE Trans. Softw. Eng. 45 (1), 68–86.

Scherr, S.A., Elberzhager, F., Holl, K., 2017. An automated feedback-based approach to support mobile app development. In: Euromicro Conference on Software Engineering and Advanced Applications. pp. 44–51.

Shah, F.A., Sirts, K., Pfahl, D., 2019. Simplifying the classification of app reviews using only lexical features. In: International Conference on Software Technologies. pp. 173–193.

Singh, A., Arun, A., Malhotra, P., Desur, P., Jain, A., Chau, D.H., Kumaraguru, P., 2022. Erasing labor with labor: Dark patterns and lockstep behaviors on google play. In: ACM Conference on Hypertext and Social Media. pp. 186–191.

Srisopha, K., Link, D., Boehm, B., 2021. How should developers respond to app reviews? Features predicting the success of developer responses. In: Evaluation and Assessment in Software Engineering. pp. 119–128.

Srisopha, K., Link, D., Swami, D., Boehm, B., 2020a. Learning features that predict developer responses for ios app store reviews. In: ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. pp. 1–11.

Srisopha, K., Phonsom, C., Lin, K., Boehm, B., 2019. Same app, different countries: A preliminary user reviews study on most downloaded ios apps. In: IEEE International Conference on Software Maintenance and Evolution. pp. 76–80.

Srisopha, K., Swami, D., Link, D., Boehm, B., 2020b. How features in ios app store reviews can predict developer responses. In: Evaluation and Assessment in Software Engineering. pp. 336–341.

Su'a, T., Licorish, S.A., Savarimuthu, B.T.R., Langlotz, T., 2017. Quickreview: A novel data-driven mobile user interface for reporting problematic app features. In: International Conference on Intelligent User Interfaces. pp. 517–522.

Sun, K., Gao, H., Kuang, H., Ma, X., Rong, G., Shao, D., Zhang, H., 2021. Exploiting the unique expression for improved sentiment analysis in software engineering text. In: IEEE/ACM International Conference on Program Comprehension. pp. 149–159.

Taba, S.E.S., Keivanloo, I., Zou, Y., Ng, J., Ng, T., 2014. An exploratory study on the relation between user interface complexity and the perceived quality. In: International Conference on Web Engineering. pp. 370–379.

Tao, C., Guo, H., Huang, Z., 2020. Identifying security issues for mobile applications based on user review summarization. Inf. Softw. Technol. 122, 106290.

Tavakoli, M., Zhao, L., Heydari, A., Nenadić, G., 2018. Extracting useful software development information from mobile application reviews: A survey of intelligent mining techniques and tools. Expert Syst. Appl. 113, 186–199.

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

Tian, Y., Nagappan, M., Lo, D., Hassan, A.E., 2015. What are the characteristics of high-rated apps? A case study on free android applications. In: IEEE International Conference on Software Maintenance and Evolution. pp. 301–310.

Tushev, M., Ebrahimi, F., Mahmoud, A., 2022. Domain-specific analysis of mobile app reviews using keyword-assisted topic models. In: International Conference on Software Engineering. pp. 762–773.

Uddin, M.K., He, Q., Han, J., Chua, C., 2020a. App competition matters: How to identify your competitor apps? In: IEEE International Conference on Services Computing. pp. 370–377.

Uddin, M.K., Qiang, H., Jun, H., Caslon, C., 2020b. Feature recommendation by mining updates and user feedback from competitor apps. In: International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services. pp. 18–28.

Van Oordt, S., Guzman, E., 2021. On the role of user feedback in software evolution: A practitioners' perspective. In: 2021 IEEE 29th International Requirements Engineering Conference. RE, pp. 221–232.

van Vliet, M., Groen, E.C., Dalpiaz, F., Brinkkemper, S., 2020. Identifying and classifying user requirements in online feedback via crowdsourcing. In: International Conference on Requirements Engineering: Foundation for Software Quality. pp. 143–159.

Vasa, R., Hoon, L., Mouzakis, K., Noguchi, A., 2012. A preliminary analysis of mobile app user reviews. In: Australian Computer-Human Interaction Conference. pp. 241–244.

Villarroel, L., Bavota, G., Russo, B., Oliveto, R., Di Penta, M., 2016. Release planning of mobile apps based on user reviews. In: International Conference on Software Engineering. pp. 14–24.

Vu, P.M., Pham, H.V., Nguyen, T.T., Nguyen, T.T., 2016. Phrase-based extraction of user opinions in mobile app reviews. In: IEEE/ACM International Conference on Automated Software Engineering. pp. 726–731.

Wang, T., Liang, P., Lu, M., 2018a. What aspects do non-functional requirements in app user reviews describe? An exploratory and comparative study. In: Asia-Pacific Software Engineering Conference. pp. 494–503.

Wang, C., Liu, T., Liang, P., Daneva, M., van Sinderen, M., 2021. The role of user reviews in app updates: A preliminary investigation on app release notes. In: Asia-Pacific Software Engineering Conference. pp. 520–525.

Wang, L., Wang, H., Luo, X., Zhang, T., Wang, S., Liu, X., Demystifying "removed reviews" in iOS app store. In: ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering. pp. 1489–1499.

Wang, R., Wang, Z., Tang, B., Zhao, L., Wang, L., 2020. Smartpi: Understanding permission implications of android apps from user reviews. IEEE Trans. Mob. Comput. 19 (12), 2933–2945.

Wang, Y., Wang, J., Zhang, H., Ming, X., Shi, L., Wang, Q., 2022a. Where is your app frustrating users? In: International Conference on Software Engineering. pp. 2427–2439.

Wang, X., Zhang, W., Lai, S., Ye, C., Zhou, H., 2022b. The use of pretrained model for matching app reviews and bug reports. In: IEEE International Conference on Software Quality, Reliability and Security. pp. 242–251.

Wang, C., Zhang, F., Liang, P., Daneva, M., Van Sinderen, M., 201b8. Can app changelogs improve requirements classification from app reviews? an exploratory study. In: ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. pp. 1–4.

Wei, L., Liu, Y., Cheung, S.C., 2017. Oasis: prioritizing static analysis warnings for android apps based on app user reviews. In: Joint Meeting on Foundations of Software Engineering. pp. 672–682.

Williams, G., Mahmoud, A., 2018. Modeling user concerns in the app store: A case study on the rise and fall of yik yak. In: IEEE International Requirements Engineering Conference. pp. 64–75.

Wohlin, C., 2014. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: International Conference on Evaluation and Assessment in Software Engineering. pp. 1–10.

Wu, H., Deng, W., Niu, X., Nie, C., 2021. Identifying key features from app user reviews. In: IEEE/ACM International Conference on Software Engineering. pp. 922–932.

Yadav, A., Sharma, R., Fard, F.H., 2020. A semantic-based framework for analyzing app users' feedback. In: IEEE International Conference on Software Analysis, Evolution and Reengineering. pp. 572–576.

Yang, A.Z., Hassan, S., Zou, Y., Hassan, A.E., 2022. An empirical study on release notes patterns of popular apps in the google play store. Empir. Softw. Eng. 27 (2), 55.

Yang, H., Liang, P., 2015. Identification and classification of requirements from app user reviews.. In: International Conference on Software Engineering and Knowledge Engineering. pp. 7–12.

Yao, Y., Zhao, W.X., Wang, Y., Tong, H., Xu, F., Lu, J., 2017. Version-aware rating prediction for mobile app recommendation. ACM Trans. Inf. Syst. 35 (4), 1–33.

Yin, H., Pfahl, D., 2018. The OIRE method - overview and initial validation. In: Asia-Pacific Software Engineering Conference. pp. 1–10.

Yu, L., Wang, H., Luo, X., Zhang, T., Liu, K., Chen, J., Zhou, H., Tang, Y., Xiao, X., 2022. Towards automatically localizing function errors in mobile apps with user reviews. IEEE Trans. Softw. Eng..

Zhang, T., Chen, J., Zhan, X., Luo, X., Lo, D., Jiang, H., 2021. Where2Change: Change request localization for app reviews. IEEE Trans. Softw. Eng. 47 (11), 2590–2616.

Zhang, Y., Guo, B., Liu, J., Guo, T., Ouyang, Y., Yu, Z., 2020a. Which app is going to die? A framework for app survival prediction with multitask learning. IEEE Trans. Mob. Comput. 21 (2), 728–739.

Zhang, Y., Liu, J., Guo, B., Wang, Z., Liang, Y., Yu, Z., 2020b. App popularity prediction by incorporating time-varying hierarchical interactions. IEEE Trans. Mob. Comput. 21 (5), 1566–1579.
Zhang, L., Tian, J.H., Jiang, J., Liu, Y.J., Pu, M.Y., Yue, T., 2018. Empirical research in software engineering—A literature survey. J. Comput. Sci. Tech. 33, 876–899.
Zhang, J., Wang, Y., Xie, T., 2019. Software feature refinement prioritization based on online user review mining. Inf. Softw. Technol. 108, 30–34.
Zhou, Y., Su, Y., Chen, T., Huang, Z., Gall, H., Panichella, S., 2021. User review-based change file localization for mobile applications. IEEE Trans. Softw. Eng. 47 (12), 2755–2770.
Zhu, H., Liu, C., Ge, Y., Xiong, H., Chen, E., 2014. Popularity modeling for mobile apps: A sequential approach. IEEE Trans. Cybern. 45 (7), 1303–1314.
Zhu, H., Xiong, H., Ge, Y., Chen, E., 2015. Discovery of ranking fraud for mobile apps. IEEE Trans. Knowl. Data Eng. 27 (1), 74–87.

Xiaohui Wang received the B.Econ degree from Zhongnan University of Economics and Law, in 2021. She is currently working towards the postgraduate degree with the School of Computer Science and Engineering, Macau University of Science and Technology (MUST), under the supervision of Prof. Tao Zhang. Her current research interests include software engineering and natural language processing.

Tao Zhang received the BS degree in automation, the MEng degree in software engineering from Northeastern University, China, and the Ph.D. degree in computer science from the University of Seoul, South Korea. After that, he spent one year with the Hong Kong Polytechnic University as a postdoctoral research fellow. Currently, he is an associate professor with the School of Computer Science and Engineering, Macau University of Science and Technology (MUST). Before joining MUST, he was the faculty member of Harbin Engineering University and Nanjing University of Posts and Telecommunications, China. His current research interests include AI for software engineering and mobile software security. He published more than 90 high-quality papers at renowned software engineering and security journals and conferences such as TSE, TOSEM, TIFS, TDSC, TR, ICSE, ESEC/FSE, ASE, etc. He is a senior member of IEEE and ACM. He is currently serving as an associate editor for TSE and JSS. He also served as the General Chair of SANER 2023 and the Program Co-Chair of Internetware 2024.

Youshuai Tan received the BEng degree from Harbin Engineering University, in 2021. He is currently working towards the postgraduate degree with the School of Computer Science and Engineering, Macau University of Science and Technology (MUST), under the supervision of Prof. Tao Zhang. His works have been published in JSS and TR. His current research interests include software engineering and natural language processing.

Weiyi Shang is an Associate Professor at the University of Waterloo. His research interests include AIOps, big bata software engineering, software log analytics and software performance engineering. He serves as a Steering committee member of the SPEC Research Group. He is ranked top worldwide SE research stars in a recent bibliometrics assessment of software engineering scholars. He is a recipient of various premium awards, including the SIGSOFT Distinguished paper award at ICSE 2013 and ICSE 2020, best paper award at WCRE 2011 and the Distinguished reviewer award for the Empirical software Engineering journal. His research has been adopted by industrial collaborators (e.g., BlackBerry and Ericsson) to improve the quality and performance of their software systems that are used by millions of users worldwide. Contact him at wshang@uwaterloo.ca; https://ece.uwaterloo.ca/~wwshang/.

Yao Li received the M.E. degree from Shantou University, in 2021. He is working towards the Ph.D. degree with the School of Computer Science and Engineering, Macau University of Science and Technology (MUST), under the supervision of Prof. Tao Zhang. His research interests lie in software engineering and malware detection.