The Impact of Feature Exploitation and Exploration on Mobile Application Evolution and Success

Shadi Shuraida (shadi.shuraida@teluq.ca), Qiang Gao, Hani Safadi, and Radhika Jain.

ABSTRACT

Mobile device applications are the largest segment of IS with an estimated 5 billion users. Yet despite their widespread and growing use, there is little research examining how these mobile applications evolve with each new release update. To ensure market success, developers need to satisfy their user base by incorporating users' reviews and feedback on the one hand and exploring new features and content that allows them to stay competitive on the other. Drawing on the organizational learning and innovation literature, the findings of the present study suggest that a mix of these two activities of exploitation and exploration in consequent app updates is likely to result in the app's success. We further contribute to this body of work by examining the influence of users' online review characteristics on exploitation and exploration activities in app development. The findings suggest that users' convergence on similar issues (review concurrence) is likely to favor an orientation towards exploitation over exploration activities, while the number of user reviews (review volume) has a curvilinear relationship with it.

Keywords: mobile application development, mobile application evolution, online user reviews, technological innovation, exploitation, exploration.

The Impact of Feature Exploitation and Exploration on Mobile Application Evolution and Success

1 Introduction

In mobile platform ecosystems, app evolution is defined as the system's adaptability to serve new purposes and exploit emerging possibilities (de Weck et al., 2011). App evolution is characterized by low development cost and high imitation rate (Wang et al., 2018), and often takes place in an extreme environment characterized by "compressed development schedules, emerging technologies, criticality of performance, [and] insufficient requirements specifications" (Bragge & Merisalo-Rantanen, 2002, p. 197), leading to short app revision cycles often measured in days. While the majority of top free apps in Google Play Store are updated every six weeks (Yang et al., 2022), many app developers often release daily updates of their applications (Openja et al., 2020). Previous empirical studies have found that frequent faster version updates that respond to users' feedback have a significant impact on their apps' quality and market success (Allon et al., 2022; Agarwal & Tiwana, 2015; Lee & Raghu, 2014; Palomba et al., 2018; Soh & Grover, 2020; Tiwana, 2015; Zhou et al., 2018). However, besides the speed and frequency of updates, a less studied aspect that defines the evolution of mobile apps is the content of the update itself, i.e., how the app's features change from one update to the next.

In the competitive environment of mobile app development, developers need to make constant decisions about how to evolve their apps with each update release. This is no simple feat given that the design landscape for mobile apps is immense (Brunswicker et al., 2019) with near-infinite possibilities of features that developers can incorporate (Boudreau, 2012). The previous literature suggests that the majority of developers draw on users' online reviews as a rich source of information to evolve their apps (Palomba et al., 2018), and that responding to these reviews increases the app’s market success (Palomba et al., 2018; Zhou et al., 2018). Such an approach is consistent with traditional IS development approaches that emphasize meeting users’ requirements (Fernandez et al., 2017).

On the other hand, app development remains a creative endeavor in which developers often must experiment with novel features that differ substantively from existing features to differentiate their apps from rival offerings (Agarwal & Tiwana, 2015; Boudreau, 2012; McIlroy et al., 2016; Reeck et al., 2023; Soh & Grover, 2022) and explore opportunities to expand beyond their existing user base. To do so, app developers rely on various internal (Al-Subaihin et al., 2021; Pagano & Bruegge, 2013) and external sources for evolving their apps (Al-Subaihin et al., 2021; Liu et al, 2023; Tiwana, 2015). Yet, the outcome of such experimentation with novel features that users did not ask for is often uncertain for the app’s success (Boudreau, 2012).

While a focus on exploitative activities such as incorporating user-driven stated features is likely to increase app quality and market return, it may also stifle the introduction of innovative features that are necessary for adaptation and differentiation from rivals (Al-Subaihin et al., 2021; Andriopoulos & Lewis, 2009; Levinthal & March, 1993). Hence, app developers need to divide their limited attention and resources between two activities: the first is the directed search to leverage and exploit the existing knowledge reflected by users’ reviews, and the second is exploration reflected by experimentation with new content and features (Benner & Tushman, 2002; McGrath, 2001, p. 118). However, little research has examined how app developers navigate between exploitation and exploration activities, and how this impacts their app’s success.

The present study fills this gap by examining the dynamics of exploitation and exploration activities within the rapidly evolving and highly competitive mobile app development environment. More specifically, it addresses the following research questions: (1) *what is the impact of developers' exploitation and exploration activities on mobile app success, and (2) what is the role of users' review feedback on developers' exploitation and exploration activities?* To address the first question, we draw on the organizational learning and innovation literature (March, 1991) to propose that developers' mix of exploitation and exploration activities positively impacts an app's success. We then draw on the information processing literature (Simon, 1956; 1973) and propose two user review characteristics that influence developers' exploitation and exploration app development activities. The first is 1) review concurrence which is the magnitude of users' convergence on similar issues, and the second is 2) review volume which is the number of different reviews following an app release. The proposed research model is tested and validated using a dataset of 1,046,553 online user reviews of 119 apps.

2 Theory Development

Apps' continuous development is synonymous with their success if they are to keep up with the competition and survive (Allon et al., 2022; Agarwal & Tiwana, 2015). Previous research has contributed to our knowledge about mobile app release evolution in terms of frequency and speed; however, little research has examined the effect of app update content and source on app success (e.g., Palomba et al., 2018; Zhou et al., 2018). While the few studies examining the content of app updates have shown that addressing users' feedback in consecutive update releases increases the likelihood of app success (Palomba et al., 2018; Zhou et al., 2018), other studies have argued that mobile applications are creative innovations that also need to explore new features (Boudreau, 2012; Ye & Kankanhalli, 2020; Yin et al., 2014) and adapt to market forces to succeed (Agarwal & Tiwana, 2015; Soh & Grover, 2020).

Indeed, prior literature on app development highlights the importance of exploiting users' reviews for satisfying and retaining the existing user base (Lee & Raghu, 2014; Nayebi et al., 2016; Palomba et al., 2018; Zhou et al., 2018). However, few studies in app and product development (Al-Subaihin et al., 2021; Greve, 2007; Jiang et al., 2019) as well as broader work in organizational learning, caution against over exploitation of "things already known" and suggest the need to also explore new possibilties (Levinthal & March, 1993, p. 105; March, 1991; O'Reilly & Tushman, 2013).

We draw on the organizational learning literature (March, 1991) to propose that developers' app update activities consist of a mix of exploitation and exploration activities. Further, while developers need to use a mix of both activities to be successful in the marketplace, we also argue that they are faced with a tradeoff decision between these two activities given that they compete over limited resources. We then focus on the role of external feedback in shaping this decision. Specifically, we argue that app developers are rationally bounded, and build on the information processing literature (Simon, 1956) to suggest that the extent of their exploitation and exploration activities is likely to be influenced by the nature of the information they receive from users' reviews. Specifically, we propose that two characteristics of users' online reviews – review concurrence and review volume – influence app developers' orientation towards exploitation vis-à-vis exploration activities, which in turn influence the app's success. The research model is presented below in Figure 1 and developed in the following sections.

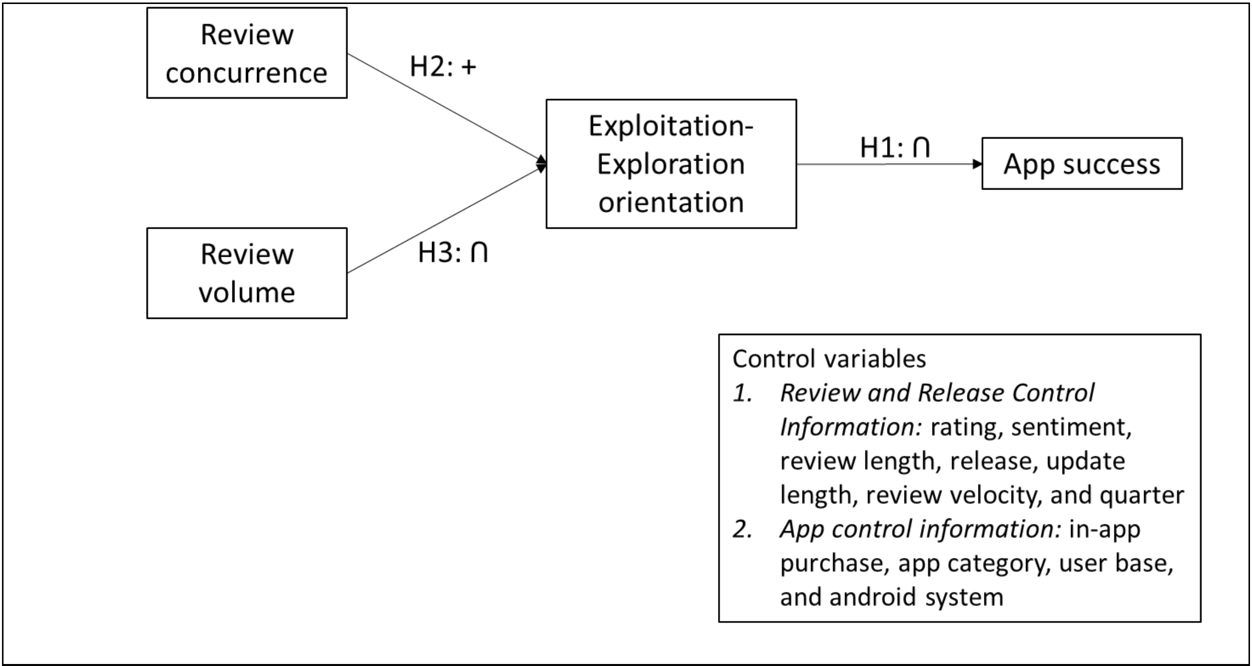
Review concurrence
H2: +
Review volume
H3: ∩
Exploitation-Exploration orientation
H1: ∩
App success
Control variables
1. Review and Release Control Information: rating, sentiment, review length, release, update length, review velocity, and quarter
2. App control information: in-app purchase, app category, user base, and android system

Figure 1: Research Model

2.1 The Notions of Exploitation and Exploration

In his seminal article about organizational learning, March (1991) put forth the notion that individuals and organizations conduct two forms of activities that could explain the "evolutionary models of organizational forms and technologies" (p. 72): exploitation and exploration. Despite the various definitions found in the literature, there is consensus amongst scholars that exploitation refers to the "refinement and extension" of existing knowledge, whereas exploration refers to the "experimentation with new alternatives" (Lavie et al., 2010; March, 1991, p. 85). Over the last three decades, a substantive body of research has consistently supported March's (1991) notion that systems need to conduct both types of activities to survive and prosper (e.g., Benner & Tushman, 2003; Gupta et al., 2006; Katila & Ahuja, 2002; Uotila et al., 2009). These findings support the idea that exploitation is necessary for improving and refining an organizational form or technology that ensures its current viability and success, whereas exploration of new forms and functions differentiates the technology to ensure its competitiveness and survival in the market.

2.1.1 Appropriate Balance Between Exploitation and Exploration

In essence, firms must decide between exploitation activities that focus on cultivating existing resources and knowledge with clear and certain returns, or exploration activities that emphasize experimentation with new alternatives that have uncertain outcomes but can potentially enable firms to address new markets and adapt to changing environments (Levinthal, 2021). Undue focus on either activity can harm the overall firm performance and thus call for approaches to appropriately balance exploitation and exploration activities (He & Wong, 2004; Uotila et al., 2009). Meta-analysis studies found a positive relationship between balancing exploitation and exploration activities and performance, which is even more pronounced in technologically dynamic environments (Junni et al., 2013; Uotila et al., 2009). In such dynamic environments, the threat of rapidly aging technologies requires constant exploration, which may undermine the ability and reward of exploiting existing technologies. Operating in such an environment requires an appropriate balance between both activities to maintain a competitive advantage.

The need for conducting both activities has been recognized by IS researchers who have examined the role of technologies such as social media (Castillo et al., 2021), digital infrastructure (Montealegre et al., 2019), digitization (Park et al., 2020) emails, knowledge repositories, and groupware (Kane & Alavi, 2007) on finding an appropriate balance (achieving desired levels of equilibria) between knowledge exploitation and exploration activities. These activities have also been viewed as distinct within software development research (Vinekar et al., 2006). For example, Temizkan and Kumar (2015) examined the role of OSS communities' social network structure on the success of their exploitative and explorative activities in the form of patch development and feature-request activities respectively. Also, Ramesh et al. (2012) identified practices for balancing exploitation and exploration activities to manage inherent tensions in the context of agile distributed development teams.

The innovation literature has also highlighted the benefits of balancing exploitation and exploration for increasing sales growth rate (He & Wong, 2004) and new product innovativeness (Katila & Ahuja, 2002). Some studies in innovation examined the effect of organizational slack resources, ROA (Greve, 2007), and process management activities such as ISO 9000 certifications (Benner & Tushman, 2002) on the level of exploitation and exploration activities. This literature has conceptualized the exploitation and exploration of innovations according to their technological trajectory (incremental/radical) or their proximity to meeting existing customer needs (Benner & Tushman, 2003). Within this view, incremental innovations that meet existing customer needs are exploitative, whereas radical innovations or ones that seek emergent needs and markets are explorative (Benner & Tushman, 2003; Levinthal & March, 1993).

While these definitions of exploitation and exploration are useful at an industry level, they need to be further specified to consider the level of novelty of an innovation for a particular firm (Greve, 2007). For example, while an innovation may be radical to a firm in relation to its existing knowledge and processes, it may be incremental at an industry level. As such, Greve (2007) defines these activities from a firm's perspective and according to its assessment regarding the probability of success of an innovation. More specifically, Greve (2007) conceptualizes exploration as the development of an innovation with less knowledge and certainty regarding "the probability of successfully […] launching it in the market," and exploitation as the development of an innovation that has predictable success in the existing market, such as ones designed according to customer requests (p. 3). We adopt this view in the present study by defining *exploitation activities in app development as ones that incorporate features that have increased certainty of acceptance by the existing customer base, and exploration activities as ones that emphasize experimentation with new characteristics that have less predictable market success* (Benner & Tushman, 2003; Greve, 2007).

2.1.2 The Exploitation-Exploration Tradeoff

In addition to the argument that finding an appropriate balance between exploitation and exploration is crucial for success, another central premise of March's (1991) argument is that there is an inherent tension between exploitation and exploration activities as a result of resource allocation that may be exacerbated by inertia (Lavie et al., 2010; Levinthal & March, 1993). According to Levinthal and March (1993), these two activities compete for limited resources, such that any time and resources dedicated to one activity implies fewer resources for the other. Further, each of the activities is self-reinforcing. The outcome of exploration activities is uncertain and often leads to failure, which in turn promotes further exploration activities that potentially lead to more failures. In contrast, exploitation activities often lead to immediate success, which in turn promotes more exploitation activities that diminish the resources available for exploration (Gupta et al., 2006; Levinthal & March, 1993; Luger et al., 2018). One notable example of the trade-off between exploitation and exploration is Meta's (formerly Facebook) investment in developing the metaverse. The market success of activities related to this project are uncertain (exploration), and the dedicated resources to it hemorrhage funds away from Meta's existing app development activities that would more certainly appeal to its customer base, i.e., exploitation activities (Hays, 2022).

Although the tradeoff perspective has been supported in some studies (Uotila et al., 2009), others suggest that firms with sufficient resources can pursue both activities (Luger et al., 2018; Raisch et al., 2009). As such, while smaller firms may have to make decisions regarding the level of exploitation and exploration activities they undertake, larger firms with slack resources may sometimes be able to conduct both activities without sacrificing one over the other (Gupta et al., 2006). In the context of mobile application development, developers need to make decisions with limited resources. For one, a large fraction of applications is developed by individuals or very small companies (Borck et al., 2021). Indeed, a typical app development company "consists of just a handful of developers, with 40% of all app developers having a separate main job and 21% working only part-time on apps" (Nayebi et al., 2016, p. 552). In addition, developers must make app update decisions within very short time constraints, often measured in days. Indeed, while it is estimated that most developers of top free apps in the Google play store update their apps within a six-week period (Yang et al., 2022), many developers release daily updates of their apps (Openja et al., 2020). Given these constraints, it is reasonable to assume that app developers make tradeoff decisions between exploitation and exploration activities in updating their apps.

2.2 Exploitation and Exploration in Mobile App Evolution

Most app developers rely on open mobile app marketplaces such as Google Play, Amazon Appstore, and Apple App Store to release and distribute their apps. While these marketplaces facilitate the adoption of new apps by reducing the barrier between developers and users, they also promote fierce competition by lowering imitation and switching costs, with often a winner-take-all outcome (Wang et al., 2018). Within this highly competitive and dynamic environment (Wen & Zhu, 2019), app developers need to retain their existing customer base by addressing their needs and improving their applications on one hand, while also exploring new features to differentiate their offering, expand their market share, and adapt to the rapidly changing environment on the other.

Yet, retaining existing app users and attracting new ones is no easy feat. A recent market study estimates that while the top 10 apps lose over 30% of their users in the first week after installation, the average app loses over 80% of its users.[1] This is not surprising given that, unlike traditional IS, app developers cater to heterogeneous and geographically dispersed users who may diverge in their needs and specific uses of the application, and who face low switching costs between an abundance of substitutes. However, an app does not necessarily need to meet all of its potential users' needs, and its success ultimately depends on satisfying a large number of users leading to its widespread mass adoption (Lee & Raghu, 2014; Noei et al., 2021).

Consequently, in order to evolve and improve their apps and retain existing customers, the majority of developers often rely on users' online feedback (Palomba et al., 2018). It is estimated that nearly a quarter to one-third of these reviews provide software requirements and user experience feedback that can be useful for developers to improve the quality of the software (Iacob & Harrison, 2013). Indeed, updates that emanate from user reviews often involve improvements that existing users appreciate (Pagano & Maalej, 2013), and that increase the app's quality and market success (Palomba et al., 2018; Zhou et al., 2018; Noei et al., 2021). This approach of employing user reviews in "designing and customizing products … based on customer requests" (Greve, 2007, p. 3) is consistent with the exploitation view that emphasizes

[1] https://andrewchen.com/new-data-shows-why-losing-80-of-your-mobile-users-is-normal-and-that-the-best-apps-do-much-better/

incorporating features with increased certainty of acceptance by the existing customer base. As such, we conceptualize incorporating users' reviews (known needs) into the release updates as exploitation activities.

On the other hand, beyond user reviews, developers also rely on other diverse sources for evolving their apps such as their internal roadmaps and vision for the app (Nayebi et al., 2016; Pagano & Bruegge, 2013), scanning rival product development efforts (Al-Subaihin et al., 2021; Jiang et al., 2019), and "opportunities opened up by external technological advances" for updating their apps (Tiwana, 2015, p. 267). For example, Nayebi et al. (2016) surveyed app developers and found that half of those who participated followed a planned app release and update strategy, and some did not deviate from that strategy throughout the app's lifecycle. In addition, many app developers scan competitors' apps for feature ideas to identify emerging trends and remain competitive (Al-Subaihin et al., 2021; Jiang et al., 2019). Further, developers often update their apps in response to market and platform changes (Soh & Grover, 2020; Tiwana, 2015).

Update activities following internal planning, market analysis, or technological changes are all exploration activities because they do not capitalize on existing certainties of what is likely to succeed in the marketplace. Whereas prior work points to the existence and importance of such exploration activities, it also cautions against their potential risk and negative consequences. For example, while following a planned feature update strategy potentially allows developers to be ahead of users' needs (Allon et al., 2022; Pagano & Bruegge, 2013) and quicker to market than their competition (Nayebi et al., 2016), it may impede developers from adjusting their plans based on user and market feedback (Liu et al., 2023; Nayebi et al., 2016). Hence, app developers who allocate their time and effort to pursuing exploratory activities may end up with updates that do not pay off. For example, many app developers took advantage of the 3D touch feature of iOS introduced in 2015, while having to re-adapt their apps when it was discontinued in 2019.[2] These updates exemplify exploration activities for which market success is ambiguous and less certain than development activities addressing user requests (Greve, 2007).

In summary, this prior work on app development (1) emphasizes the importance of exploitation activities to satisfy and retain the user base, (2) points to several exploration opportunities traditionally taken by app developers, and (3) highlights the uncertainties and risks associated with these exploration activities. We draw on these findings and prior work in organizational learning in the next section to suggest that app success can benefit from a mix of exploitation and exploration activities.

2.2.1 The Effect of Exploitation-Exploration Orientation on App Success

In updating and releasing their apps, developers can exploit and focus on existing users' needs by incorporating feedback from users' reviews or explore new unanticipated needs by experimenting with new features. The rapid turnaround time between consecutive app releases and limited resources available mean that any increase in exploitation is at the expense of exploration activity, and vice versa. Therefore, the degree to which developers engage in exploitation vis-à-vis exploration activities, is an important strategic decision. Following prior work that examined the tradeoff between exploitation and exploration activity (Uotila et al., 2009), we refer to this factor as exploitation-exploration orientation.

[2] We are grateful to the reviewer for providing us with this example.

Hence, exploitation-exploration orientation reflects the mix of both activities, i.e., the degree to which a firm engages in one activity vis-à-vis the other. This view is consistent with March's (1991) original conceptualization of exploration-exploitation as activities that compete for resources and attention, and therefore exist on two ends of a continuum (see Gupta et al., 2006 and Lavie et al. 2010 for a more detailed review). Hence, exploitation-exploration orientation is viewed as a continuum representing the degree of developers' exploitation vis-à-vis exploration activities, with absolute exploration and absolute exploitation located on polar opposites of the continuum. Accordingly, high exploitation-exploration orientation reflects the emphasis towards exploitation over exploration activities, while low exploitation-exploration orientation reflects the emphasis towards exploration activities. It is worth noting that while the organizational learning literature suggests firms balance exploitation and exploration (March, 1991; O'Reilly & Tushman, 2013), an appropriate balance does not necessarily imply an even 50/50 split, but rather refers to an optimal mix of both activities that is contingent upon the firm and its environment (Uotila et al., 2009; Junni et al., 2013; Luger et al., 2018).

Within app development, prior studies have established the necessity of exploitation activities for developing successful apps (Lee & Raghu, 2014; Nayebi et al., 2016; Pagano & Bruegge, 2013; Palomba et al., 2018; Zhou et al., 2018). This is not surprising given that app success ultimately depends on user satisfaction and adoption. As such, developers are likely to improve their apps and retain their customer base by incorporating users' feedback into the consecutive app updates, i.e., exploitation activities. Hence, increased exploitation-exploration orientation (more exploitation) is likely to increase app success.

However, some exploration activities remain important, especially in competitive and dynamic environments because they enable firms to quickly adapt to rapid changes (Benner & Tushman, 2002; Levinthal & March, 1993). In an environment characterized by an abundance of substitutes, low switching costs, and rapid technological advances (Wang et al., 2018; Wen & Zhu, 2019), app developers need to explore new features to keep up with the competition (Agarwal & Tiwana, 2015; Tiwana, 2015) to retain and expand their user base. As such, insufficient exploration (i.e., too much exploitation) can diminish the app's capacity to adapt to its competitive environment.

Thus, while exploitation is necessary for app success, over-exploitation can compromise it. Hence, when considering the tradeoff between these two activities, the emphasis on exploitation over exploration activities results in increased app success up to a certain point, after which it starts to decline. This leads to our first hypothesis:

H1: Exploitation-exploration orientation in app updates has a curvilinear (inverted U-shape) effect on app success.

2.3 The Role of Feedback on the Degree of Exploitation and Exploration Activities

The majority of prior work assumes the decision to balance exploration and exploitation to be internal to the organization as either a strategic decision with calculative considerations of alternative strategies (e.g., Miranda et al., 2022) or as a result of an arrangement of organizational forms and processes (Manso, 2017; March, 1991).

The choice between exploration and exploitation is a crucial decision made by both organizations and individuals. The behavioral theory of the firm links organizational decision-making to the decisions made by individuals within the organization. This theory suggests that while organizations use information to make decisions, the decisions are influenced by the limited rationality of the individuals involved and their ability to process information from the environment (March and Simon, 1958; Cyert and March, 1963; Simon, 1956; 1979). In line with the behavioral theory of the firm, our study considers app developers as individuals who make app update decisions based on rational limitations in order to achieve greater app success. Thus, developers' decisions regarding exploration and exploitation, whether made individually or as a group, can be better understood by taking into account their individual cognitive limitations and the information they receive from their environment.

However, little research has investigated what drives the decision to exploit or explore (Raisch et al., 2009). Some scholars suggest that external performance feedback mechanisms are likely to impact decision-making activities (Adner & Levinthal, 2002; Levinthal, 2021). The impact of external feedback is likely to be greater for small enterprises. Although large and established organizations have slack resources that allow them to pursue exploration and exploitation activities (Luger et al., 2018), their elaborate organizational structures can make them insensitive to environmental feedback as signals need to travel from front-liners to upper management (Adner & Levinthal, 2002; Levinthal, 2021).

In contrast, small enterprises such as app development ones, need to be attuned and responsive to their external environment as they lack the resources for engaging in superfluous activities (Luger et al., 2018) that do not contribute to their market success and survival (Adner & Levinthal, 2008). As such, feedback from the external environment plays an important role in small organizations' strategic decisions, including balancing exploration and exploitation. This effect will likely be amplified in the highly dynamic app development environment characterized by short update times (McIlroy et al., 2016; Yang et al., 2022) and immense competition (Wen & Zhu, 2019). Because developers eventually make decisions about app updates, they need to pay close attention to feedback available to them about the performance of their apps.

In the present paper, we argue that developers' exploitation-exploration orientation is influenced by the nature of the information they receive. More specifically, we propose that the nature of users' online app reviews influences this orientation. This view is consistent with previous empirical studies in the information processing and decision-making domains, which have shown that decision makers' attention is driven by the characteristics of the information they receive (Haas et al., 2015).

2.3.1 The Role of Users' Reviews on Exploitation-Exploration Orientation

It is well documented in the literature that users' online app reviews constitute a major – if not the main – source of feedback that developers use for updating their apps in consecutive iterations (Al-Subaihin et al., 2021; Nayebi et al., 2016; Noei et al., 2021; Pagano & Bruegge, 2013; Palomba et al., 2018; Zhou et al., 2018). This feedback enables developers to improve the quality of their apps by providing them with information about system bugs, unappreciated or requested features and functionalities, in addition to usage scenarios (Al-Subaihin et al., 2021).

To use these reviews for improving the quality of their apps, developers need to first identify the relevant information (Noei et al., 2021; Pagano & Maalej, 2013), then prioritize and decide which user feedback to incorporate in the subsequent releases (Noei et al., 2021; Pagano & Bruegge, 2013). This is not an easy feat given that users' reviews are often unstructured, unclear, and potentially overwhelming in quantity. This problem is further exacerbated by repetitive and uninformative reviews (Al-Subaihin et al., 2021). While the advancements in text review mining tools are promising, they remain in their infancy and are relatively expensive (Demoulin & Coussement, 2020) keeping them out of reach for most small app developers. Even if these tools are used to aggregate users' feedback, developers still need to review and decide which user input to address in consecutive app releases. As such, many developers rely on their intuition when it comes to identifying, prioritizing, and implementing users' feedback in their consecutive app release iterations (Al-Subaihin et al., 2021; Pagano & Bruegge, 2013). This is not surprising given the time and resource limitations app developers face (Nayebi et al., 2016) in evaluating the high volume and velocity generated user reviews. Under such circumstances, app developers face a bounded rationality problem (Simon, 1956; 1973).

The fundamental argument underlying bounded rationality is that under complex and uncertain conditions, individuals are limited in their ability to gather and process information to make inferences for making decisions. These limitations are a result of task and environment complexity, human cognitive capacities, and finite resources such as time. Hence, individuals make decisions based on heuristics that result in satisfactory rather than optimal choices that are determined by the available information they have and the environmental context in which they operate (Simon, 1979; Lejarraga & Pindard-Lejarraga, 2020). More specifically, bounded rationality suggests that decision makers use heuristics to search and process a subset of select information to make decisions quickly and frugally (Simon, 1979; Gigerenzer & Gaissmaier, 2011). This view is in contrast with the classical and neoclassical rational choice perspective, where decision makers need to collect exhaustive information and evaluate all possible options to make an optimal decision. Rather, bounded rationality deviates from the classical rationality perspective and examines the success of decision makers' strategies with respect to the task environment and its objectives (Gigerenzer, 2004).

Within the app development context, developers largely rely on quantitative (Zhou et al., 2018) and qualitative (Pagano & Bruegge, 2013) information cues they receive from users' online reviews to identify the relevant and important features to include in successive app releases. Hence, app developers use these information cues to make inferences regarding their level of exploitation vis-à-vis exploration activities (Zhou et al., 2018; Palomba et al., 2018) to satisfy the largest number of users. Prior studies in the social psychology and marketing literatures that have investigated the effect of information cues from multiple disparate sources on decision makers (e.g., Harkins and Petty, 1987; Thakur, 2018) have notably focused on message content and volume. Drawing on this literature, we propose and develop two new constructs in the present paper that characterize the quantity and quality of user reviews, and test their influence on developers' exploitation-exploration orientation. The first is review concurrence, which is the magnitude of users' convergence on similar issues, and the second is review volume, which is the number of reviews following an app release.

2.3.2 Review Concurrence

Prior work suggests that developers attend to recurrent themes found in different users' reviews, thereby focusing on issues that affect the largest number of users (Noei et al., 2021; Pagano & Bruegge, 2013). Accordingly, we develop the construct of review concurrence which is the magnitude of users' convergence on similar issues and suggest that app developers are more likely to attend to issues reported in more user reviews (higher review concurrence).

It is reasonable for developers to attempt to satisfy the largest number of users given that the ultimate success of an app relies on its mass adoption (Lee & Raghu, 2014; Noei et al., 2021). And within the information-rich environment of app user reviews, developers will rely on heuristics to form judgments about which issues afflict many users. Previous studies have found that a recipient evaluates repeated messages from different sources as more important (Gallivan & Keil, 2003) and credible (Harkins & Petty, 1987). For example, Gallivan and Keil (2003) found that developers of a traditional IS attended to the issues most cited by the different users at the expense of other system issues. This is in accordance with studies in cognitive psychology which found that a message received from multiple sources increased the importance and credibility of the received information (Harkins & Petty, 1987). These studies suggest that messages repeated by multiple sources are taken more seriously and attended to.

In the context of app development, issues reported by more app users serve as a signal to the developers about the importance of those issues to the users. More specifically, users' concurrence on certain issues in their reviews provides a clear indication regarding what content developers need to address in their next release update in order to satisfy a certain number of their user base. Further, issues cited by multiple users are also likely to signal quality issues that developers can address to improve the quality of their apps. The fact that multiple users note these issues further signals the significance of the feedback to the developers.

As such, in order to improve the quality of their apps and satisfy and retain their existing users, developers are likely to spend their time addressing issues that users agree on in their reviews, i.e., exploitation activities. Hence, app developers are likely to attend to issues that more users report in their reviews at the expense of exploring new features, i.e., review concurrence is likely to result in increased exploitation-exploration orientation.

H2: Review concurrence leads to an increased exploitation-exploration orientation in app updates.

2.3.3 Review Volume

It is quite common for a moderately successful app to receive thousands of new user reviews per release[3] that potentially provide developers with valuable information they can employ to improve the quality of their apps (Al-Subaihin et al., 2021) and increase user satisfaction and engagement (Thakur, 2018; Sheng, 2019; Park & Allen, 2013). Most often, users write online reviews to express their negative or positive experiences with a product (Thakur, 2018).[4] As such, the volume of user reviews following a version release is a cue for developers regarding app issues or user engagement with their app (Thakur, 2018; Sheng, 2019).

The volume of reviews related to app issues is likely to correspond to the number of issues identified in the app release, or the significance of the issues that is causing user dissatisfaction. In this case, developers are likely to devote their efforts to addressing the stated app issues in the next release. Thus, as issue-related review volume increases, developers are likely to dedicate more of their limited time towards addressing users' issues, i.e., exploitation activities. This response is reasonable given the developers' objective to improve the quality of their apps in order to satisfy the largest number of users.

In addition, some reviews may reflect engaged users' experience with the app and reflect their desires and expectations (Park & Allen, 2013; Thakur, 2018). In this sense, engaged users are participants in the development of the app. Prior research in marketing has found that firms that respond to user reviews quickly strengthen these users' engagement and loyalty to the firm (Sheng, 2019). By responding to their reviews, developers signal to engaged users that their

[3] https://www.apptentive.com/blog/2016/10/20/average-mobile-apps-ratings-and-reviews-by-category/
[4] We are adopting the commonplace assumption that reviews tend to be bipolar and thus more review volume results in high negative or positive valence. We validate this assumption empirically in Appendix E, Figure E-1.

voices are heard and addressed, which in turn increases users' engagement with the app. This will likely motivate developers to devote their time and effort toward addressing users' stated online needs. Accordingly, as the volume of online reviews increases, developers are likely to increase their efforts towards exploitation activities at the expense of exploration, i.e., increase their exploitation-exploration orientation.

However, after a certain level, a higher review volume is likely to result in a higher cognitive load for the developers which diminishes their ability to process the information. While prior studies have found that individuals' decision quality increased with increased information, there was an inverted U-curve between the amount of information and decision quality under conditions of time pressure (Herbig & Kramer, 1994; Hahn et al., 1992; Buchanan & Kock, 2001). More specifically, under time constraints, individuals' decision quality improved with increased information and then decreased as the volume of information increased. This U-curve relation is explained by information overload, which suggests that especially under conditions of time pressure, individuals become confused as they are unable to locate and identify the relevant and critical information to make effective decisions (Buchanan & Kock, 2001; Herbig & Kramer, 1994). Consequently, individuals will cope with information overload by ignoring a large amount of the available information (Malhotra, 1984).

Similarly, given their time and resource constraints, developers facing information overload are likely to ignore a large number of users' reviews. Hence, while an increasing volume of reviews provides developers with the information they can use to update their apps, at a certain point, the large volume of information will overwhelm them, thus reducing their ability to process the information and address users' reviews, i.e. reduce their exploitation-exploration orientation. As such, whereas increased user reviews provide developers with information they can use to improve their apps and satisfy their users, too much information is likely to confuse them. This suggests that with increasing information overload, developers become less certain about what information to use from users' feedback in updating their apps, and the success of addressing this feedback becomes more ambiguous. Increased review volumes beyond a certain point are therefore likely to overload developers and diminish their ability to process the content of users' reviews and incorporate them into the app updates, i.e., reducing exploitation activities.

Hence, given that exploitation and exploration activities compete for developers' effort, the net result will be a curvilinear effect on exploitation-exploration orientation in the following app update release.

H3: Review volume has a curvilinear (inverted U shape) effect on the exploitation-exploration orientation in app updates.

3 Methods

3.1 Data

We sampled Android apps and then collected and integrated data about the apps' release history and user reviews from the following sources. First, a list of 8,950 popular Google Play app titles was obtained from APK4Fun. Second, we collected each app's general information and release history from AppAnnie, a leading aggregator of mobile market data. For each app, this platform provides detailed app information, including description, category, and review rating (e.g., number of stars). It also provides detailed information about each app's version release, including the release date and the new features and updates.

Third, we collected the app reviews from AppBot, which is a mainstream app review aggregation platform. This platform provides complete reviews and analytics of Google Play store apps. Because the analytics are only available for apps released after 2014, we selected apps that were created after April 12th, 2015 in order to capture all user reviews of an app and its trajectory since its inception. Consequently, given that our study examines the effect of user reviews on app updates, we removed apps with fewer than ten releases and had no reviews between consecutive releases. This left us with a sample of 1190 apps. AppBot usage limitation constrained us to a random selection of 10% of the apps. We reviewed the selected apps and found that our selection is representative of major app segments such as business, lifestyle, and education. Finally, for the 119 selected apps, we extracted all reviews written in English between April 13, 2015, and May 26, 2017. Each app had an average of 13.83 releases resulting in a final dataset of 1,046,553 reviews for the 119 apps. Each review includes a review date, content, subject, rating, app version, author, and emotion. This data was aggregated at the release level of each app to construct an app-release panel dataset.

3.2 Variables and Measurements

The dependent (i.e., exploitation-exploration orientation and app success) and independent variables (i.e., review concurrence and review volume) were operationalized as follows.

3.2.1 Exploitation-Exploration Orientation

We measured *Exploitation-Exploration Orientation* as the degree of exploitation to which developers have devoted their app update efforts. The choice to measure orientation from an exploitation perspective is a result of our intent to examine the effects of user reviews on app developers. As such, *Exploitation-Exploration Orientation* is measured using the following formula:

$$Exploitation - Exploration\ \ Orientation = \frac{Update\ Exploitation}{Update\ Exploitation + Update\ Exploration}$$

Update Exploitation measures the app version updates that reflect users' review feedback following the prior release. This part is quantified using the similarity between the description of the current release of the app and the content of the reviews following the previous release. More specifically, for each app release, we compared two texts. The first is all user reviews generated between the previous and the current release of the app. The second is the updated feature texts of the current release. These texts were then cleaned according to standard text preprocessing procedures (Bird, 2006) which included removing the stop words and stemming. The count of the resulting matched content words between the two texts provided the measurement for *Update Exploitation.* In contrast, *Update Exploration* is measured as the count of content words in updated feature texts of the current release that are not in the user's previous reviews. A higher *Exploitation-Exploration Orientation* value indicates that app developers' activities tilt more towards exploitation, i.e., the app updates reflect users' feedback.

Our *Exploitation-Exploration Orientation* measure is based on the assumption that there is a tradeoff between exploitation and exploration, which implies a negative correlation between both activities. This assumption is validated with an average correlation factor of -0.48 ($p < 0.01$) between these two measurements within the same app, indicating that developers engage in exploitation at the expense of exploration and vice versa.

3.2.2 App Success

While information systems success has been defined in numerous ways including rating (Tiwana, 2015) and survival on top charts (Lee & Raghu, 2014), Agarwal and Tiwana (2015) argue that a system's mortality may be "the most meaningful" outcome measure (p. 478). This is particularly a meaningful view of app success because of the fast-changing nature of mobile devices and the need to continuously update apps for them to work on new operating systems and hardware infrastructure. In this environment, apps that fail to continuously release new updates quickly become obsolete (Lee & Raghu, 2014; Palomba et al., 2018; Zhou et al., 2018). As such, we define an app's market success by its survival in the marketplace. This survival depends on users' adopting the app, which in turn provides developers with resources to continue updating their apps.

The market success of the app is operationalized as the release of new updates, which serves as an indicator of survival in the marketplace. However, this data is right censored because app developers steadily release new versions in a stochastic manner over a period of time, and then become inactive. While inactivity signals exiting the market, developers may still release an update after a period of inactivity. Accordingly, we employ survival analysis to estimate the probability of survival using the app release frequencies as well as their timing.

To model whether app developers will release new versions using past data, we use the Pareto/Negative binomial distribution (Pareto/NBD) survival models (Schmittlein et al., 1987). This model is widely accepted and used for forecasting customer churn (Fader et al., 2005) which is a similar context to ours; although customers purchase regularly for a period, they may become inactive. This model includes two components. The first is the Pareto component which models the time to become inactive, and the second is the NBD component which models the frequency of purchases. This model requires two main inputs to estimate the probabilities of survival: recency, which describes when the last update occurred, and frequency, which shows how many releases an app had within a specific time period. We use the previous release history to estimate the probability of a new release for each app (i.e., *survival)*. Further, given that the average time in days between each app release is about a month, we use the Lifetimes Python package[5] to estimate the probability of app short-term and long-term survival within one month and three years consecutively.[6]

3.2.3 Review Volume and Review Concurrence

Following previous studies that viewed *Review Volume* as the number of competing issues for attention (Haas et al., 2015), we measure it as the number of reviews in thousands created before the current release and after the previous release for the same app. The new concept of *Review Concurrence* was measured as follows. We first used the standard text processing procedure of removing the stop words and stemming the remaining words to extract content words. Then, we kept the words appearing in at least 2 reviews, leaving a total of 2827 words, that were ranked based on their frequencies. Finally, one expert familiar with app development went through the remaining list and removed irrelevant words. Our final list included 467 words related to apps and their features. Then, for each release of every app, we extracted all reviews generated between the current release and the previous release for the same app and also implemented the standard text processing procedure. Next, we counted the number of keywords in these reviews and the number of reviews that contain each keyword, to calculate *Review Concurrence* as follows:

$$Review\ Concurrence_{ij} = \frac{\sum_{k=1}^{n} \#ofReviews_{kij}}{\#ofReviews_{ij} \times K_{ij}}$$

Where $Review\ Concurrence_{ij}$ is for release j of an app *i*. $\#ofReviews_{kij}$ is the number of reviews that contain the keyword k for release j of the app *i*. $\#ofReviews_{ij}$is the number of

[5] Details on this package can be found at the following link: https://pypi.org/project/Lifetimes/0.2.2.2/
[6] The average app survival length in our data is around 18 months.

reviews for release *j* of the app i. K_{ij} *is* the total number of keywords for release *j* of the app *i.*

3.2.4 Control Variables

We measured two groups of variables to control for the heterogeneity of reviews and app releases. The first group is *Review and Release Control Information*, which consists of seven variables that measure the characteristics of reviews created between the current release and the previous release of an app. These are *rating, sentiment, review length, release, update length, review velocity,* and *quarter*. The second group, *App Control Information*, consists of four variables that measure the characteristics of the current release: *in-app purchase*, *app category*, *user base*, and *android system.*

The variables' definitions and measures are described in Table 1, and their descriptive statistics and correlation matrix are shown in Table 2.

4 Analyses and Results

4.1 Effect of Exploitation-Exploration Orientation on App Success

4.1.1 Main Analysis

We test the effect of developers' exploitation-exploration orientation activities on apps' success measured by their survival in the marketplace over time (H1). Given that the dependent variable is a probability (survival) that is censored, we apply Tobit models at the app level of analysis, with *Exploitation-Exploration Orientation* as the independent variable. The following model is estimated:

$$Y_{it} = \alpha + \beta_0 \times Update\ Exploitation_{it} + \beta_1 \times Exploitation - Exploration\ Orientation_{it} + \beta_2 \times Exploitation - Exploration\ Orientation_{it}^{2} + \gamma \times X_{it} + \varepsilon_{it} \quad (1)$$

where Y_i is the survival probability of an app i, β_1 is the $Exploitation - Exploration\ Orientation_{\ it}$ parameter that measures the linear relationship of exploitation to exploration, and β_2, is the quadratic term of $Exploitation - Exploration\ Orientation_{\ it}$ that measures the non-linear effects of exploitation. X_{it} represents a vector of covariates that may influence developers' decisions, including review information, release, app control information, and quarter dummies as specified in Table 1. We further control for $Update\ Exploitation_{\ it,}$ which measures the update exploitation. It is worth noting this model excludes *Update Exploration* given the high multicollinearity between Update Exploitation and Update Exploration (VIF value of 8.22). We estimate two model specifications using *Survival* at *1 month* and *Survival* at *3 years* as the dependent variables which respectively examine short-term and long-term survival.

The results are reported in Table 3 and are consistent with our Hypothesis 1 demonstrating an inverted U-shaped relationship between *Exploitation-Exploration Orientation* and app short and long-term survival.[7] While the coefficients of *Exploitation-Exploration Orientation* variables are statistically (positively) related to app short and long-term survival, the quadratic terms are significantly (negatively) associated with app survival. We found that *Exploitation-Exploration Orientation* is positively associated with app survival *up to a point* after which this positive relationship declines. To calculate the threshold's inflection point, we used the following equation for the polynomial regression of *Exploitation-Exploration Orientation*: $Y = \beta_0 + \beta_1 Exploitation - Exploration\ Orientation + \beta_2\ Exploitation - Exploration\ Orientation^2 + \varepsilon$. We then computed the inflection point by the coefficients of

[7] We also replace Exploitation with Exploration as a numerator in our *Exploitation-Exploration Orientation* measure as a robustness test for our hypothesis I. The results are consistent with our expectations providing an inverted U-relationship with app success as detailed in Appendix D Table D1.

the predictor *Exploitation-Exploration Orientation* as shown by the following equation: *Orientation inflection* = – $\beta_1/2\beta_2$. The inflection point is 0.433 (exploitation activities are 0.76 (=0.433/(1-0.433)) times more than developer's exploration activities) for short-term survival; additionally, this U-shaped pattern is illustrated in Figure 2. Thus, Hypothesis 1 is supported.

4.1.2 Robustness Tests

In our main analysis, we measured exploitation and exploration according to whether the features described in the app updates appeared in app users' reviews. This is a conservative method that captures exactly matched text content found in the user reviews and the update descriptions, and may not capture the semantic similarity between user reviews and mobile app update descriptions. For example, a user request could suggest that "it would be better to show playing duration of the song," while the app update text could state that "Added an option to display the song remaining time on the player screen." Although both texts discuss the same topic – displaying song play time, our conservative approach of exact matching will not capture this review as being exploited. To address this concern and also test the robustness of our findings, we implemented BERT (Bidirectional Encoder Representations for Transformers) to capture the semantic similarity of these two texts (see Koroteev, 2021 for a review). The output of this process is a semantic similarity score ranging from 0 (absolutely not similar) to 1 (almost identical). The higher the score, the more exploitation activity is undertaken by the developer. In the given example above, the similarity score of 0.803 indicates that these two texts address similar content. We implemented this approach to quantify the semantic similarity between app update descriptions and app user reviews and also used Amazon Mechanical Turk (AMT) service to verify the performance of our similarity analysis.

We then repeated our H1 analysis using this alternative measurement. Specifically, we replaced *Update Exploitation*, *Exploitation-Exploration Orientation*, and the *Exploitation-Exploration Orientation* quadratic term in the main analysis with this newly created measurement and its quadratic term. The results are consistent with those in our main analysis. That is, the semantic similarity between app update descriptions and app user reviews is positively associated with app survival *up to a threshold* after which this positive relationship declines. The detailed process of this robustness test and its results are provided in Appendix A.

In the second and third robustness tests, we used two alternative dependent variables to test H1. Specifically, we replaced the dependent variable (app survival) with 1) users' app ratings, and 2) users' sentiments regarding the app. Accordingly, we estimated two model specifications of the effect of addressing users' reviews in app updates on their ratings and sentiment. For each app, we used the change in users' ratings between two consecutive versions as the first new dependent variable in one specification, and the change in user sentiment between them as the second new dependent variable in another specification. The main independent variables in both models were *Exploitation-Exploration Orientation* from the earlier app version and its quadratic term. We also included the same set of controls as our main analysis. The results of this analysis are shown in Appendix B Table B-1 and are also consistent with our main analysis results.

Table 1: Construct definitions and measurements

Construct	Definition	Measurement
Success	The survival of the app in the marketplace.	Probabilities that measure the likelihood of an app surviving over one month (short term) and three years (long term).
Update Exploitation	The extent of exploitation activities in a new app release	The number of matched content words between feature texts of the text of the current release update and the reviews created before this current release and after the previous release for the same app.
Update Exploration	The extent of exploration activity in a new app release	The number of content words in the feature texts of the new release update but not in previous reviews for the same app.
Exploitation-Exploration Orientation	The degree to which developers engage in exploitation vis-à-vis exploration activities in the new app release	This variable is measured using the following formula: $Exploitation - Exploration\ Orientation = \frac{Update\ Exploitation}{Update\ Exploitation + Update\ Exploratoin}$
Review Concurrence	The magnitude of users' convergence in their reviews on the same issue following an app release.	The value is calculated using the following formula: $Review\ Concurrence_{ij} = \frac{\sum_{k=1}^{k} \#ofReviews_{kij}}{\#ofReviews_{ij} \times K_{ij}}$ $Review\ Concurrence_{ij}$ is for release j of app i. $\#ofReviews_{kij}$ is the number of reviews that contain keyword k for release *j* of app *i*. $\#ofReviews_{ij}$ is the number of reviews for release *j* of app *i*. K_{ij} *is* the total number of keywords for release *j* of app *i.*
Review Volume	The number of user reviews developers receive following an app release.	The number of reviews (in thousands) created before a new release and after the previous release, for the same app.
User reviews and app release control variables		
Avg Rating	App reputation measured with the average rating.	The average rating given by consumers to an app release before the new release and after the previous release. This variable ranges from 1 to 5.
Avg # of Reviews(k)	The average number of user reviews per release till current release.	The average number of reviews (in thousands) per release till the current release for an app.

User reviews and app release control variables (cont.)		
Avg Sentiment	Consumer sentiment towards the app.	The average sentiment score for an app release created before the new release and after the previous release, coded as 1 (positive), -1 (negative), or neutral (0).
Review Length	The average length of the reviews.	The average number of words in a review for an app release created before the new release and after the previous release.
Review Velocity	The average days between consecutive reviews	Log values of average number of hours between consecutive reviews for for an app release created before the new release and after the previous release.
Release	The current release number for the focal app.	The sequential number of the current release for the focal app, starting from 1.
Update Length	The length of update contents of new release.	The number of content words in the new release.
Quarter	The quarter when the new release is published.	A variable shows the quarter of the year of the most recent app release, indicating 1st, 2nd, 3rd, or 4th quarter.
App control variables		
In-app Purchase	Paid app	An indicator that equals 1 if it is a paid app, otherwise, 0
App Category	The category the app belongs to.	Indicates the app category as provided by the Google Play store. There are 5 app categories: *Education, Business, Entertainment, Lifestyle, and Travel*
User Bases	The estimated number of downloads for an app.	A variable that indicates the estimated range of downloads for an app. There are 4 ranges: <1 million, 1-5 million, 5-50 million, and more than 50 million.
Android System	The Android OS version of the app.	A variable represents the Android system for the app. There are 13 versions of this operation system starting from 2.1 and up to 5.0 and up.

Table 2. Main Variables Descriptive Statistics and Correlation Matrix

Panel A: Descriptive statistics of model variables (N = 1952)

#		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	Min	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	-1.00	2.20	0.00	1.00	10.00	0.00
	Max	0.99	0.84	1.00	3.71	4.08	0.81	66.79	12.36	5.00	1.00	6.24	6.66	71.00	621.00	1.00
	Q1	0.90	0.76	0.12	1.10	1.95	0.11	0.01	0.04	3.63	0.12	3.85	0.33	5.00	106.00	0.00
	Median	0.98	0.81	0.31	1.79	2.57	0.23	0.06	0.13	4.11	0.25	4.26	0.76	10.00	211.50	0.00
	Q3	0.99	0.83	0.56	2.57	3.09	0.38	0.29	0.59	4.43	0.39	4.61	1.54	19.00	393.00	1.00

Panel B: Correlation matrix (Pearson)*

#		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	*Success-Survival(1month)*	1														
2	*Success-Survival(3years)*	**0.97**	1													
3	*Exploitation-Exploration Orientation*	**0.13**	**0.12**	1												
4	*Update Exploitation*	**0.15**	**0.14**	**0.75**	1											
5	*Update Exploration*	**-0.06**	**-0.06**	**-0.50**	**0.11**	1										
6	*Review Concurrence*	***0.05***	***0.05***	**0.34**	**0.17**	**-0.27**	1									
7	*Review Volume*	0.13	0.12	**0.12**	***0.06***	**-0.07**	***0.05***	1								
8	*Avg # of Reviews(k)*	**0.10**	**0.09**	**0.45**	**0.24**	**-0.32**	**0.46**	**0.13**	1							
9	*Avg. Rating*	0.04	0.03	0.03	***0.05***	0.04	**0.08**	**0.07**	**0.14**	1						
10	*Avg. Sentiment*	**-0.07**	**-0.07**	-0.03	0.00	**0.07**	0.03	0.02	***0.05***	**0.64**	1					
11	*Review Length*	**-0.19**	**-0.19**	**0.13**	**0.20**	0.02	-0.03	**-0.18**	**-0.06**	**-0.37**	-0.01	1				
12	*Review Velocity*	***0.05***	***0.05***	**0.14**	**0.07**	**-0.12**	**0.26**	0.03	**0.16**	**0.18**	**0.05**	**-0.19**	1			
13	*Release*	***0.05***	0.03	**0.08**	**0.11**	0.02	-0.02	0.04	**0.14**	**0.09**	0.02	**0.06**	-0.02	1		
14	*Update Length*	-0.01	-0.01	0.00	**0.51**	**0.78**	**-0.06**	0.01	**-0.10**	0.02	0.04	**0.10**	-0.04	-0.01	1	
15	*In-app Purchase*	***-0.05***	***-0.05***	-0.03	0.01	**0.07**	0.00	0.01	0.01	***0.05***	***0.04***	***0.05***	***-0.04***	***-0.05***	0.02	1

* Bold values indicate correlations that are significant at $p < 0.01$; Italics and bold indicate correlations that are significant at $p < 0.05$.

Table 3. Regression of Survival on Exploitation-Exploration Orientation

Variables	**Survival (1 month)**	**Survival (3 years)**
Exploitation-Exploration Orientation	0.1468**	0.1342**
	(0.0704)	(0.0587)
Exploitation-Exploration Orientation × *Exploitation-Exploration Orientation*	-0.1692**	-0.1494**
	(0.0753)	(0.0627)
Update Exploitation	0.0238***	0.0198***
	(0.0078)	(0.0065)
# of Reviews(k)	-0.0105*	-0.0090**
	(0.0054)	(0.0045)
Avg Rating	-0.0842***	-0.0930***
	(0.0251)	(0.0209)
Avg Sentiment	0.1247*	0.1570***
	(0.0705)	(0.0587)
Review Length	-0.0026***	-0.0024***
	(0.0003)	(0.0002)
Review Velocity	-0.0086	-0.0080
	(0.0062)	(0.0052)
Update Length	-0.0080	-0.0065
	(0.0100)	(0.0083)
Business	-0.2721***	-0.2300***
	(0.0200)	(0.0167)
Entertainment	0.0100	0.0063
	(0.0116)	(0.0096)
User Base(<1 million)	0.0332*	0.0435***
	(0.0195)	(0.0162)
User Base(1-10 million)	-0.0881***	-0.0641***
	(0.0131)	(0.0109)
In-app Purchase	-0.0284***	-0.0214**
	(0.0107)	(0.0090)
Constant	1.6336***	1.4672***
	(0.1208)	(0.1007)
Observations	1,952	1,952
Android Version Dummies	YES	YES

Standard errors in parentheses; *** p<0.01, ** p<0.05, * p<0.1

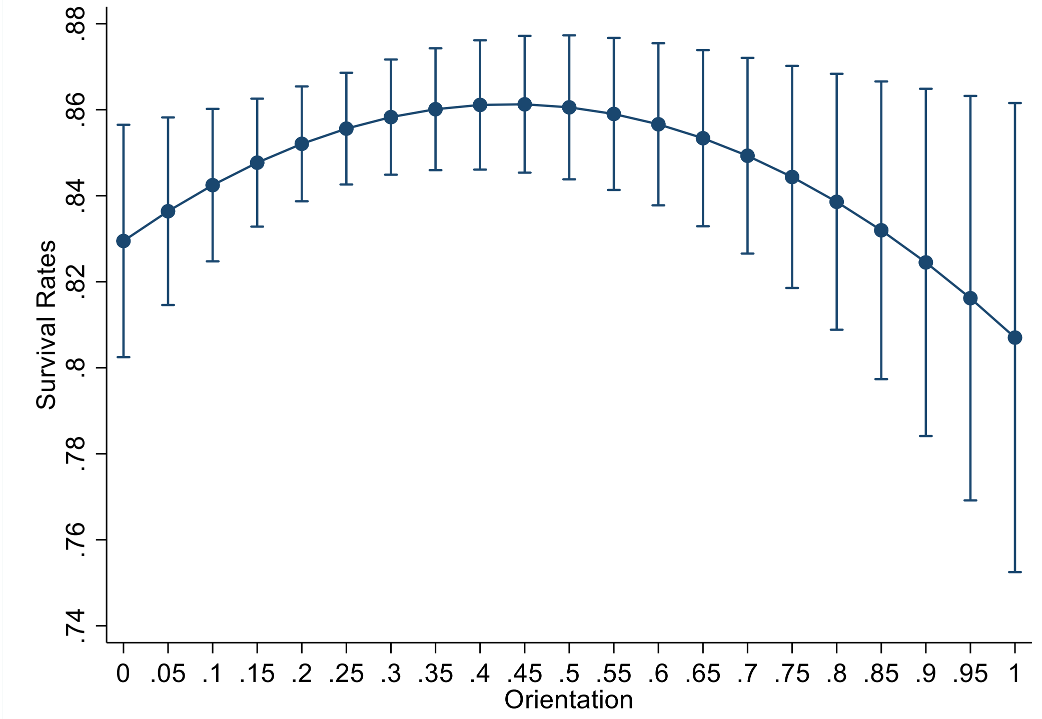

Figure 2. Exploitation-Exploration Orientation and App Survival with 95% CIs

4.2 The Effect of Review Concurrence and Volume on Exploitation-Exploration Orientation

4.2.1 Main Analysis

Initially, a visual approach is used to explore the curvilinear effect of review volume on *Exploitation-Exploration Orientation* (H3). Specifically, we regress the *Exploitation-Exploration Orientation* on *Review Volume* and its quadratic term. Then, we plot the resulting fitted line along with a 95% confidence interval which shows support for a curvilinear relation. As shown in Figure 3, *Exploitation-Exploration Orientation* initially increases with the increase of review volume, then gradually drops after review volume reaches the value of around 3.5 (the log value of the number of reviews in thousands).

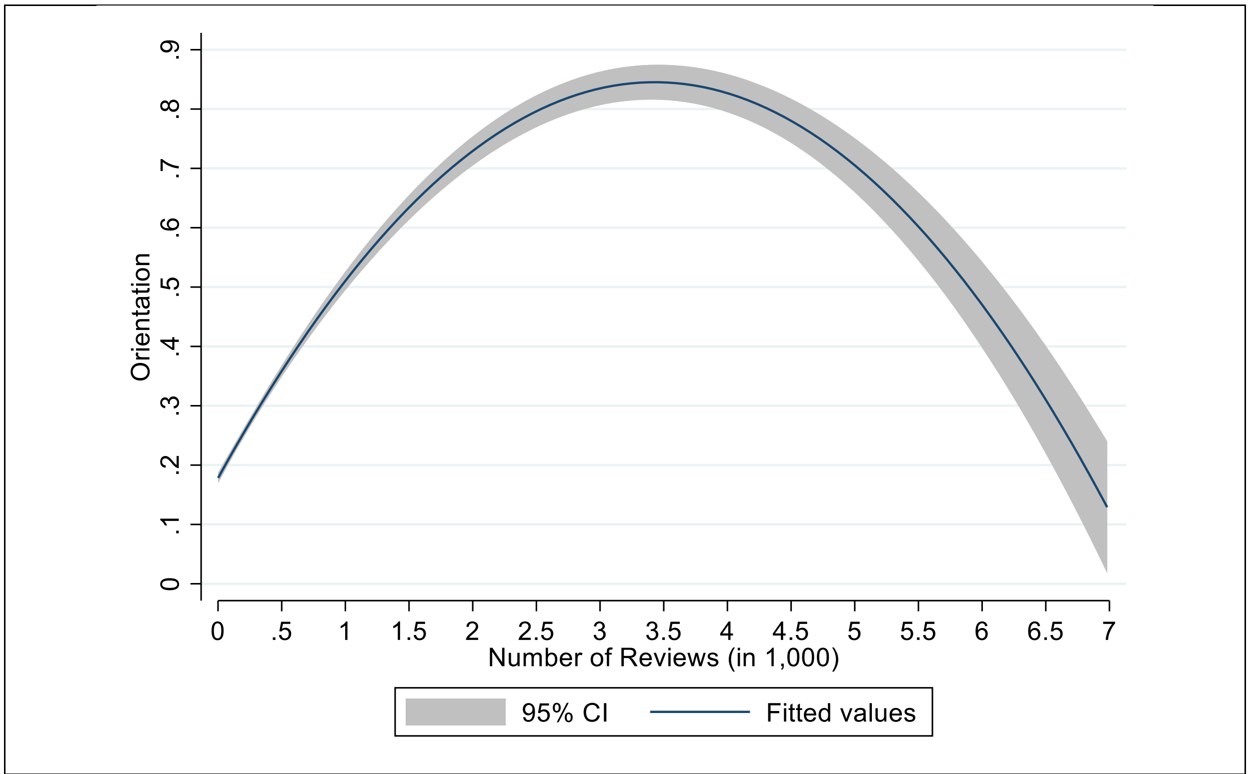

Figure 3. Review Volume and Exploitation-Exploration Orientation

We ran two regression specifications to test H2 and H3. The dependent variable is *Exploitation-Exploration Orientation*. Specification I includes *Review Concurrence,* and *Review Volume* and its quadratic term as the independent variables, testing their effects on exploitation-exploration orientation without covariates. On top of Specification 1, Specification 2 includes additional control variables in ***User reviews and app release control variables*** group in Table 1. In both specifications, we also included quarter dummies to control for time effects. The following OLS models with app fixed effects are estimated:

$$Y_{it} = \alpha + A_i + T_t + \beta_0 \times ReviewVolume_{it} + \beta_1 \times ReviewVolume_{it} \times ReviewVolume_{it_{it}} + \beta_2 \times ReviewConcurrence_{it} + \varepsilon_{it} \quad (2)$$

$$Y_{it} = \alpha + A_i + T_t + \beta_0 \times ReviewVolume_{it} + \beta_1 \times ReviewVolume_{it} \times ReviewVolume_{it_{it}} + \beta_2 \times ReviewConcurrence_{it} + \gamma \times X_{it} + \varepsilon_{it} \quad (3)$$

where Y_{it} is the *Exploitation-Exploration Orientation* between update contents of the *i*th release and review contents between *i*th release and *(i-1)*th release. *ReviewVolume*$_{it}$ and *ReviewConcurrence*$_{it}$ are the main independent variables. Parameters β_1, β_2, and β_3 measure the effects of $ReviewVolume_{it}$ and its quadratic and $ReviewConcurrence$ on Y_{it}. A_i and T_t are the app and time fixed effects, respectively. X_{it} is a vector of control covariates that may affect app developers' decisions. It includes *User reviews and app release control variables* such as the days between reviews (*Review Velocity*) noted in Table 1 (*Avg # of reviews(k)* is dropped from the estimation due to multicollinearity). Our unit of analysis is at the release level. Given that we include app fixed effects, the estimate will remove the time-invariant effects related to the app characteristics. We cluster the errors at the app level. Further, other review and release characteristics and time dummy control the time-varying effects.

The results reported in Table 4 support H2 and H3.[8] In both model specifications, *Review Concurrence* coefficients are positive and statistically significant therefore supporting H2. Further, the coefficients of *Review Volume* variable are positive and statistically significant at the 1% level and the coefficient of its quadratic term is statistically significant with a negative value (-0.0090, P<0.01) in Specification 2. These results imply a curvilinear (inverted-U) relationship between *Review Volume* and *Exploitation-Exploration Orientation*, therefore supporting H3.

[8] We also replace Exploitation with Exploration as a numerator in our *Exploitation-Exploration Orientation* measure as a robustness test for our hypotheses H2 and H3. The results show that coefficients of *Review Concurrence* is positively related to *Exploitation-Exploration Orientation* and *Review Volume* has an inverted U-shape relationship with *Exploitation-Exploration Orientation*, confirming hypotheses H2 and H3. The details of this analysis is provided in Appendix D Table D2.

Table 4. Regression of Exploitation-Exploration Orientation on Review Volume and Concurrence

Variables	M1	M2
Review Volume	0.0553***	0.0585***
	(0.0096)	(0.0100)
Review Volume × Review Volume	-0.0090***	-0.0090***
	(0.0002)	(0.0002)
Review Concurrence	0.0442***	0.0553***
	(0.0166)	(0.0156)
Review Velocity		-0.0215
		(0.0133)
Avg Rating		0.0194*
		(0.0112)
Avg Sentiment		-0.0474**
		(0.0200)
Review Length		0.0809***
		(0.0120)
Release		-0.0002
		(0.0010)
Update Length		-0.0012
		(0.0121)
Constant	0.3199***	-0.0653
	(0.0297)	(0.1111)
Observations	1,952	1,952
R-squared	0.5697	0.5800
Year-quarter	YES	YES

Robust standard errors in parentheses
*** p<0.01, ** p<0.05

4.2.2 Addressing Endogeneity

There are three potential endogeneity risks in our analysis. The first is reverse causality, which is not an issue in the present study given that the outcome dependent variable lags the independent variables by one time period (i.e., previous reviews cause subsequent app updates). The other two potential endogeneity risks are omitted variables and simultaneity. With regards to omitted variables, developers' decisions may be influenced by other information such as privacy policies that necessitate additional security features. These variables correlate with our endogenous variables (review concurrence and volumes) and enter into the error term when omitted, resulting in endogeneity. In contrast, simultaneity arises when the independent variables are simultaneously determined along with the dependent variable. In particular, apps within the same category cross-influence app developers and users. An additional feature in other apps could lead to app developers of the focal app to add that feature and users of focal apps to comment on that feature.

These endogenous concerns are commonly addressed by using instrumental variables (IV) (Wooldridge, 2010). The instrumental variables need to be exogenous (uncorrelated with the error term) but partially correlated with endogenous variables. Yet, finding appropriate instrumental variables in panel data is challenging because of time-related shocks that may occur during data generation of both independent and exogenous variables (Menon & Kohli, 2013). Following common practice in IS studies using panel data (e.g., Chen et al., 2015; Greenwood & Gopal, 2015; Jabr & Zheng, 2014), we implement a dynamic model proposed by Arellano and Bond (1991) to address these concerns. In the dynamic panel model, the lag terms of the dependent variable and endogenous explanatory variables (i.e., review volumes and review concurrence) are exogenous, and can therefore be used as instrumental variables (Mallipeddi et al., 2021). More specifically, we implement the system generalized method of moment (GMMs) estimator (Arellano & Bond, 1991) to help resolve the endogenous issues described above. The estimator employs a system of two equations:

$$\Delta Y_{it} = \alpha \Delta Y_{it-1} + \beta \times \Delta W_{it} + \emptyset \Delta X_{it} + \Delta \varepsilon_{it} \text{ (first-differences equation)} \quad (4)$$

$$Y_{it} = \alpha Y_{it-1} + \beta \times W_{it} + \emptyset X_{it} + \varepsilon_{it} \text{ (levels equation)} \quad (5)$$

where W_{it} is a vector of endogenous regressors ($ReviewVolume_{it}$ and its quadratic terms, and $ReviewConcurrence_{it}$). After removing the fixed effects through the first-differences equation

(equation 4), the system GMM estimator uses the lagged values of the differences as instruments with a minimum lag difference of one and a maximum lag difference of five periods.

Table 5 reports the results. The coefficient for the lagged dependent variable is less than one, indicating a converged estimate. The Arellano-Bond test suggests that the first-order serial correlation (AR1) is significant but we do not find evidence of the second-order serial correlation (AR2). This implies that using t-1 is adequate for our analysis. Further, both coefficients for *Review Volume* and *Review Concurrence* are significantly positive at the 5% level, and the coefficients of the *Review Volume* quadratic term are significantly negative at the 1% level. These results confirm our findings in the main analysis.

Table 5. Regression of Exploitation-Exploration Orientation on Review Volume and Concurrence

(Dynamic Panel Model Analysis)

Variables	Specification 1	Specification 2
Exploitation-Exploration Orientation$_{(t-1)}$	-0.0147*	-0.0099**
	(0.0083)	(0.0046)
Review Volume	0.1142***	0.0865***
	(0.0275)	(0.0245)
Review Volume ^{2}	-0.0030***	-0.0031**
	(0.0008)	(0.0014)
Review Concurrence	0.0113***	0.0425**
	(0.0010)	(0.0177)
Review Velocity		-0.0500***
		(0.0158)
Avg Rating		0.0003
		(0.0002)
Avg Sentiment		0.0107
		(0.0172)
Review Length		-0.0389
		(0.0292)
Release		0.0920***
		(0.0192)
Update Length		0.0033**
		(0.0016)
Constant		-0.0329**

		(0.0166)
Wald x^2	617.39	803.45
AR(2) Statistics	0.123	0.165
Hansen's Test Statistics	0.535	0.900
Observations	1,714	1,714
Time fixed effects	YES	YES

Robust standard errors in parentheses
*** $p<0.01$, ** $p<0.05$, * $p<0.1$

4.2.3 Robustness Tests

To further confirm our findings of H2 and H3, we ran a series of robustness tests. We first replaced *Exploitation-Exploration Orientation* with the semantic similarity score used in Section 4.1 as a dependent variable. The two model specifications of equations (2) and (3) above were then estimated using the alternate dependent variable. The results of this analysis are reported in columns 2 and 3 of Table 6, further confirming our initial findings that support H2 and H3.

Table 6. Robustness Tests (Alternative Model and Dependent Variable)

Variables	Alternative Dependent Variable		Mixed Effects	
Review Volume	0.0293***	0.0288***	0.0583***	0.0598***
	(0.0066)	(0.0070)	(0.0040)	(0.0041)
Review Volume × Review Volume	-0.0050***	-0.0050***	-0.0090***	-0.0090***
	(0.0001)	(0.0001)	(0.0001)	(0.0001)
Review Concurrence	0.0292***	0.0303***	0.0445***	0.0584***
	(0.0093)	(0.0086)	(0.0128)	(0.0128)
Review Velocity		-0.0025		-0.0148**
		(0.0071)		(0.0063)
Avg Rating		0.0089*		0.0214**
		(0.0048)		(0.0094)
Avg Sentiment		-0.0177**		-0.0555***
		(0.0074)		(0.0201)
Review Length		0.0208***		0.0861***
		(0.0048)		(0.0107)
Release		-0.0010		-0.0004
		(0.0006)		(0.0006)
Update Length		-0.0014		0.0014
		(0.0060)		(0.0059)
Constant	0.1001***	-0.0085	0.2950***	-0.1366*

	(0.0134)	(0.0504)	(0.0256)	(0.0750)
Observations	1,952	1,952	1,952	1,952
R-squared	0.5476	0.5527		
Number of groups	119	119	119	119
Quarter Dummies	YES	YES	YES	YES

Robust standard errors in parentheses
** p<0.01, * p<0.05

For the second robustness test, we used a mixed-effects linear model. In the main analysis above (equations 2 and 3 in section 4.2.1 above), we ran an app fixed effects analysis assuming the estimated parameter β does not vary for the different apps, i.e., the estimate is at the app level, considering aggregated reviews. However, in reality, these estimates may differ within each app. To account for this in our robustness test, we conducted a linear mixed-effects rather than a linear fixed-effects model. This analysis allows us to explore the difference in effects between and within apps (Torres-Reyna, 2007). A Maximum Likelihood estimator was run, and the assumption to allow both fixed and random effects in our estimates was relaxed. As such, we implemented a multilevel mixed-effects linear model (Rabe-Hesketh et al., 2002), to consider the app-level and release-level variables. This model accommodates the dependence between app releases and reviews, in addition to the variability in exploitation-exploration orientation within and between apps. The random effects are at the app level. All models include the same set of controls.

The results shown in columns 4 and 5 of Table 6 confirm our initial findings that review concurrence positively affects exploitation-exploration orientation and that the relationship between review volume and exploitation-exploration orientation is curvilinear. Therefore, these results confirm our main findings, supporting H2 and H3.

In the last robustness test, we explore how developers are likely to respond to the different types of review contents (e.g., system bugs, new features, praise, etc.). More specifically, we implemented automatic approaches to categorize user reviews content, then investigated whether app developers responded differently to these reviews. The results of this analysis support our main findings by demonstrating that app developers are more likely to include features frequently mentioned by app users. In addition, it demonstrates that developers prioritize issues over new features requested by users. Appendix C provides the details of this analysis.

4.2.4 Post-Hoc Analysis: Moderating Effect of Review Sentiment

One common finding among existing studies on user online reviews is that online reviews can affect consumers' decisions via review sentiment (e.g., Duan et al., 2008; Yin et al., 2017). App developers (i.e., sellers) in our context are likely aware of the potential effects of review sentiment on app users (i.e., consumers) and thus are likely to consider review sentiment when updating apps. Thus, we explore how review sentiment can moderate the relationship between review volume/review concurrence and exploitation-exploration orientation. We briefly present this analysis here and provide the results in more details in Appendix E.

To measure the moderating effect of review sentiment on our findings, we estimate the same model as our main model but add the interaction terms between review sentiment and review volumes and concurrence. The results in Appendix E Table E-1 (and Figures E-2 and E-3) show that with increased negative review sentiment, review volume will have a more positive effect on exploitation-exploration orientation. As noted above, this is expected given that app users may write online reviews to express their negative or positive experiences with the app. Negative reviews often reflect dissatisfaction from app users, and the extremity of such negative sentiment likely implies the significance of the issues. As such, a large number of negative sentiment reviews are likely to focus app developers' attention and efforts towards addressing the stated issues in the next version release.

The results in Table E1 (Appendix E) also show that with increased review sentiment, there is an insignificant, negative relationship between review concurrence and exploitation-exploration orientation. We would expect that with more recurring positive themes, app developers may interpret that the existing features and functionalities of the app have fewer critical issues or complaints about the app. Consequently, they may make fewer changes in the new releases or focus on exploring new innovative features to stay ahead of the competition. However, this insignificant relationship suggests that the relationship between review concurrence and exploitation-exploration orientation is less influenced by review sentiment. That said, the negative relationship suggests that while recurring negative themes may compel developers to address the stated issues of dissatisfied users, recurring positive themes discourage developers from making more changes to further meet users' needs and desires.

5 Discussion and Contributions

In the present study, we examined the influence of users' online reviews on mobile application evolution and success. We distinguished between two types of app evolution mechanisms: exploitation activities that rely on users' online reviews, and exploration activities that are guided by other internal and external sources. The findings suggest that the content and quantity of user reviews influence developers' resource allocation to exploitation and exploration activities, which in turn impact app market success. More specifically, the findings support the notion that developers' design decisions are more likely to attend to issues that affect a large number of users (review concurrence), i.e., review concurrence positively influences exploitation-exploration orientation. Further, the findings suggest that more reviews following a release engage developers and provide them with the information they can use to improve the apps using users' feedback, thereby increasing their exploitation activities. However, beyond a certain volume of reviews, developers are overwhelmed by the amount of information and become incapable of processing the review issues. This finding supports the notion that the number of reviews has a curvilinear relationship with exploitation-exploration orientation. We further find that the relationship is negatively moderated by review sentiment. In addition, and consistent with the principles of learning and innovation (March, 1991), we find that an appropriate mix of exploitation to exploration activities leads to an increased likelihood of app survival in the market.

5.1 Theoretical Contributions

From a theoretical standpoint, our findings contribute first to the IS development literature. Mobile apps represent the largest segment of adopted systems with nearly five billion users, and their widespread proliferation is increasing over time. In 2020, approximately 142.9 billion apps were downloaded (Iqbal, 2021). Yet despite the importance and prevalence of these systems in the IS field, there is surprisingly very little research examining their development, particularly how developers make design decisions and incorporate users' feedback in consecutive updates to increase their app's market success. The large body of previous IS development literature has provided developers with valuable approaches for acquiring users' organizational and enterprise system needs (Fernandez et al., 2017; Tuunanen et al., 2007). However, these approaches mainly rely on direct communication between users and developers (e.g., Appan & Browne, 2010; Gallivan & Keil, 2003; Hartwick & Barki, 2001; Marakas & Elam, 1998; Pitts & Browne, 2007) that are not likely to be useful in the context of app development. App developers lack direct communication with their geographically dispersed, mass heterogeneous user base. Our study extends the IS development literature to the context of mass-distributed apps, and suggests that developers can acquire knowledge of users' needs from their online feedback and that they need to strike an appropriate balance between satisfying users' requests to improve their apps while experimenting with new features.

Further, the present study also contributes to the organizational learning and innovation literature. First, while prior work has largely conceptualized exploitation and exploration according to an innovation's industry-level technological or market trajectory (incremental or radical) (Benner & Tushman, 2003), the present study considers these activities at a firm level according to the probability of their acceptance by the existing customer base. More specifically, we conceptualize exploitation activities as incorporating users' known needs, and exploration activities as incorporating features not suggested by users in the consecutive app releases. Further, we argue that these activities are incompatible in the context of a small size, dynamic, and competitive app development environment, and that app developers have limited resources that necessitate making a tradeoff between these two activities. This is consistent with the view proposed by Gupta et al. (2006) and Luger et al. (2018), who suggest that the organizational context determines whether these two activities are complementary or contradictory.

Second, there is a paucity of prior work investigating what drives the decision to exploit and explore (Raisch et al., 2009). For the most part, prior studies have considered balancing exploration and exploitation as strategic organizational decisions with calculative considerations (Levinthal, 2021). For example, some organizations are exploring alternative investment and competitive strategies such as blockchain as an alternative to centralized IT (Miranda et al., 2022). On the other hand, the decision could also be the result of organizational forms and processes (March, 1991), where for instance, companies like IBM, 3M, and Google give employees the freedom to pursue projects of their personal interests without management involvement (Manso, 2017). However, the decision to engage in exploration or exploitation activities can also be driven by feedback from the external environment. While prior work has examined the role of external feedback in reinforcing organizational decision-making, it has focused on large organizations where there is a distance between the top management decision-makers and the entrepreneurs (e.g., engineers and programmers) (Levinthal, 2021).

We contribute to this body of work on organizational learning by demonstrating the role of the external environment in exploitation and exploration activity decisions, especially for small organizations that are more sensitive to market feedback. Two characteristics differentiate these small organizations from large ones when it comes to exploration and exploitation decisions. First, in contrast to large organizations, small organizations are tightly coupled with the external environment (Adner & Levinthal, 2002). They have a higher selection pressure as they lack resources for long-term survival without achieving market success. Hence, small organizations pay more attention to signals about the performance of their products. Second, in small organizations, the distance between the entrepreneurs (i.e., developers) and their customers is short which allows them to scan the environment for further cues and details beyond the market success of their products. It is typical for entrepreneurs to seek feedback to improve their products and services (Dabas et al., 2021; Elias et al., 2018). The present study contributes to this work by providing evidence regarding how feedback characteristics, i.e., review volume and concurrence, can ultimately affect app developers' development activities. The present study further contributes to this line of study by suggesting that review sentiment can moderate the effects of those feedback characteristics on development activities.

Finally, app development presents a case of products that are tightly coupled with the market (Adner & Levinthal, 2002) and are developed primarily by small organizations. Our findings show that these organizations, while sensitive to positive market feedback, still need to infuse exploration into their activities. In addition, the decision to balance exploitation with exploration is guided by market feedback. This presents a dilemma for these organizations and their members. They need to attend to this feedback but not overly so. Our findings show that bounded rationality provides one constraint to attending to this feedback. Future studies can examine how organizational members can productively limit their attention to external feedback.

5.2 Practical Implications

Our findings have strategic and policy implications for app developers and app stores in the context of app development. For app developers, the findings demonstrate that app developers need to exploit users' reviews *and* explore new innovative features in order to succeed in these competitive markets. This finding should guide developers to conduct a mix of exploitation and exploration. The specific mix of these activities will likely rely on numerous contextual variables (such as app category, user base size, and free or purchased app as identified in Table 3).

Our findings regarding user feedback suggest that app developers should regularly collect and analyze user data to understand user behavior and preferences, which can inform both exploitation and exploration efforts. They can leverage user feedback and analytics such as review volumes, concurrence, and sentiment to identify areas for improvement and to gather insights for potential new features. Since a typical app is generally developed by a handful of developers who have limited resources (Nayebi et al., 2016), they can develop a strategic roadmap that outlines both short- and long-term goals for their apps so they can effectively allocate their limited resources. Specifically, app developers can design apps with a modular and scalable architecture which will allow them to implement A/B testing and experimentation frameworks to systematically test new features and variations with a subset of users (Xu and Chen, 2016). By using gathered empirical evidence to create different versions of an app to reflect varying levels of exploitation to exploration, these developers are rewarded with quicker feedback regarding the 'optimal' mix of exploitation and exploration in their app updates. Given the short time horizon in app updates, this flexibility could be crucial for success in a competitive market.

Further, the finding that app developers' exploitation and exploration activities are influenced by the information they receive from users' reviews could potentially draw developers' attention to the role of information and bounded rationality in their app development activities. While boundedly rational individuals use information to make rational satisfactory decisions (Gigerenzer, 2004), an overload of information is likely to diminish their ability to process the information (Simon, 1956). As such, app developers need to be mindful of the potential for cognitive overload as they draw on users' feedback to update their apps and may wish to use text review mining tools or AI to help them curate and organize users' reviews.

App store management can also benefit from our findings. Store management can implement policies to attract both developers and users by helping practitioners to balance both exploitation and exploration. They can provide analytics tools to track user behavior, engagement metrics, and conversion rates and analyze this data to gain insights into user preferences and identify and highlight areas for improvement. They can create systems to facilitate feedback about user experience.

The balance between exploitation and exploration is a critical consideration in policy and strategy discussions across various domains including business, technology, and innovation. Our findings provide insights for these fields as well.

5.3 Study Limitations and Future Research Opportunities

A potential limitation of this study is the biased selection of studied apps. Given our objective of examining the effect of user reviews on app evolution, we excluded apps with less than 10 releases, and then randomly selected 10% of those given the manual effort required to collect user reviews from 3rd party aggregators. Yet, the randomly selected apps were not significantly different from those excluded in terms of average rating (t =-0.578, P=0.564) or average length of release update (t = -1.389, P=0.165), indicating both groups had a similar profile. Future research may validate this study's findings using a different and larger sample of applications.

This study also focused on apps in open marketplaces. App development in other contexts, such as enterprise, may differ given the lesser role of external feedback. Furthermore, as software marketplaces are becoming the norm for distributing all kinds of applications (mobile, desktop, and server), it is interesting to test the limitations of our findings in these other contexts. The results of some control variables are also interesting and could stimulate the study of other online review variables that could influence exploitation-exploration orientation. For example, Table 4 above shows that *Review Length* positively affects app developers' exploitation activities, while *Review Velocity* does not. One possible explanation that can be examined is that app developers normally release regular updates rather than ad hoc updates and their decisions are less likely to be affected by review velocity in our studied context. Further, future research may want to explore the alternate forces that shape mobile app evolution including the desire to imitate successful apps (Wang et al., 2018) and the desire to stand out from the competition by introducing distinctive features.

The findings of this study may also be extended to other IS development contexts. This study found that app developers need to use a mix of exploitation and exploration to evolve successful apps. In contrast, traditional IS development has focused on exploiting users' requirements and needs. To our knowledge, there is no research that has examined the effect of exploration on traditional IS development success. Future research could investigate the generalizability of this study's findings in enterprise IS development.

6 References

Adner, R., & Levinthal, D. (2008). Doing versus seeing: Acts of exploitation and perceptions of exploration. *Strategic Entrepreneurship Journal, 2*(1), 43-52.

Adner, R., & Levinthal, D. A. (2002). The emergence of emerging technologies. *California Management Review, 45*(1), 55-66.

Agarwal, R., & Tiwana, A. (2015). Editorial — evolvable systems: Through the looking glass of IS. *Information Systems Research, 26*(3), 473-479. http://dx.doi.org/10.1287/isre.2015.0595

Al-Subaihin, A. A., Sarro, F., Black, S., Capra, L., & Harman, M. (2021). App store effects on software engineering practices. *IEEE Transactions on Software Engineering, 47*(2), 300-319. https://doi.org/10.1109/TSE.2019.2891715

Allon, G., Askalidis, G., Berry, R., Immorlica, N., Moon, K., & Singh, A. (2022). When to be agile: Ratings and version updates in mobile apps. *Management Science*, *68*(6), 4261-4278.

Andriopoulos, C., & Lewis, M. W. (2009). Exploitation-exploration tensions and organizational ambidexterity: Managing paradoxes of innovation. *Organization Science, 20*(4), 696-717.

Appan, R., & Browne, G. J. (2010). Investigating retrieval-induced forgetting during information requirements determination. *Journal of the Association for Information Systems, 11*(5), 250-275. https://doi.org/10.17705/1jais.00228

Arellano, M., & Bond, S. (1991). Some tests of specification for panel data: Monte Carlo evidence and an application to employment equations. *The Review of Economic Studies, 58*(2), 277-297.

Benner, M. J., & Tushman, M. (2002). Process management and technological innovation: A longitudinal study of the photography and paint industries. *Administrative Science*

Quarterly, 47(4), 676-707.

Benner, M. J., & Tushman, M. L. (2003). Exploitation, exploration, and process management: The productivity dilemma revisited. *Academy of Management Review, 28*(2), 238-256.

Bird, S. (2006). NLTK: the natural language toolkit. COLING/ACL 2006 Interactive Presentation Sessions, Sydney.

Borck, J., Caminade, J., & Wartburg, M. v. (2021). *A global perspective on the Apple app sttore ecosystem: An exploration of small businesses within the App Store ecosystem*. https://www.apple.com/newsroom/pdfs/apple-app-store-study-2020.pdf

Boudreau, K. J. (2012). Let a thousand flowers bloom? An early look at large numbers of software app developers and patterns of innovation. *Organization Science, 23*(5), 1409-1427.

Bragge, J., & Merisalo-Rantanen, H. (2002). Engineering e-collaboration processes to obtain innovative end-user feedback on advanced web-based information systems. *Journal of the Association for Information Systems, 10*(3), 196-22.

Brunswicker, S., Almirall, E., & Majchrzak, A. (2019). Optimizing and satisficing: The interplay between platform architecture and producers' design strategies for platform Performance *MIS Quarterly, 43*(4), 1249-1277.

Buchanan, J., & Kock, N. (2001). Information overload: A decision making perspective. In *Multiple Criteria Decision Making in the New Millennium* (pp. 49-58). Springer, Berlin, Heidelberg.

Castillo, A., Benitez, J., Llorens, J., & Braojos, J. (2021). Impact of social media on the firm's knowledge exploration and knowledge exploitation: The role of business analytics talent. *Journal of the Association for Information Systems, 22*(5), 1472-1508. https://aisel.aisnet.org/jais/vol22/iss5/1

Chen, H., De, P., & Hu, Y. J. (2015). IT-enabled broadcasting in social media: An empirical study of artists' activities and music sales. *Information Systems Research, 26*(3), 513-531.

Cyert, R. M., & March, J. G. (1963). *A behavioral theory of the firm*. New York: Prentice-Hall.

Dabas, S., Sharma, S., & Manaktola, K. (2021). Adoption of digital marketing tools in independent businesses: Experiences of restaurant entrepreneurs in India and United Kingdom. *Worldwide Hospitality and Tourism Themes, 13*(2), 214-235. https://doi.org/10.1108/WHATT-09-2020-0120

de Weck, O. L., Roos, D., & Magee, C. L. (2011). *Engineering systems: Meeting human needs in a complex technological world*. MIT Press, Cambridge, Massachusetts.

Demoulin, N. T., & Coussement, K. (2020). Acceptance of text-mining systems: The signaling role of information quality. *Information & Management, 57*(1), 103-120.

Duan, W., Gu, B., & Whinston, A. B. (2008). Do online reviews matter? An empirical investigation of panel data. *Decision Support Systems, 45*(4), 1007-1016.

Elias, S. R. S. T. A., Chiles, T. H., Duncan, C. M., & Vultee, D. M. (2018). The aesthetics of entrepreneurship: How arts entrepreneurs and their customers co-create aesthetic value. *Organization Studies, 39*(2-3), 345–372. https://doi.org/10.1177/0170840617717548

Fader, P. S., Hardie, B. G. S., & Lee, K. L. (2005). “Counting your customers” the easy way: An alternative to the Pareto/NBD model. *Marketing Science, 24*(2), 275-284.

Fernandez, D. M., Wagner, S., Kalinowski, M., Felderer, M., Mafra, P., & Vetr. (2017). Naming the pain in requirements engineering. *Empirical Software Engineering, 22*(5), 2298-2338. https://doi.org/10.1007/s10664-016-9451-7

Gallivan, M. J., & Keil, M. (2003). The user-developer communication process: a critical case study. *Information Systems Journal, 13*(1), 37-68. https://doi.org/10.1046/j.1365-2575.2003.00138.x

Gigerenzer, G. (2004). Fast and frugal heuristics: The tools of bounded rationality. *Blackwell Handbook of Judgment and Decision Making*, 62–88.

Gigerenzer, G., & Gaissmaier, W. (2011). Heuristic decision making. *Annual Review of Psychology*, *62*(1), 451-482.

Greenwood, B. N., & Gopal, A. (2015). Research note—Tigerblood: Newspapers, blogs, and the founding of information technology firms. *Information Systems Research, 26*(4), 812-828.

Greve, H. R. (2007). Exploration and exploitation in product innovation. *Industrial and Corporate Change, 16*(5), 945-975.

Gupta, A. K., Smith, K. G., & Shalley, C. E. (2006). The interplay between exploration and exploitation. *Academy of Management Journal, 49*(4), 693-706. https://doi.org/10.5465/AMJ.2006.22083026

Haas, M. R., Criscuolo, P., & George, G. (2015). Which problems to solve? Online knowledge sharing and attention allocation in organizations. *Academy of Management Journal, 58*(3), 680-711. https://doi.org/10.5465/amj.2013.0263

Hahn, M., Lawson, R., & Lee, Y. G. (1992). The effects of time pressure and information load on decision quality. *Psychology & Marketing*, *9*(5), 365-378.

Harkins, S. G., & Petty, R. E. (1987). Information utility and the multiple source effect. . *Journal of Personality and Social Psychology, 52*(2), 260.

Hartwick, J., & Barki, H. (2001). Communication as a dimension of user participation. *Professional Communication, IEEE Transactions on, 44*(1), 21-36.

Hays, K. (2022). Facebook and Mark Zuckerberg just raised a giant middle finger to Wall Street. The company says its metaverse business will lose even more money next year. *Business Insider*. Retrieved December 21, 2022 from https://www.businessinsider.com/facebook-doubles-down-on-metaverse-spending-despite-calls-cut-costs-2022-10.

He, Z.-L., & Wong, P.-K. (2004). Exploration vs. exploitation: An empirical test of the ambidexterity hypothesis. *Organization Science, 15*(4), 481-494

Herbig, P. A., & Kramer, H. (1994). The effect of information overload on the innovation choice process: Innovation overload. *Journal of Consumer Marketing, 11*(2), 45-54.

Iacob, C., & Harrison, R. (2013). Retrieving and analyzing mobile apps feature requests from online reviews. *IEEE International Working Conference on Mining Software Repositories*, 41-44. https://doi.org/10.1109/MSR.2013.6624001

Iqbal, M. (2021). *App download and uage statistics*. Retrieved December 2021 from https://www.businessofapps.com/data/app-statistics/

Jabr, W., & Zheng, Z. (2014). Know yourself and know your enemy. *MIS Quarterly, 38*(3), 635-654.

Jiang, H., Zhang, J., Li, X., Ren, Z., Lo, D., Wu, X., & Luo, Z. (2019). Recommending new features from mobile app descriptions. *ACM Transactions on Software Engineering and Methodology, 28*(4), 1-29. https://doi.org/10.1145/3344158

Junni, P., Sarala, R. M., Taras, V. A. S., & Tarba, S. Y. (2013). Organizational ambidexterity and performance: A meta-analysis. *Academy of Management Perspectives, 27*(4), 299-312.

Kane, G. C., & Alavi, M. (2007). Information technology and organizational learning: An investigation of exploration and exploitation processes. *Organization Science, 18*(5), 796-812.

Katila, R., & Ahuja, G. (2002). Something old, something new: A longitudinal study of search behavior and new product introduction. *Academy of Management Journal, 45*(6), 1183-1194.

Koroteev, M. V. (2021). BERT: A review of applications in natural language processing and understanding. *arXiv preprint arXiv:2103.11943*.

Lavie, D., Stettner, U., & Tushman, M. L. (2010). Exploration and exploitation within and across organizations. *Academy of Management Annals, 4*(1), 109-155.

Lee, G., & Raghu, T. S. (2014). Determinants of mobile apps' success: Evidence from the app sttore market. *Journal of Management Information Systems, 31*(2), 133-169. https://doi.org/10.2753/MIS0742-1222310206

Lejarraga, J., & Pindard-Lejarraga, M. (2020). Bounded rationality: Cognitive limitations or adaptation to the environment? The implications of ecological rationality for management learning. *Academy of Management Learning & Education*, *19*(3), 289-306.

Levinthal, D. A. (2021). *Evolutionary processes and organizational adaptation: A mendelian perspective on strategic management*. Oxford University Press, Oxford, UK.

Levinthal, D. A., & March, J. G. (1993). The myopia of learning. *Strategic Management Journal, 14*(S2), 95-112. https://doi.org/10.1002/smj.4250141009

Liu, T., Wang, C., Huang, K., Liang, P., Zhang, B., Daneva, M., & van Sinderen, M. (2023). RoseMatcher: Identifying the impact of user reviews on app updates. *Information and Software Technology*, *161*, 107261. https://doi.org/10.1016/j.infsof.2023.107261

Liu, Y. (2006). Word of mouth for movies: Its dynamics and impact on box office revenue. *Journal of Marketing, 70*(3), 74-89.

Luger, J., Raisch, S., & Schimmer, M. (2018). Dynamic Balancing of Exploration and Exploitation: The Contingent Benefits of Ambidexterity. *Organization Science, 29*(3), 449-470.

Malhotra, N.K. (1984): Reflections on the information overload paradigm in consumer decision making. *Journal of Consumer Research*, *10*(4), 436-440.

Mallipeddi, R. R., Janakiraman, R., Kumar, S., & Gupta, S. (2021). The effects of social media content created by human brands on engagement: Evidence from Indian general election 2014. *Information Systems Research, 32*(1), 212-237.

Manso, G. (2017). Creating incentives for innovation. *California Management Review, 60*(1), 18-32. https://doi.org/10.1177/0008125617725287

Marakas, G. M., & Elam, J. J. (1998). Semantic structuring in analyst acquisition and representation of facts in requirements analysis. *Information Systems Research, 9*(1), 37-63. https://doi.org/10.1287/isre.9.1.37

March, J. G. (1991). Exploration and exploitation in organizational learning. *Organization Science, 2*(1), 71-87.

March, J. G. and Simon, H. A. (1958). *Organizations*. New York: Wiley.

McGrath, R. G. (2001). Exploratory learning, innovative capacity, and managerial oversight. *Academy of Management Journal, 44*(1), 118-131.

McIlroy, S., Ali, N., & Hassan, A. E. (2016). Fresh apps: an empirical study of frequently-updated mobile apps in the Google play store. *Empirical Software Engineering, 21*(3), 1346-1370.

Menon, N. M., & Kohli, R. (2013). Blunting Damocles' sword: A longitudinal model of healthcare IT impact on malpractice insurance premium and quality of patient care. *Information Systems Research, 24*(4), 918-932.

Miranda, S. M., Wang, D. D., & Tian, C. A. (2022). Discursive fields and the diversity-coherence paradox: An ecological perspective on the blockchain community discourse. *MIS Quarterly*, *46*(3), 1421-1452.

Montealegre, R., Iyengar, K., & Sweeney, J. (2019). Understanding ambidexterity: Managing contradictory tensions between exploration and exploitation in the evolution of digital infrastructure. *Journal of the Association for Information Systems, 20*(5), 647-680. https://aisel.aisnet.org/jais/vol20/iss5/1

Nayebi, M., Adams, B., & Ruhe, G. (2016). Release practices for mobile apps -- What do users and developers think? *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER),* 552-562.

Noei, E., Zhang, F., & Zou, Y. (2021). Too many user-reviews! What should app developers look at first?. *IEEE Transactions on Software Engineering*, *47*(2), 367-378. https://doi.org/10.1109/TSE.2019.2893171

O'Reilly, C. A., & Tushman, M. L. (2013). Organizational Ambidexterity: Past, Present, and Future. *Academy of Management Perspectives*, *27*(4), 324–338. https://doi.org/10.5465/amp.2013.0025

Openja, M., Adams, B., & Khomh, F. (2020). Analysis of modern release engineering topics: a large-scale study using stackoverflow. *IEEE international conference on software maintenance and evolution (ICSME)*, 104–114. IEEE.

Pagano, D., & Bruegge, B. (2013). User involvement in software evolution practice: A case study. *Proceedings - International Conference on Software Engineering*, 953-962. https://doi.org/10.1109/ICSE.2013.6606645

Pagano, D., & Maalej, W. (2013). User feedback in the appstore: An empirical study. *2013 21st IEEE International Requirements Engineering Conference, RE 2013 - Proceedings*, 125-134. https://doi.org/10.1109/RE.2013.6636712

Palomba, F., Linares-Vsquez, M., Bavota, G., Oliveto, R., Penta, M. D., Poshyvanyk, D., & Lucia, A. D. (2018). Crowdsourcing user reviews to support the evolution of mobile apps. *Journal of Systems and Software, 137*, 143-162. https://doi.org/10.1016/j.jss.2017.11.043

Park, S. Y., & Allen, J. P. (2013). Responding to online reviews: Problem solving and engagement in hotels. *Cornell Hospitality Quarterly*, *54*(1), 64-73.

Park, Y., Pavlou, P. A., & Saraf, N. (2020). Configurations for achieving organizational ambidexterity with digitization. *Information Systems Research, 31*(4), 1376-1397. https://doi.org/10.1287/isre.2020.0950

Pitts, M. G., & Browne, G. J. (2007). Improving requirements elicitation: An empirical investigation of procedural prompts. *Information Systems Journal, 17*(1), 89-110. https://doi.org/10.1111/j.1365-2575.2006.00240.x

Rabe-Hesketh, S., Skrondal, A., & Pickles, A. (2002). Reliable estimation of generalized linear mixed models using adaptive quadrature. *The Stata Journal, 2*(1), 1-21.

Raisch, S., Birkinshaw, J., Probst, G., & Tushman, M. L. (2009). Organizational ambidexterity: balancing exploitation and exploration for sustained performance. *Organization Science, 20*(4), 685–695. https://doi.org/10.1287/orsc.1090.0428

Ramesh, B., Mohan, K., & Cao, L. (2012). Ambidexterity in agile distributed development: An empirical investigation. *Information Systems Research, 23*(2), 323-339. https://doi.org/10.1287/isre.1110.0351

Reeck, C., Posner, N. A., Mrkva, K., & Johnson, E. J. (2023). Nudging app adoption: Choice architecture facilitates consumer uptake of mobile apps. *Journal of Marketing*, *87*(4) 510-527.

Schmittlein, D. C., Morrison, D. G., & Colombo, R. (1987). Counting your customers: Who-are they and what will they do next? *Management Science, 33*(1), 1-24.

Sheng, J. (2019). Being active in online communications: Firm responsiveness and customer engagement behaviour. *Journal of Interactive Marketing*, *46*(1), 40-51.

Simon, H. A. (1956). Rational choice and the structure of the environment. *Psychological Review, 63*(2), 129.

Simon, H. A. (1973). The structure of ill structured problems. *Artificial Intelligence, 4*(3-4), 181-201. https://doi.org/10.1016/0004-3702(73)90011-8

Simon, H. A. (1979). Rational decision making in business organizations. *The American Economic Review*, *69*(4), 493-513.

Soh, F., & Grover, V. (2020). Effect of release timing of app innovations based on mobile platform innovations. *Journal of Management Information Systems*, *37*(4), 957-987.

Soh, F., & Grover, V. (2022). Leveraging Platform Boundary Resources: The Role of Distributed Sensemaking. *Journal of Management Information Systems*, *39*(2), 366-394.

Temizkan, O., & Kumar, R. L. (2015). *Exploitation and exploration networks in open source software development: An artifact-level analysis*. *Journal of Management Information Systems, 32*(1), 116-150. https://doi.org/10.1080/07421222.2015.1029382

Thakur, R. (2018). Customer engagement and online reviews. *Journal of Retailing and Consumer Services*, *41*, 48-59.

Tiwana, A. (2015). Evolutionary competition in platform ecosystems. *Information Systems Research, 26*(2), 266-281. https://doi.org/10.1287/isre.2015.0573

Torres-Reyna, O. (2007). Panel data analysis fixed and random effects using Stata (v. 4.2). *Data & Statistical Services, Priceton University*.

Tuunanen, T., Rossi, M., Saarinen, T., & Mathiassen, L. (2007). A contingency model for requirements development. *Journal of the Association for Information Systems, 8*(11), 569-597. https://doi.org/10.17705/1jais.00143

Uotila, J., Maula, M., Keil, T., & Zahra, S. A. (2009). Exploration, exploitation, and financial performance: analysis of S&P 500 corporations. *Strategic Management Journal, 30*(2), 221-231. https://doi.org/10.1002/smj.738

Vinekar, V., Slinkman, C. W., & Nerur, S. (2006). Can agile and traditional systems development approaches coexist? An ambidextrous view. *Information Systems Management, 23*(3), 31-42.

Wang, Q., Li, B., & Singh, P. V. (2018). Copycats vs. original mobile apps: A machine learning copycat-detection method and empirical analysis. *Information Systems Research, 29*(2), 273-291.

Wen, W., & Zhu, F. (2019). Threat of platform-owner entry and complementor responses: Evidence from the mobile app market. *Strategic Management Journal, 40*(9), 1336–1367.

Wooldridge, J. M. (2010). *Econometric analysis of cross section and panel data.* MIT Press, Cambridge, MA.

Xu, Y., & Chen, N. (2016). Evaluating mobile apps with A/B and quasi A/B tests. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 313-322. August, 2016.

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. *Empirical Software Engineering*, *27*(2), 1-41.

Ye, H. J., and Kankanhalli, A. . (2020). Value cocreation for service innovation: Examining the relationships between service innovativeness, customer participation, and mobile app performance. *Journal of the Association for Information Systems, 21*(2), 292-311.

Yin, P.-L., Davis, J. P., and Muzyrya, Y. . (2014). Entrepreneurial Innovation: Killer Apps in the IPhone Ecosystem *American Economic Review, 104*(5), 255-259.

Yin, D., Bond, S. D., & Zhang, H. (2017). Keep your cool or let it out: Nonlinear effects of expressed arousal on perceptions of consumer reviews. *Journal of Marketing Research, 54*(3), 447-463.

Zhou, S., Qiao, Z., Du, Q., Wang, G. A., Fan, W., & Yan, X. (2018). Measuring customer agility from online reviews using big data text analytics. *Journal of Management Information Systems, 35*(2), 510-539.

Appendices: The Impact of Feature Exploitation and Exploration on Mobile Application Evolution and Success

Table of Contents

Appendix A: Using Similarity Scores between Mobile App Update Texts and User Reviews as Alternative Main Independent Variable

We define mobile app features described in update descriptions as either exploitation or exploration based on whether these features appeared in app users' reviews. This is a conservative method since only exact matched text content in both texts will be included. This method may not capture the semantic similarity between user reviews and mobile app update descriptions. For example, while the mobile app update description states that "Added an option to display the song remaining time on the player screen," a mobile app review says that "it would be better to show playing duration of the song." Although both texts discussed the same topic of displaying song playing time, our conservative approach, exact matching, will not capture the exploitive side of user reviews.

A-1. Semantic Similarity Score

To address this concern and also test the robustness of our findings, we implemented BERT (Bidirectional Encoder Representations for Transformers) to capture the semantic similarity of these two texts. The output of this process is a semantic similarity score ranges between 0 (absolutely not similar) and 1 (almost identical). The higher a score is, the more the app developer is engaged in exploitation activities. In the given example above, the similarity score of 0.803 shows that these two texts likely talk about the same content.

BERT is a state-of-the-art deep learning approach for many natural language processing (NLP) tasks including measuring the semantic similarity of texts (see Koroteev, 2021for a review). This method has been widely implemented by IS researchers for tasks such as identifying whether a text is written by medical professionals (Mousavi, Raghu, & Frey, 2020), defining media text complexity (Shin, He, Lee, Whinston, Cetintas, & Lee, 2020), and classifying whether a text is a fake news (Wei, Zhang, Zhang, Chen, & Zeng, 2020).

BERT is a language representation model introduced by Devlin et al. (2018). It uses Transformers to learn contextual relationships between tokens (e.g. words) in a text document. Transformers are models that process words in relation to all the other words in a text. That is, BERT models consider the full context of a word by looking at the words that come before and after it. BERT is a pre-trained model and trained and built using text data from both Wikipedia and BookCorpus. For each word, BERT model will return a vector representation of words. These representations will be used in NLP tasks. The details about BERT models, model training, and applications can be found in Devlin et al. (2018)1 and Reimers and Gurevych (2019).[2]

In our study, we are interested in comparing the semantic similarity between mobile app update descriptions and user reviews. Figure 1 shows the generating of semantic representation and similarity score process used in this study. In step (1), we implemented natural language processing techniques to process our review and mobile app update data. Specifically, we first cleaned all texts (e.g., removing html tags) and converted cleaned review and app update texts into lowercase. In step (2), we used BERT Base model to generate vector representations for both reviews and mobile app update texts. BERT Base returns a vector of length 768 for each word in each text. These vectors represent the semantic meaning of the word. In step (3), the vector representations of two texts were then used in cosine similarity task to generate a similarity score between 0 (absolutely not similar) and 1 (almost identical).

[1] https://github.com/google-research/bert
[2] https://github.com/UKPLab/sentence-transformers

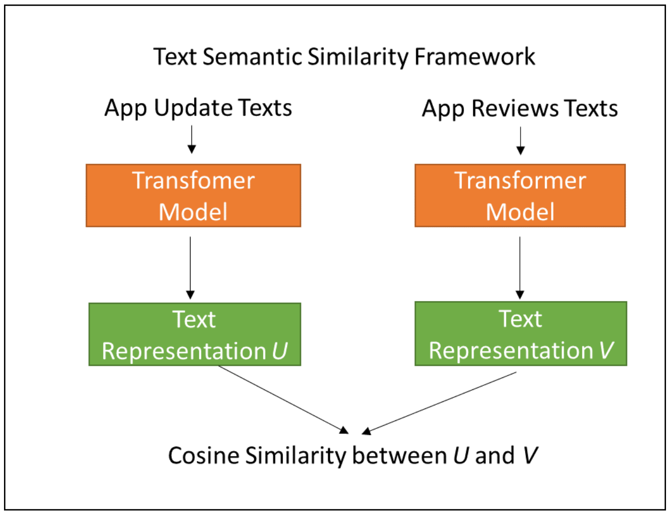

Figure A-1. Process to Generate Semantic Similarity Scores between Mobile App Updates and User Reviews

A-2. Similarity Verification Using Amazon Mechanical Turk (AMT)

We used Amazon Mechanical Turk (AMT) service to verify our similarity analysis approach by examining the correlation between similarity scores of a pair of texts obtained from our approach and AMT workers' perception about the similarity between these texts. Recall that similarity scores are computed based on automated, easily scalable methods. Thus, it is possible that either humans disagree with our metrics, or our measures do not capture the semantic similarity in the same manner as most humans.

We started with creating pairs of text samples. We first calculated similarity scores between mobile app update texts for a new release and review texts given by app users for the period between a previous release and a new release. Then, we split pairs of these texts into four quartiles based on quartiles of similarity scores. Next, we randomly extracted 20 pairs from each quartile as coding samples.

We recruited 240 Amazon Mechanic Turk workers (i.e., Turkers) residing in the United States. We restricted participation to those who had at least 500 approved hits and 95% approval ratings. First, all participants were instructed to read a pair of randomly assigned texts. After reading these texts, these participants needed to rate the similarity between the pair of texts using a 5-point Likert scale by performing the task of "please rate the similarity between these two texts from highest (5-absolutely similar) to the lowest (1-absolutely not similar)." Each pair of texts is randomly assigned to three Turkers. A Pearson's correlation analysis shows a positive, significant correlation between our semantic similarity scores and Turkers' perceived similarity for pairs of texts (0.73, $p < 0.01$). This result shows that our semantic similarity analysis achieves our purpose.

A-3. Analysis and Results

We repeat our analyses in section 4.1 that studies the relationship between exploitation-exploration orientation and mobile app survival using this newly created measurement. Specifically, we replace *Update Exploitation, Exploitation-Exploration Orientation* and its quadratic term in our main analysis with this newly created measurement (i.e., *Review-Update Similarity*) and its quadratic term. The results in Table A-1 show that the coefficients of *Review-Update Similarity* are positive and statistics and coefficients of its quadratic terms are negative and statistically significant. Specifically, we find that semantic similarity between app update descriptions and app user reviews to be positively associated with app short and long-term survival up to a threshold, after which this positive relationship declines. These results further confirm our main findings and support H1

Table A-1. Regression of App Survival on Review-Update Similarity and its Quadratic Terms

Variables	1 month	3 years
Review-Update Similarity	0.3026***	0.2723***
	(0.1078)	(0.0898)
Review-Update Similarity× Review-Update Similarity	-0.4167**	-0.3730**
	(0.2089)	(0.1741)
# of Reviews(k)	-0.0125**	-0.0107**
	(0.0053)	(0.0044)
Avg Rating	-0.0789***	-0.0885***
	(0.0251)	(0.0209)
Avg Sentiment	0.1098	0.1440**
	(0.0705)	(0.0587)
Review Length	-0.0026***	-0.0024***
	(0.0003)	(0.0002)
Review Velocity	-0.0109*	-0.0099*
	(0.0062)	(0.0052)
Update Length	0.0069	0.0060
	(0.0092)	(0.0076)
Business	-0.2760***	-0.2332***
	(0.0201)	(0.0167)
Entertainment	0.0053	0.0023
	(0.0116)	(0.0096)
User Base(<1 million)	0.0298	0.0405**
	(0.0196)	(0.0163)
User Base(1-10 million)	-0.0901***	-0.0658***
	(0.0132)	(0.0110)
In-app Purchase	-0.0290***	-0.0217**
	(0.0108)	(0.0090)
Constant	1.5521***	1.3997***
	(0.1183)	(0.0986)
Observations	1,952	1,952
Android Version Dummies	YES	YES

Standard errors in parentheses
*** p<0.01, * *p<0.05, * p<0.1

Appendix B: Robustness Tests: Effects of Orientation on Alternative Dependent Variables

We test the robustness of our findings using alternative dependent variables (users' app ratings and their sentiments regarding the app). Table B-1 reports the results. Coefficients of *Exploitation-Exploration Orientation* in both models are positive and statistically significant at different levels. Coefficients of its quadratic term in both models are all negative and statistically significant at different levels. These results show that app developers who address users' review content are more likely to receive higher ratings and positive sentiment from users in the marketplace, and these relationships are non-linear. These results are consistent with our findings in the main analysis.

Table B-1. Regression of User Rating and Sentiment on Exploitation Orientation (Alternative Dependent Variables)

Variables	User's Rating Change as DV	User's Sentiment Change as DV
Exploitation-Exploration Orientation	7.4695***	0.2130*
	(0.3419)	(0.1160)
Exploitation-Exploration Orientation ^{2}	-3.0952***	-0.2191*
	(0.3701)	(0.1255)
Update Exploitation	0.3024***	0.0055
	(0.0384)	(0.0130)
Avg # of Reviews(k)	0.3162***	0.0003
	(0.0267)	(0.0091)
Avg Rating	0.0211	0.0040
	(0.1219)	(0.0413)
Avg Sentiment	-0.0339	-0.0277
	(0.3409)	(0.1156)
Review Length	-0.0123***	0.0000
	(0.0013)	(0.0004)
Review Velocity	-0.1541***	-0.0040
	(0.0484)	(0.0164)
Update Length	0.1730***	0.0124
	(0.0303)	(0.0103)
Business	0.0986	0.0055
	(0.0963)	(0.0327)
Entertainment	-0.1221**	-0.0033

	(0.0558)	(0.0189)
User Base(<1 million)	-0.7491***	-0.0092
	(0.0949)	(0.0322)
User Base(1-10 million)	-0.3500***	-0.0078
	(0.0633)	(0.0215)
In-app Purchase	-0.0567	-0.0018
	(0.0517)	(0.0175)
Constant	3.5331***	0.1009
	(0.5873)	(0.1992)
Observations	1,833	1,833
R-squared	0.8060	0.0114
Android Version Dummies	YES	YES

Standard errors in parentheses
*** p<0.01, * *p<0.05, * p<0.1

Appendix C: Review Content and App Updates

C-1. Review Content Analysis

User feedback is an important information source for developers to improve mobile application quality (Groen et al. 2017). Marketplaces such as AppStore and PlayStore allow users to submit feedback in the form of reviews for mobile applications. These user reviews contain valuable information such as bug reports, feature requests, and feature evaluations that is helpful for developers to improve their apps (Chen et al. 2014). However, the enormous volume of reviews received for an app makes the manual analysis of these reviews unpractical. We, therefore, implemented automatic approaches to identify user intentions and extract app features from user app reviews.

C-2. Identifying User Intentions

We first identify user's review intentions. Several approaches have been proposed to automate the process of identifying users' intentions (see Dąbrowski, et al., 2022 for an excellent review). These approaches include tools such as MARA (Mobile App Review Analyzer) (Iacob & Harrison 2013) to rank informative reviews or SURF (Summarizer of User Reviews Feedback) to summarize reviews (Di Sorbo et al. 2017). In this study, we adopted ARdoc (App Reviews Development Oriented Classifier), one of the most well-accepted approaches for identifying user review intentions (Panichella et al. 2016).

ARdoc, a mobile app reviews classifier, is developed based on the taxonomy developed by Panichella et al. (2015) to identify user intentions. It combines natural language processing (NLP) and text analysis techniques to automatically mine intentions in user reviews. We chose this approach over other methods because this approach not only satisfies our objective of identifying user intentions from app user reviews, but is also validated through real-world user studies (app developers) and achieving high precision and recall (over 85% in both metrics).

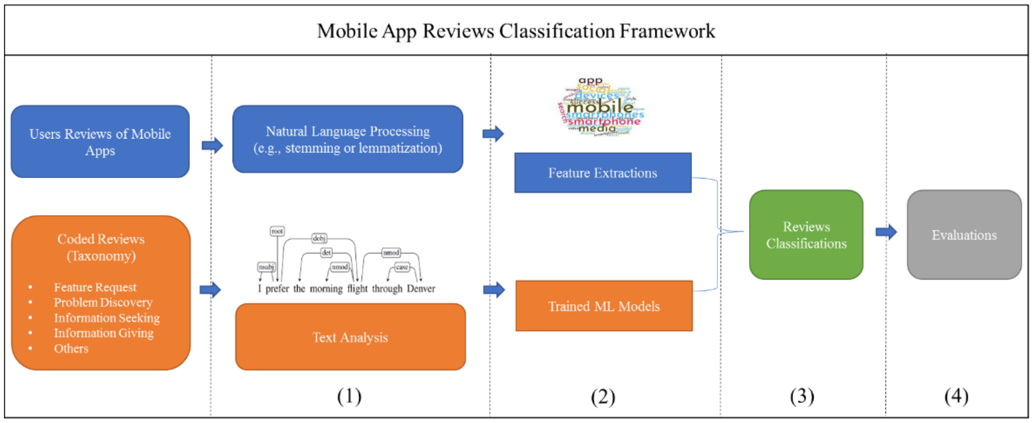

Figure C-1

We follow the study of Panichella et al. (2016) to classify user app reviews into review categories. Figure 1 shows the app review classification process used in this study. In step (1), we implemented natural language processing techniques to process our review data. Specifically, we first converted each review text into lowercase and then broke it up into individual sentences. Then, we turned words in each sentence into their base form (i.e., stemming or lemmatization). Finally, we removed function and stop words. At the same time, we performed text analysis to create NLP heuristics from coded review data provided by Panichella et al. (2016). In step (2), we extracted text features from processed review texts and built machine learning (ML) models using NLP heuristics extracted from step (1). In step (3), we used the built ML model to classify each review sentence into one of the review categories (i.e., feature request, problem discovery, information seeking, information giving, or other). We implemented ARdoc Java API[3] till this step. Finally, we used Amazon Mechanical Turk (AMT) to evaluate classification outcomes for feature requests and problem discovery since these two categories are the focus of this analysis.

[3] https://spanichella.github.io/tools.html#ardoc

When we evaluated classification outcomes, we first randomly extracted 600 review sentences classified as either feature request or problem discovery (300 sentences in each category). Then, we hired workers from AMK to answer one of the following questions after reading a randomly assigned review sentence. If the sentence is classified as problem discovery category, a worker assigned to this sentence needs to answer the question of: “Is this review for a mobile app talking about issue(s) in the mobile app?” If the sentence is classified as feature request category, a worker assigned to this sentence needs to answer the question of: “Is this review for a mobile app requesting feature(s) to be added to this mobile app?” Each sentence is randomly assigned to two workers, and each worker can only view one sentence from each classification category. We conducted Cohen's inter-rater reliability test for outcomes and obtained Kappa coefficient (κ) of 0.8, which shows substantial agreement between raters. 76.7% of AMK workers agree with our identified user intentions. Table 1 provides explanations and review distributions for each review category.

Table C-1 Review Categories Definitions, Examples, and Distributions

Types of Reviews	Explanations	Examples	#
Feature Request	sentences expressing ideas, suggestions or needs for improving or enhancing the app or its functionalities.	• “It would be nice if you could buy the ticket through the web site and then save it in My Trips.” • “Needs more message colors”	55,262
Problem Discovery	sentences describing issues with the app or unexpected behaviors.	• “Guest rewards section is unusable, crashes” • “The arrival time/status function doesn't work.”	100,666

Information Seeking	sentences related to attempts to obtain information or help from other users or developers.	• "How do I sign out help me?" • "how to use this apps, still trying"	21,945
Information Giving	sentences that inform or update users or developers about an aspect related to the app.	• "I travel from Irvine to San Diego weekly and I love this app!" • "Menus, ability to order online, what's open, what services available, and ratings."	80,989
Other	Sentences do not provide any useful feedback to developers	• "Thanks for a great app." • "Very useful app"	1,542,485

We further analyze review sentences classified as problem discovery to obtain more insights about what problems user reviews disclose. We first implemented a natural language processing procedure to clean review texts (e.g., removing URL and stop and functional words). Then, we find frequencies of uni- and bi-gram within each review sentence. Finally, we rank words/word phrases based on their frequencies. Figure 2 shows a word cloud of the top 100 words/word phrases that appeared in reviews classified as bug reports. Most frequent words or word phrases in user reviews reported issues such as freezes, crashing, bug, lost updates, and stop working.

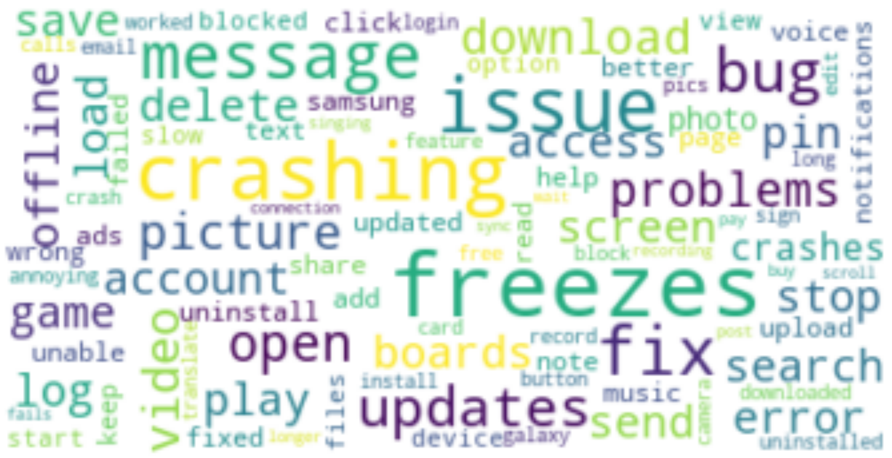

Figure C-2

C-3. App Feature Extractions

We next extracted app features from app user reviews and app description texts. Several studies implemented classification approaches such as topic modeling (Guzman and Maalej 2014) and supervised tagging models (e.g., SUR-miner from Gu and Kim, 2015, and AR-miner from Chen el at., 2014) to automatically classify app features. However, Maalej and Nabil (2015) suggest that extracting such complex features may not be necessary and comparable results could be obtained by using simple lexical characteristics. In this study, we implemented SAFE (Simple Approach for Feature Extraction), a rule-based approach, to extract app features (Johann, Stanik, and Maalej 2017). Johann et al. (2017) use 18 Part-of-Speech (POS) patterns and five sentence patterns for automatically extracting app features from app descriptions and app reviews and show better performance than other previously proposed approaches. We follow their procedure specified in Figure 3 to extract features from user app reviews and app descriptions. We implemented the Python SAFE package for this process and identified total of 69,288 unique features.[4] Among sentences classified as feature requests, top 20 most frequently requested

[4] https://github.com/faizalishah/SAFE_REPLICATION

features are shown in Figure 3.

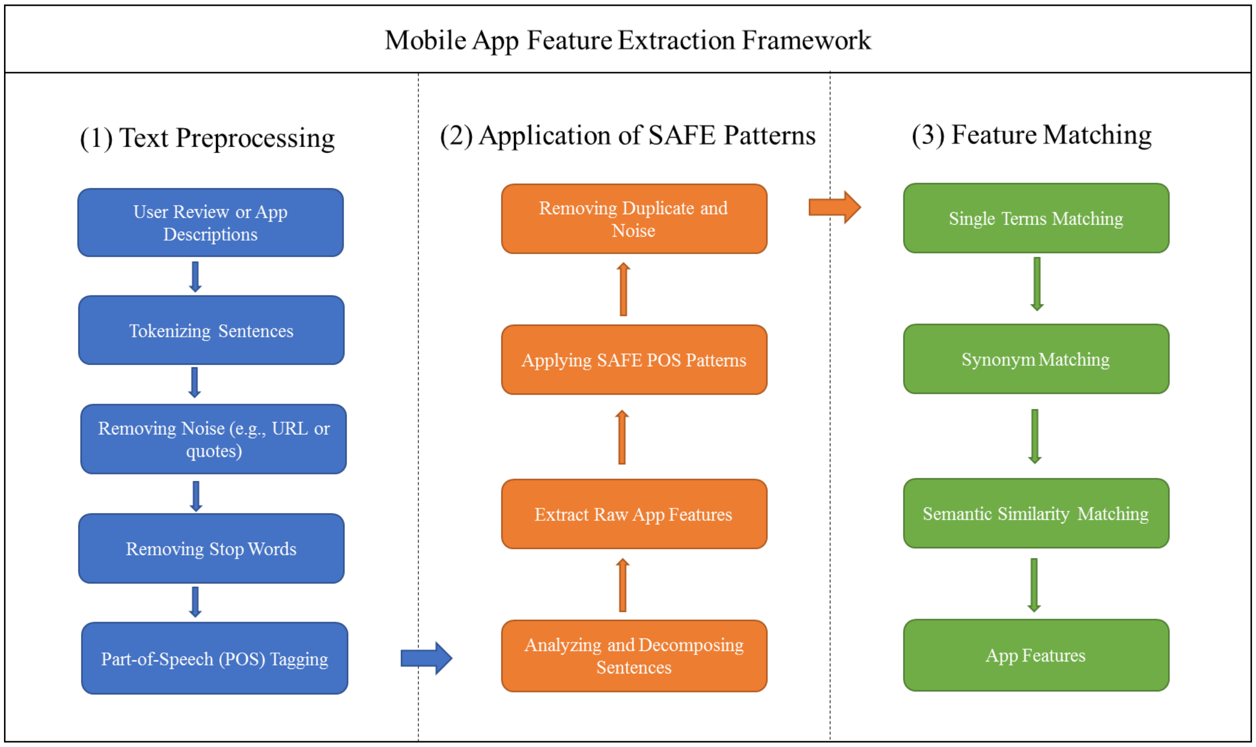
Mobile App Feature Extraction Framework
(1) Text Preprocessing
User Review or App Descriptions
Tokenizing Sentences
Removing Noise (e.g., URL or quotes)
Removing Stop Words
Part-of-Speech (POS) Tagging
(2) Application of SAFE Patterns
Analyzing and Decomposing Sentences
Extract Raw App Features
Applying SAFE POS Patterns
Removing Duplicate and Noise
(3) Feature Matching
Single Terms Matching
Synonym Matching
Semantic Similarity Matching
App Features

Figure C-3

After extracting features from review sentences, we verified extracted results using workers from Amazon Mechanical Turk. Specifically, we created a random number generator and then randomly chose 600 sentences with extracted features. Then, we randomly assigned each sentence to two different workers and asked each worker to verify whether the identified app features in the sentence are focal features. Our Cohen's inter-rater reliability test shows substantial agreement between raters with Kappa coefficient (κ) of 0.74. Over 78.6% of AMK workers agree with our identified app features. Figure C-4 shows top 20 features mentioned in user reviews.

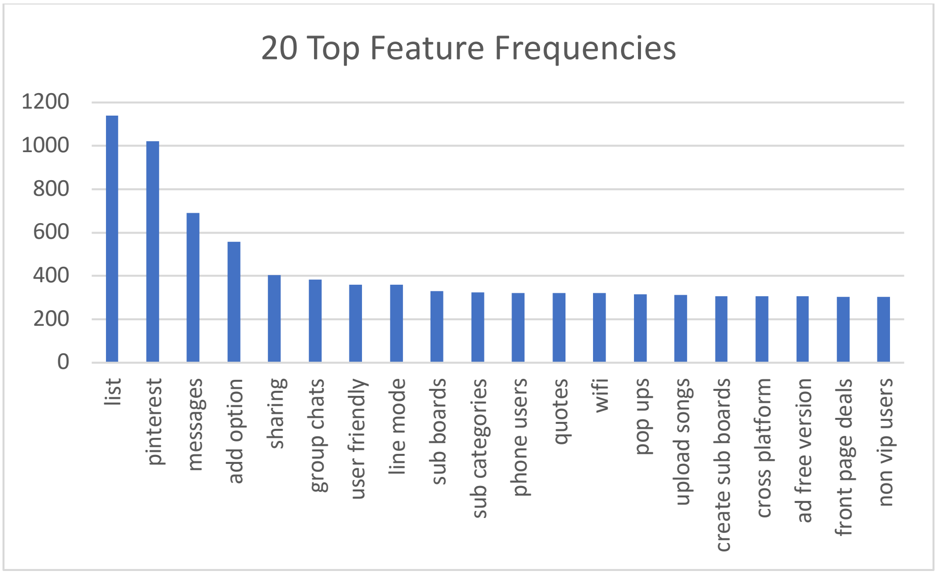

Figure C-4

We also explore how app developers consider app features that appear in user reviews. Since app developers are unlikely to include all app features mentioned in user reviews, they likely prioritize some features over others. We focus on features that had issues or were requested by users. We consider that app developers are more likely to first update app features that address app users' common concerns rather than include new features. We test this conjecture using all app revision update data, and app features that appeared in user reviews that are classified as either problem discovery or feature request categories. There is a total of 20,645 features. The analysis is at the app version-feature level. The dependent variable is binary, indicating whether a feature was included in the app's next update. The main independent variables include feature category (*Feature Categories*, which equals 1 when the type of a feature is problem discovery and 0 when the type of a feature is a requested feature), the frequency of this feature (*Feature Frequencies*), and the interaction of these two independent variables. We estimate a regression model with app version fixed effects.

Table C-2 reports the results. The coefficients of all variables are positive and statistically significant at a 0.01 level. These results show that app developers are more likely to include features that are related to app issues than features requested by users (0.3641, p<0.001) and features that are frequently mentioned by app users (0.0010, p<0.001) in revisions. Most importantly, we find that app developers are more likely to include features more frequently mentioned by users among app bug features. The plot in Figure A-5 shows the interaction term between *Feature Categories* and *Feature Frequencies* and further confirms our findings.

Table C-2. Effects of Feature Categories, Frequencies, and Their Interaction on Feature Inclusion

Variables	Coefficients	SE
Feature Categories	0.3641***	0.0223
Feature Frequencies	0.0010***	0.0002
Feature Categories× Feature Frequencies	0.0062***	0.0006
Observations	145,118	
App Version Fixed Effects	YES	

*** p<0.01

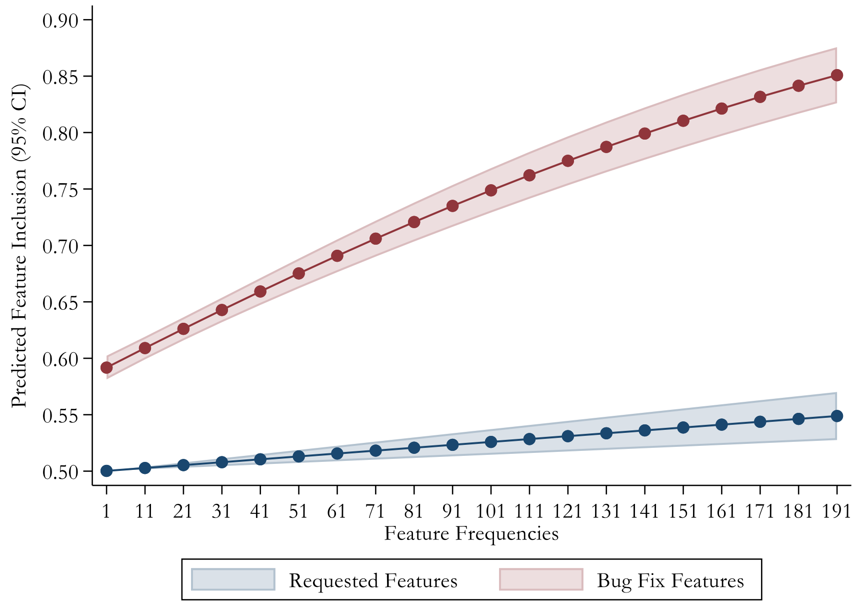

Figure C-5

Appendix D: Robustness Tests (Alternative *Exploitation-Exploration Orientation*)
D-1. The Effects of Exploration Orientation on Survival

We tested the effect of developers' *Exploitation-Exploration Orientation* on apps' success measured by their survival in the marketplace over time (H1) in our main analysis, and found that *Exploitation-Exploration Orientation* has an inverted U-shape relationship with mobile app survival. *Exploitation-Exploration Orientation* (i.e., *Exploitation Orientation*) thus far is measured from an exploitation perspective in Table 2 as a result of our intent to examine the effects of user reviews on app developers. We recreate this measure from an exploration perspective:

$$Exploitation - Exploration\ Orientation\ (i.e., Exploration\ Orientation) = \frac{Update\ Exploration}{Update\ Exploitation + Update\ Exploration}$$

Since exploration and exploitation orientations have a perfect negative correlation as shown in Figure D-1, we expect that exploitation orientation also has an inverted U-shape relationship with mobile app survival. We test our conjecture in this section.

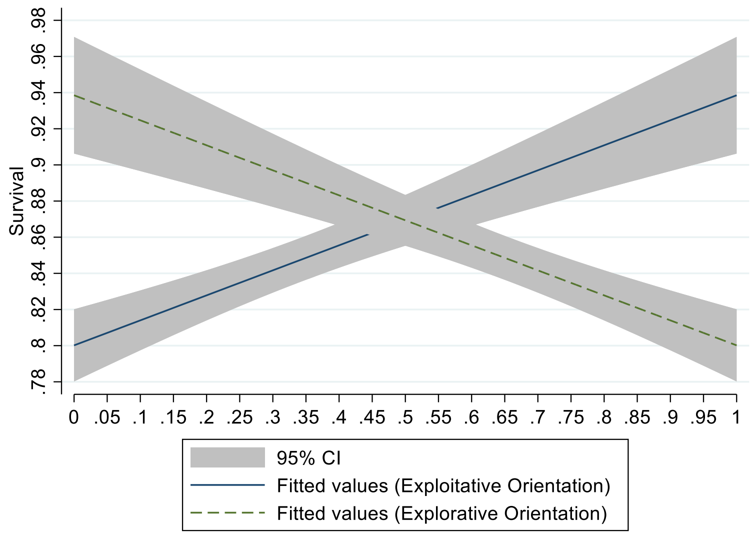

Figure D-1 Relationships Between Exploitation and Exploration Orientations and Survival

We estimate the same model specification as equation (1) but replaced *Exploitation-Exploration Orientation* (i.e., *Exploitation Orientation*) with this new *Exploitation-Exploration Orientation* (i.e., *Exploration Orientation*). The dependent variable is app survival at 1 month and 3 years. Table D-1 reports the results. In both model specifications, while the coefficients of *Exploitation-Exploration Orientation* are positive and statistically significant, the coefficients of its quadratic term, *Exploitation-Exploration Orientation^2* are negative and statistically significant. We found that *Exploitation-Exploration Orientation* is positively associated with app survival *up to a threshold* after which this positive relationship declines. We implemented the same method used in the main analysis and computed the inflection point. The inflection point is 0.493 for short-term survival; additionally, this U-shaped pattern is illustrated in Figure D-1. Thus, Hypothesis 1 is further confirmed.

Table D-1. Regression of Survival on Exploitation-Exploration Orientation (Corresponding to Table 3)

Variables	Survival (1 month)	Survival (3 years)
Exploitation-Exploration Orientation	0.2394**	0.2064**
	(0.1055)	(0.0879)
Exploitation-Exploration Orientation × *Exploitation-Exploration Orientation*	-0.2427***	-0.2116***
	(0.0797)	(0.0664)
Update Exploration	-0.0095	-0.0082
	(0.0084)	(0.0070)
# of Reviews(k)	-0.0117**	-0.0100**
	(0.0054)	(0.0045)
Avg Rating	-0.0821***	-0.0912***
	(0.0251)	(0.0209)
Avg Sentiment	0.1257*	0.1578***
	(0.0706)	(0.0588)
Review Length	-0.0026***	-0.0024***
	(0.0003)	(0.0002)
Review Velocity	0.0124	0.0107
	(0.0119)	(0.0099)

Update Length	-0.0098	-0.0089*
	(0.0062)	(0.0052)
Business	-0.2777***	-0.2347***
	(0.0200)	(0.0167)
Entertainment	0.0060	0.0030
	(0.0115)	(0.0096)
User Base(<1 million)	0.0330*	0.0433***
	(0.0195)	(0.0163)
User Base(1-10 million)	-0.0885***	-0.0645***
	(0.0131)	(0.0110)
In-app Purchase	-0.0317***	-0.0241***
	(0.0107)	(0.0089)
Constant	1.5084***	1.3639***
	(0.1365)	(0.1138)
Observations	1,952	1,952
Android Version Dummies	YES	YES

Standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1

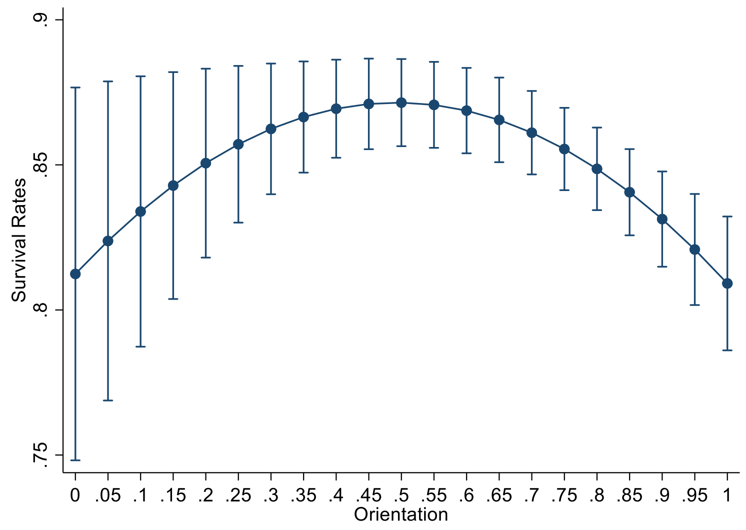

Figure D-1 (Exploration) Orientation and Survival (1 month)

D-2. The Effect of Review Volume and Concurrence on Exploitation-Exploration Orientation

We further ran two regression specifications to test the effects of review volume and concurrence on exploration orientation. In our main analysis, we found that review volume has an inverted U-shape relationship with exploitation orientation and that review concurrence has a positive relationship with exploitation orientation. Given the perfect negative correlation between exploration and exploitation orientation, if our findings are valid, we will expect a U-shape relationship between review volume and exploration orientation and a positive relationship between review concurrence and exploration orientation. We estimate the two models specified in equations (2) and (3) of the main analysis but replaced exploitation orientation with exploration orientation as the dependent variables.

Table D-2 reports the results. In both model specifications, the coefficients of *Review Volume* variable are negative and statistically significant at the 1% level and the coefficient of its quadratic term is statistically significant with a positive value in Specification 2. These results imply a curvilinear (U-shape) relationship between *Review Volume* and *Exploration Orientation*. Further, *Review Concurrence* coefficients are negative and statistically significant in both specifications. These results exactly match our expectations and further support our findings for H2 and H3.

Table D-2. Regression of Exploration Orientation on Review Volume and Concurrence

Variables	Specification 1	Specification 2
Review Volume	-0.0553***	-0.0585***
	(0.0096)	(0.0100)
Review Volume × Review Volume	0.0009***	0.0009***
	(0.0002)	(0.0002)
Review Concurrence	-0.0442***	-0.0553***
	(0.0166)	(0.0156)
Review Velocity		0.0215
		(0.0133)
Avg Rating		-0.0194*
		(0.0112)
Avg Sentiment		0.0474**
		(0.0200)
Review Length		-0.0809***
		(0.0120)
Release		0.0002
		(0.0010)

Update Length		0.0012
		(0.0121)
Constant	0.6801***	1.0653***
	(0.0297)	(0.1111)
Observations	1,952	1,952
R-squared	0.6232	0.6374
Year-quarter	YES	YES

Standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1

Appendix E: Moderating Effect of Sentiment on the Relationship between Review Volume/Concurrency and Exploitation-exploration Orientation

Numerous studies on user online reviews in the information systems and marketing fields have examined the effect of online reviews on consumer behaviors. One common finding is that online review sentiment can affect consumers' decisions (e.g., Duan et al., 2008, Yin et al., 2017). In the context of app development, developers (i.e., sellers) are also likely aware of the potential effects of review sentiment on app users (i.e., consumers) and thus consider review sentiment when updating apps. In this section, we explore the moderating role of review sentiment on our findings.

We initially explore the relationship between review volume and review sentiment. Previous scholars suggest that consumers often write online reviews to either express their negative or positive experiences with a product (Thakur, 2018). Our initial exploration in Figure E-1 confirms this in the context of app reviews, implying that review sentiment (either negative or positive) likely moderates the relationship between review volume and exploitation-exploration orientation. On the other hand, our initial exploration shows no significant correlation between review sentiment and review concurrence (0.03, *p*=0.17).

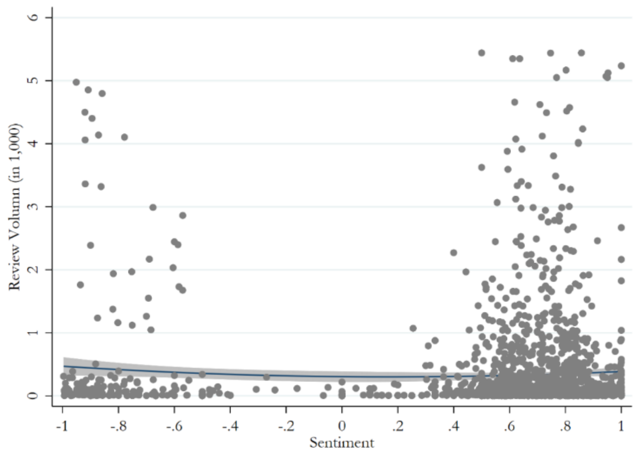

Figure E-1 Review Sentiment – Review Volume

In order to explore the moderating effect of review sentiment, we estimate the same model as our main model but add the interaction terms between review sentiment and review volumes and concurrence using all app release data. Appbot, from which we obtained our dataset, provides a sentiment score for each user review. The sentiment outcome is generated by using state-of-the-art machine learning approaches achieving 93% accuracy. It is also widely used by 500 Fortune companies such as Microsoft, LinkedIn, and Expedia (Garousi et al. 2022). *Avg Sentiment* captures the average sentiment score for an app release created before the new release and after the previous release.

Table E-1 reports the results. We first find that the coefficient of the interaction term between *Avg Sentiment* and *Review Volume* is negative and statistically significant at .01 level. This result shows that as review sentiment becomes more negative, review volume has a more positive effect on exploitation-exploration orientation. Figure E-2 further confirms our findings: for a given review volume, exploitation-exploration orientation increases when review sentiment becomes more negative. We also find that although the coefficient of the interaction term between *Avg Sentiment* and *Review Concurrence* is negative, it is insignificant. This result shows that when the review sentiment becomes more positive, review concurrence is likely to have insignificant but negative effects on exploitation-exploration orientation. This result is also shown in Figure E-3.

Table E-1. Regression of Exploitation-Exploration Orientation on Interaction Terms between Sentiment and Review Volume and Concurrence

Variables	Coefficients
Review Volume	0.1973***

	(0.0238)
Review Volume× Review Volume	-0.0087***
	(0.0012)
Avg Sentiment	-0.0304
	(0.0214)
Avg Sentiment × Review Volume	-0.2419***
	(0.0761)
Avg Sentiment × Review Volume× Review Volume	0.0133***
	(0.0035)
Review Concurrence	0.0323**
	(0.0160)
Avg Sentiment × Review Concurrence	-0.0099
	(0.0322)
Review Velocity	-0.0308**
	(0.0119)
Avg Rating	0.0224**
	(0.0102)
Review Length	0.0791***
	(0.0116)
Release	0.0004
	(0.0010)
Update Length	0.0003
	(0.0121)
Constant	-0.0794
	(0.1074)
Observations	1,952
R-squared	0.6690
Year-quarter	YES

Robust standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1

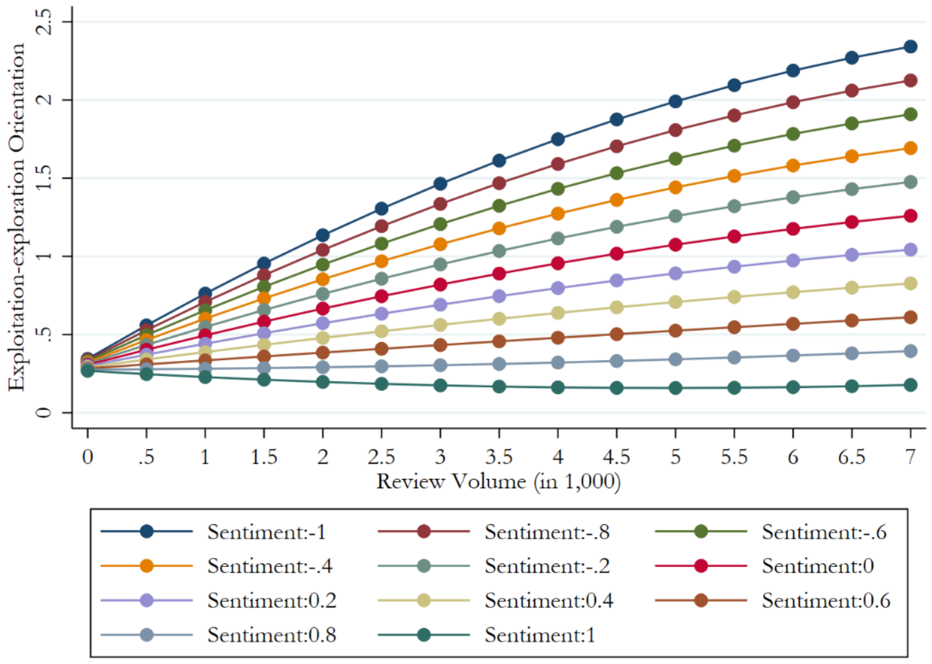

Figure E-2. Effects of Sentiment Interacts with Review Volume on Exploration Orientation

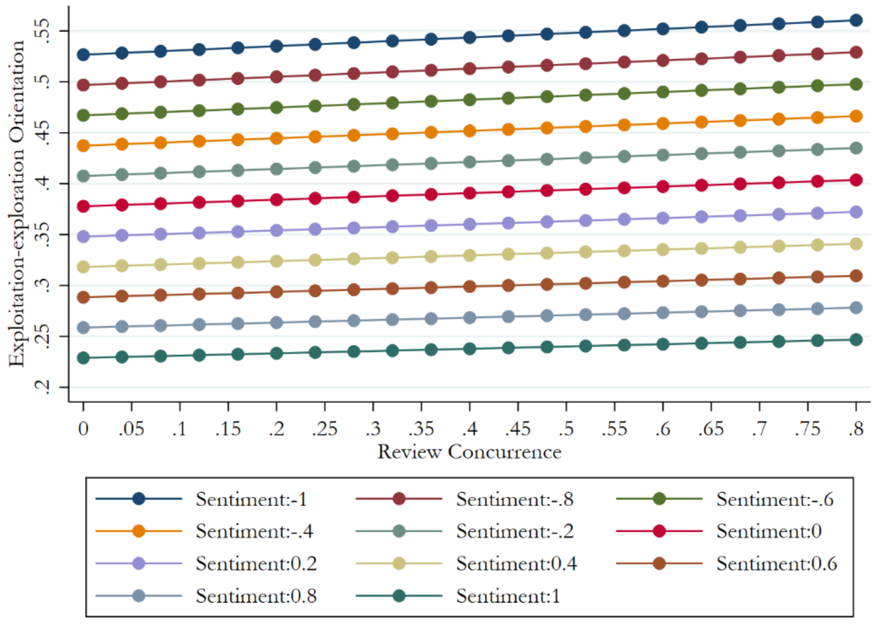

Figure E-3. Effects of Sentiment Interacts with Concurrence on Exploration Orientation

References

Chen, N., Lin, J., Hoi, S. C., Xiao, X., & Zhang, B. (2014, May). AR-miner: mining informative reviews for developers from mobile app marketplace. In *Proceedings of the 36th international conference on software engineering*, 767-778.

Dąbrowski, J., Letier, E., Perini, A., & Susi, A. (2022). Analysing app reviews for software engineering: a systematic literature review. *Empirical Software Engineering, 27*(2), 1-63.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Di Sorbo, A., Panichella, S., Alexandru, C. V., Visaggio, C. A., & Canfora, G. (2017, May). SURF: summarizer of user reviews feedback. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C),* 55-58.

Duan, W., Gu, B., & Whinston, A. B. (2008). Do online reviews matter?—An empirical investigation of panel data. *Decision Support Systems, 45*(4), 1007-1016.

Garousi, V., Cutting, D., & Felderer, M. (2022). Mining user reviews of COVID contact-tracing apps: An exploratory analysis of nine European apps. *Journal of Systems and Software, 184*, 111136.

Groen, E. C., Seyff, N., Ali, R., Dalpiaz, F., Doerr, J., Guzman, E., ... & Stade, M. (2017). The crowd in requirements engineering: The landscape and challenges. *IEEE Software*, *34*(2), 44-52.

Gu, X., & Kim, S. (2015, November). " What parts of your apps are loved by users?"(T). In *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 760-770.

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

Iacob, C., & Harrison, R. (2013, May). Retrieving and analyzing mobile apps feature requests from online reviews. In *2013 10th working conference on mining software repositories (MSR),* 41-44.

Johann, T., Stanik, C., & Maalej, W. (2017, September). Safe: A simple approach for feature extraction from app descriptions and app reviews. In *2017 IEEE 25th international requirements engineering conference (RE)*, 21-30.

Koroteev, M. V. (2021). BERT: A review of applications in natural language processing and understanding. *arXiv preprint arXiv:2103.11943*.

Liu, Y. (2006). Word of mouth for movies: Its dynamics and impact on box office revenue. *Journal of Marketing, 70*(3), 74-89.

Maalej, W., & Nabil, H. (2015, August). Bug report, feature request, or simply praise? on automatically classifying app reviews. In *2015 IEEE 23rd international requirements engineering conference (RE),* 116-125.

Mousavi, R., Raghu, T. S., & Frey, K. (2020). Harnessing artificial intelligence to improve the quality of answers in online question-answering health forums. *Journal of Management Information Systems*, *37*(4), 1073-1098.

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

Panichella, S., Di Sorbo, A., Guzman, E., Visaggio, C. A., Canfora, G., & Gall, H. C. (2016, November). Ardoc: App reviews development oriented classifier. In *Proceedings of the 2016 24th ACM SIGSOFT international symposium on foundations of software engineering*, 1023-1027.

Reimers, N., & Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Shin, D., He, S., Lee, G. M., Whinston, A. B., Cetintas, S., & Lee, K. C. (2020). Enhancing social media analysis with visual data analytics: A deep learning approach. *MIS Quarterly*, *44*(4), 1459-1492.

Thakur, R. (2018). Customer engagement and online reviews. *Journal of Retailing and Consumer Services*, *41*, 48-59.

Wei, X., Zhang, Z., Zhang, M., Chen, W., & Zeng, D. D. (2020). Combining Crowd and Machine Intelligence to Detect False News on Social Media. *MIS Quarterly*, *46*(2), 977-1008.

Yin, D., Bond, S. D., & Zhang, H. (2017). Keep your cool or let it out: Nonlinear effects of expressed arousal on perceptions of consumer reviews. *Journal of Marketing Research, 54*(3), 447-463.