The Journal of Systems and Software 125 (2017) 207–219

Contents lists available at ScienceDirect

The Journal of Systems and Software

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

A systematic literature review: Opinion mining studies from mobile app store user reviews

Necmiye Genc-Nayebi*, Alain Abran

Department of Software Engineering and Information Technologies, École de technologie supérieure, University of Québec, 1100 Rue Notre-Dame O, Montreal, QC H3C 1K3, Montréal, Canada

ARTICLE INFO

Article history:
Received 22 April 2016
Revised 13 November 2016
Accepted 14 November 2016
Available online 17 November 2016

Keywords:
Mobile application
App stores opinion mining
Systematic literature review
Requirements engineering

ABSTRACT

As mobile devices have overtaken fixed Internet access, mobile applications and distribution platforms have gained in importance. App stores enable users to search for, purchase and install mobile applications and then give feedback in the form of reviews and ratings. A review might contain information about the user's experience with the app and opinion of it, feature requests and bug reports. Hence, reviews are valuable not only to users who would like to find out what others think about an app, but also to developers and software companies interested in customer feedback.

The rapid increase in the number of applications and total app store revenue has accelerated app store data mining and opinion aggregation studies. While development companies and app store regulators have pursued upfront opinion mining studies for business intelligence and marketing purposes, research interest into app ecosystem and user reviews is relatively new. In addition to studies examining online product reviews, there are now some academic studies focused on mobile app stores and user reviews.

The objectives of this systematic literature review are to identify proposed solutions for mining online opinions in app store user reviews, challenges and unsolved problems in the domain, any new contributions to software requirements evolution and future research direction.

1. Introduction

With the rapid development of web and mobile devices, customers can now buy goods and services directly from online websites and digital distribution platforms. Users often rely on others' reviews or recommendations either from online purchase web sites or review sites to finalize their purchasing decisions. However, reading all reviews is time consuming and, sometimes, deceptive for users because of misleading or spam reviews. Therefore, researchers are looking into developing automated systems to identify, classify and summarize the opinions or sentiments and also to detect spam in an online text. Various researchers have studied opinion mining since the late 90s; however, the introduction of Machine Learning techniques and annotated datasets such as customer review datasets (Hu and Liu, 2004; Ding et al., 2008), pros and cons datasets Ganapathibhotla and Liu (2008), Amazon product review data (Jindal and Liu, 2008) and blog author gender classification dataset (Mukherjee and Liu, 2010) accelerated the research in the domain. With the emergence of different opinion mining domains such as social media (Facebook, Twitter, Instagram, App Store), app ecosystems, micro blogs, etc.), the focus of studies has since shifted into short-length texts, spam detection and contradiction analysis.

There also exists quite a number of survey studies on opinion mining and sentiment analysis in the literature. Pang and Lee (2008) made a comprehensive contribution into opinion mining and sentiment analysis survey studies by covering applications, major tasks of opinion mining, extraction and summarization, sentiment classification and also the common challenges in the research field. Tsytsarau and Palpanas (2012) surveyed the development of sentiment analysis and opinion mining research studies including spam detection and contradiction analysis. Their survey study provided 26 additional papers compared to Pang and Lee's (2008) preliminary survey. The survey of Tang et al. (2009) has a narrower scope, examining the opinion mining problem only for customer reviews on the web sites that couple reviews with e-commerce like Amazon.com or the sites that specialize in collecting user reviews in a variety of areas like Rottentomates.com. Cambria et al. (2013) revealed the complexities involved in opinion

* Corresponding author.
E-mail addresses: necmiye.genc.1@ens.etsmtl.ca (N. Genc-Nayebi), alain.abran@etsmtl.ca (A. Abran).
URL: https://www.etsmtl.ca (A. Abran)

http://dx.doi.org/10.1016/j.jss.2016.11.027
0164-1212/

mining with respect to current demand along with future research directions.

Along with internet and world wide web, mobile devices have gained popularity because of their portability, accessibility, and location awareness. Concurrently, the ever increasing demand for various kinds of mobile apps running on different devices has led to a corresponding increase in mobile developers and competitive mobile app markets. App ecosystem opinion mining studies did not start until the early 2010s, soon after the launch of the Apple app store, the first application distribution platform, in July 2008. The success of the Apple app store has led to the launch of other similar stores and services, with an exponential growth both in number of applications and revenue. The Apple app store generated over 10 billion dollar in revenue for developers in 2014 and currently offers about three million apps (Statista, 2014). Data mining and opinion aggregation from these platforms has therefore become a serious research topic.

User ratings and reviews are user-driven feedback that may help improve software quality and address missing application features. However, it is difficult for an individual to read all the reviews and reach an informed decision due to the ever growing amount of textual review data. Hence, over the last several years, various techniques and automated systems have been proposed to mine, analyze and extract user opinion and sentiment from app store review text. Our first research question aims to reveal the data mining techniques used for reviews on software distribution platforms.

One challenge in app store opinion data mining is vocabulary, which can vary, with the same term having different meanings in different contexts and domains. For example, even though “unpredictable’ may have a positive meaning for a movie or book review, it could indicate a negative opinion in a mobile app review and be associated with a possible bug or a quality issue. Since the linguistic context of terms used in reviews plays a key role in opinion mining, domain adaptation and transfer learning aspects should also be considered. Secondly, the reviews found in app ecosystems are relatively short (71 characters on average) and have different vocabulary compared to other commodity marketplaces (Fu et al., 2013). Harman et al. (2012) have pointed out that app ecosystems are a new form of software repository and very different from traditional repositories. The granularity in an app store ecosystem is finer and the information collected (such as price, customer rating, number of downloads and application features, in addition to user reviews) allows empirical analysis. Our second research questions looks for research studies that explore this domain dependency challenge.

Unbiased or non-spam user reviews may be numerous but of varying quality. The terms such as ‘Liked’, “Not recommend’, “OK app” do not convey any information about why users like an application or which aspects they like the most. Secondly, most reviews are poorly written and the information they contain often not useful, or highly personal and device- or technology-specific. Sophisticated ranking schemes, as found in the Apple app store and Google Play, measure reviews by their “helpfulness” as rated by users. In the Apple app store, the button under each user review allows other users to vote on whether the review is helpful or not; reviews may also be sorted from Most Helpful to Less Helpful based on these voting results. However, for newly written reviews or less popular applications, there would not be enough “helpfulness” voting to be of any use. Our third research question searches for studies that automatically asses and rank reviews in accordance with their usefulness or helpfulness.

As consumers increasingly rely on user reviews and ratings, there has been a stronger incentive to create fraudulent reviews in order to boost sales and damage competitors” reputations. Fraudulent reviews not only mislead customers into poor purchase decisions, but also degrade user trust in online reviews. Various studies and techniques have been proposed for detecting spam reviews. In an app ecosystem, spam app developers and opinion spammers (including those who would like to gain monetary profit or leak valuable user data such as contact lists or credit card information) tend to post spam reviews using Internet bots and puppet user accounts (Chandy and Gu, 2012). Despite some existing studies on opinion spam, the identification of spam in app stores has become another promising topic for researchers. Our fourth research question investigates spam identification and ranking fraud detection methods and techniques.

Users prefer having comparisons of specific features of different products available rather than having to gather isolated opinions about a single product themselves. In addition to average rating on a five-star scale and corresponding ranking on the app store, users prefer learning about others’ experience with the app, including which aspects/features they liked or disliked most. Each user has his/her own preferences and while one user might feel strongly about the appearance, others may focus on functional or technical aspects. Hence, there is a need to extract and rate individual application features. However, to be able to make such comparisons, domain knowledge (ability to spot features) and commonsense knowledge about how to identify text polarity are required. Our last research question searches for aspect-based opinion mining studies extracting application features from mobile app store reviews.

Even though some surveys have reviewed the techniques and methods in opinion mining and sentiment analysis from text, no SLR has reviewed the literature regarding mobile app store data mining, opinion aggregation and spam detection. Martin et al. (2016) provided an initial survey into literature that covers the period of 2000 to November 27,2015, however their survey is not a SLR and they particularly interested in studies that combine technical (Application Program Interface (API) usage, size, platform version and etc) and non-technical attributes (category, rating, reviews, installs and etc) of mobile apps. The goal of our SLR is to methodically review and gather research results for specific research questions and to develop evidence-based guidelines for app store practitioners. We developed a set of five research questions to guide the literature review process and performed an extensive search to find publications that answer the research questions.

The rest of this paper is structured as follows. Section 2 presents our research methodology, including the research questions. Section 3 presents the results obtained by our SLR and identifies the challenges and avenues in this new field. Section 4 presents discussions about the mobile app store opinion mining studies.

2. Research methodology

The SLR was conducted following the guidelines of Kitchenham (2004). The activities performed in the course of the SLR were structured into three phases: (1) planning, (2) conducting the review, and (3) reporting. See Fig. 1. The individual tasks performed in each activity are described in Sections 2.1–2.3 .

2.1. Planning

The planning phase clarified the specific objectives of the SLR, that is, to identify mobile app store studies, the challenges faced when mining app store data, how these challenges have been overcome, and any unsolved challenges. In addition, we specified the following five research questions and the motivations behind the questions.

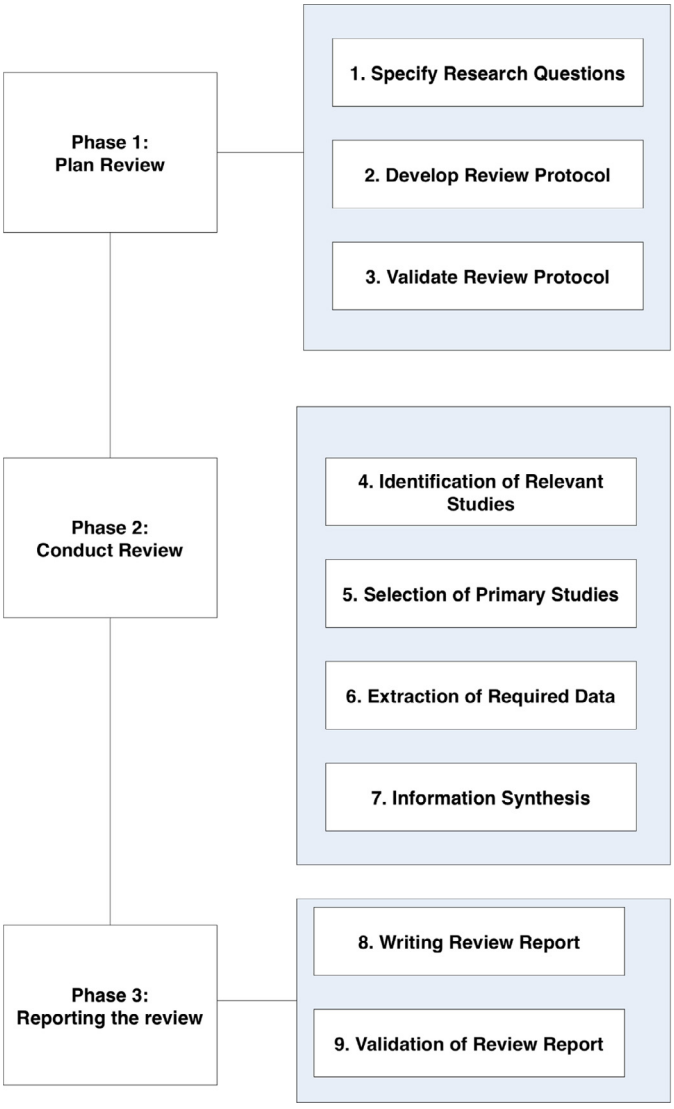

Fig. 1. SLR process.

2.1.1. Research questions

RQ1: Which specific data mining techniques are used for reviews on software distribution platforms?

Motivation: App stores provide a wealth of information in the form of customer reviews. Opinion mining and sentiment analysis systems have been applied to various kinds of texts including newspaper headline, novels, emails, blogs, tweets and customer reviews. Different techniques and automated systems have been proposed by researchers to extract user opinions and sentiments within the text over the years. Unlike documents or long length text, mobile app store reviews have some unique characteristics such as being short, informal and sometimes even ungrammatical consisting of incomplete sentences, elongations and abbreviations that make them difficult to handle. This question targets to present approaches and techniques proposed particularly for app store user review mining and opinion extraction problems.

RQ2: How do the studies remedy the 'domain dependency' challenge for app store reviews?

Motivation: Vocabulary varies within different context and domains, and same term might mean different opinions. An opinion classifier trained using opinionated words from one domain might perform poorly when it is applied to another domain. The reason is that not only the words and the phrases, but also the fact that language structure could differ from one domain to another. Hence, language structure and linguistic context of opinion and sentiments terms plays a key role in opinion mining, domain adaptation methods are also required to be considered while dealing with app store user reviews. This research question aims to reveal how mobile app store opinion mining studies tackle domain adaptation problem.

RQ3: What criteria make a review useful?

Motivation: Quality varies from review to review and low quality reviews might not convey any necessary signals to be used for information extraction. To tackle spam identification problem, it is critical to have a mechanism or a criterion that assesses the quality of reviews and filter out low-quality/noisy reviews. While review helpfulness is assessed manually by users in mobile app stores, there also exists some automated systems that assess and rank reviews in accordance with their usefulness or helpfulness. This research question aims to expose the methods or criteria used to differentiate useful app store reviews from the others. Besides, this research question also searches for automated systems that evaluate review usefulness and helpfulness.

RQ4: How can spam reviews be differentiated from legitimate reviews?

Motivation: As number of online reviews increased and fraudsters who produce deceptive or untruthful reviews emerged, tit is an essential task to identify and filter out the opinion spam. Different studies and techniques have been proposed for spam review detection problem. The opinion spam identification task has great impacts on industrial and academia communities. Our objective with this research question is to investigates spam review and ranking fraud detection methods and techniques for online stores and mobile app stores.

RQ5: Does the study extract targeted/desired software features from application reviews?

Motivation: Apart from app's average rating over 5-star scale and its corresponding ranking on app store, users would like to learn about others' experience with the app and which aspects/features they liked or disliked most. The information obtained from mobile app reviews is also valuable for developers to get user feedback about most liked or expected features (Requirements Elicitation) and bugs on the application (Software Quality and Software Evaluation). This research question focuses on aspect-based opinion mining studies extracting application features and aims to reveal the studies that make automated application feature extraction and rating in the face of user reviews.

2.1.2. Development and validation of the review protocol

The review protocol defines the activities required to carry out the literature review. A review protocol helps reduce researcher bias and defines the source selection and searching processes, quality criteria and information synthesis strategies. This subsection presents the details of our review protocol.

The following digital libraries were used to search for primary studies:

- Science Direct
- IEEExplore
- ACM Digital Library
- Citeseer library (citeseer.ist.psu.edu)
- Springer Link
- Google Scholar

The following search query was created by augmenting the keywords with possible synonyms. While conducting the review, we examined the reference list of primary studies to determine if there were additional studies not captured by our research query.

((mobile OR software) OR ((apps OR app OR application) OR (market OR ecosystem OR AppStore OR store))) AND ((data OR (on-

Table 1
Inclusion and exclusion criteria.

Inclusion criteria	Exclusion criteria
Case studies and surveys of text analysis, opinion mining and sentiment analysis from app store reviews.	Papers that present opinions without sufficient and reliable supporting evidence.
Preliminary analysis of mobile app store reviews, vocabulary, trends.	Studies not related to the research questions.
Papers searching application feature requests and bug reports within review text.	Papers that do not comply with the evaluation criteria in Table 2.
Papers that describe the criteria of what makes a review useful and helpful for readers.	Preliminary conference papers of journal papers by same author(s).
Papers that distinguish fake reviews and spams from legitimate ones.	

Table 2
Quality checklist Keele (2007).

No	Question
1	Are the aims of the study stated clearly?
2	Is the basis of evaluative appraisal clear?
3	How defensible is the research design?
4	Are data collection methods described adequately?
5	Has the approach to, and formulation of, analysis been conveyed adequately?
6	Has the diversity of perspectives and contexts been explored
7	Are there any links between data, interpretation and conclusions?
8	Is the reporting clear and coherent?
9	Has the research process been documented adequately?
10	Could the study be replicated?

line OR review) OR user OR (text OR comment OR vocabulary)) OR rating OR (opinion OR sentiment) OR (mining OR analysis OR processing) OR (feature OR requirement) OR request OR expectation OR (bug OR quality OR complain OR issue) OR (usefulness OR helpfulness))

Study Selection Procedure: We systematically selected the primary studies by applying the following four steps:

1. We examined the paper titles to eliminate studies unrelated to our research focus.
2. We reviewed the abstracts and keywords in the remaining studies. If either the abstracts or keywords did not provide the necessary information, we reviewed the results and conclusion sections to determine if the study was relevant.
3. We filtered the remaining studies in accordance with the inclusion and exclusion criteria given in Table 1.
4. We double-checked the reference list of the initial primary studies to identify additional studies that might be relevant to our search.

We evaluated the quality of the primary studies using the checklist adapted from Keele (2007). Each study was evaluated according to the quality checklist questions given in Table 2. The studies that provided a 'yes' answer to at least seven questions from the checklist were selected.

2.2. Conducting the review

2.2.1. Identification and selection of relevant studies

We followed Wohlin's (2014) snowballing procedure in order to identify relevant studies. In the first step called database search, we identified the keywords and formulated search string as given in Session 2.1.2. Our research with the search query generated more than 500 hits that will build up our start set. After examining the paper title, abstract, keywords, results and conclusions (if necessary) to filter out unrelated studies, 63 studies remained as start set. We used the reference list of our start set papers to identify new papers to include. Afterwards, we went through the reference list and exclude the papers that do not fulfil the basic criteria such as title, language and publication venue. We also performed forward snowballing to identify new papers based on those papers citing the paper being examined. Each candidate citing the paper is examined by screening the information provided in Google Scholar. If this information is not sufficient enough for a decision, the citing paper is examined in more details. After implementing backward snowballing and forward snowballing steps, we ended up with 45 research papers. Using the inclusion and exclusion criteria and quality checklist, examination of the remaining literature produced 24 primary studies.

2.2.2. Extraction of data

We used the data extraction form in Table 3 to extract data from the 24 primary studies. Even though the same data items were searched with RQ1 and RQ4, opinion mining and spam analysis studies respectively, the results obtained are presented in distinct tables. See Tables 4 and 5 in the Appendix.

2.2.3. Information synthesis

We read the 24 selected studies noting the methods and findings that were repeated. Inconsistencies and contradictions in the information were also recorded and are presented in the discussion and principal findings sections.

2.3. Reporting the review

Data extracted from the primary studies were used to answer our five research questions. The guidelines of Kitchenham (2004) were closely followed in the reporting of results.

3. Results

3.1. RQ1: Which specific data mining techniques are used for the reviews on software distribution platforms?

Analysis of the 24 primary studies identified a number of specific opinion mining and opinion extraction techniques used for reviews on software distribution platforms.

Chen and Liu (2011) identified useful app features ((i) static (e.g., application name, provider), (ii) dynamic (e.g., current rate, update date) and (iii) comment (e.g., user rate, comment content)) to predict app popularity and trained a model for an automated popularity prediction task. To create the dataset, they sampled 102,337 applications and a list of dynamic features were accumulated for top 200 paid and free applications by tracking their daily ranking. They used a Classification and Regression Tree (CART) model as a popularity prediction model and leveraged static (app name, provider, category, etc.), dynamic (current rank, all version count, all version rate, etc.) app and app store features and also comment features (user rate, comment title and comment content). As a result, they found that the top-ranked (the search ranking on the app store that factors in average app store rating, rating/review volume, download and install counts and app usage

Table 3
Data extraction form.

Search focus	Data item	Description
General	Identifier	Reference number given to the article
	Bibliography	Author, year, title, source
	Type of article	Journal/conference/technical report/etc.
	Study aims	Aims or goals of the study
RQ1 / RQ4	Text and data mining methods and techniques used	Algorithms, models and measures
	Selected or obtained review features	The subset of text features used or identified in the study
	Dataset	List of chosen applications, number of reviews
	Performance/Results	Precision, recall, accuracy/Obtained results
RQ2	Domain-specific text and data mining techniques used for app store reviews	App store and app review specific algorithms, methods
	Specific features used for app store reviews	App store and app review specific text features
	Performance improvement	Performance improvement compared to conventional opinion mining studies
RQ3	User review helpfulness/usefulness assessment framework	Predictors, variables, features that specify review quality
	Model used for automated usefulness task	Algorithms, models and measures
	Selected features	Subset of text features used in the study
	Performance	Precision, recall, accuracy
RQ5	Extracted app features	Mobile app features retrieved from online review text
	Method	Approaches, techniques used for automatically extracting application features
	Performance	Precision, recall, accuracy

Table 4
List of mobile app store data mining studies.

No	Reference	Study Name	Method	Features	Dataset	Performance Results
1	Chen and Liu (2011)	Predicting Popularity of Online Distributed Applications	CART	Static features, Dynamic features, Comment features	200 paid applications sampled from 102,237 applications	Preliminary observations were presented as results
2	Vasa et al. (2012), Hoon et al. (2012)	A Preliminary Analysis of Mobile App User Reviews, A preliminary analysis of vocabulary in mobile app user reviews	Summary statistics, Box plots, Distribution charts	Word frequencies	8.7 million reviews from 17,330 apps	Top 20 most frequent words were presented as results
3	Harman et al. (2012)	App Store Mining and Analysis: MSR for App Stores	Correlation analysis and greedy algorithm for extraction and grouping of features	Price, Rank of downloads, Rating mean	32,108 non-zero priced apps	Correlation between customer rating and the rank of app downloads is presented
4	Iacob et al. (2013)	What Are You Complaining About: A Study of Online Reviews of Mobile Applications	Manual analysis	Positive, negative, comparative, price related, missing requirements, issue reporting, usability, customer supports and versioning	Randomly selected 161 apps and 3279 reviews from Google Play Store	Distribution of code classes is given in results
5	Ha and Wagner (2013)	Do Android Users Write About Electric Sheep?	Manual classification	PAdjective (positive/negative), ads (positive/negative), aesthetics, company, comparison, feature/functionality, model, money, permissions, preinstalled, recommendations, resources, tips, uninstalled, used to be, work/doesn't work	556 reviews from 59 applications	Results include percentage of how often broad topics appeared in reviews
6	Wano and Iio (2014)	Relationship between Reviews at App Store and the Categories for Software	Manual text analysis	N/A	500 applications from various categories	Review styles are different with software categories
7	Gómez et al. (2015)	A Recommender System of Buggy App Checkers	LDA (for topic mining) and J48 for learning patterns	N/A	User reviews from 46,644 reviews	N/A
8	Mojica Ruiz et al. (2015)	An Examination of the Current Rating System used in Mobile App Stores	Hexbin plot	N/A	242,089 app versions of 131,649	Store rating is very resilient to changes in the version rating

Table 5
List of mobile app store spam identification studies.

No	Reference	Study Name	Method	Features	Dataset	Performance/Results
1	Chandy and Gu (2012)	Identifying Spam in the iOS App Store	CART	Decision tree model and latent class graphical model	User average rating, user number of reviews, application average rating, app number of reviews, number of instances with 2, 3, 4 stars, developer number of applications, developer average rating, binary class indicators	6.4% classification error with false positive rate 6.3% and false negative rate 40.9.

statistics) paid applications were not closely related to customer ratings.

Vasa et al. (2012) and Hoon et al. (2012) made a preliminary analysis of mobile app user reviews. They initially analyzed the data using summary statistics with a one-way ANOVA test, box plots and cumulative distribution charts to confirm their hypothesis that rating and category have an affect on the length of the review. They analyzed 8.7 million reviews from 17,330 app and according to their analysis, users take the time to express their discontent by writing longer reviews, in contrast to short reviews when content with the application. They also identified a strong correlation between positive-negative sentiments and one- and five- star ratings. Unexpectedly, more than 50% of the two and three-star rated user reviews did not include any sentiment.

Harman et al. (2012) mined the Blackberry app store using Spearman's Rank Correlation method and identified a strong correlation between application rating and number of downloads, whereas there is no correlation between price and rating, nor price and number of downloads. They tested their approach to the 32,108 non-zero priced apps. Iacob and Harrison (2013) manually analyzed reviews and identified nine classes of feedback: positive, negative, comparative, price related, request for requirements, issue reporting, usability, customer support and versioning. They first randomly choose 169 apps and collected 3279 user reviews and then manually examined and classified reviews based on their content and then coded the categories, for example: aesthetics, company, comparison, feature/functionality, model, permissions, money, etc. They observed a correlation between review positivity and feature or functionality request.

Ha and Wagner (2013) manually analyzed Android users' reviews to see what they write about when reviewing Google Play applications. They crawled Google Play to collect information about 202,264 free applications and they selected 60 free applications with 556 reviews. As a result, they found that small subset of reviews had pointed privacy and security implications, whereas the majority of the reviews focused on the quality of the applications. Wano and Iio (2014) performed a manual text analysis and determined that review styles differ with software categories. The study used the search API and also used RSS Feed Generator by Apple. The number of targeted software is 500 and for each software, the targeted reviews are restricted up to 50 because of the API restriction. They concluded that consumers should pay attention to bias in reviews.

Gómez et al. (2015) mined reviews with LDA and error-suspicious permission patterns with the J48 decision tree algorithm (a Weka implementation of the C4.5 algorithm), revealing potential correlations between error-sensitive permissions and error-related reviews over time. They built a dataset that consists of a random sample of all the mobile apps available on Google Play Store. They collected 500 applications from 27 different categories.

Mojica Ruiz et al. (2015) made an overall evaluation of app stores and user rating schema and concluded that the current store rating of apps was not dynamic enough to capture the changing user satisfaction levels along with evolving application versions. Their dataset was extracted by crawling Google Play and this resulted in 242,089 app versions of 131,649 mobile apps. After the filtration, they ended up with 238,198 versions of 128,195 apps. They used hexbin plots to examine whether there would be a noticeable change in the store-rating of an app given a rise or drop in the rating of a specific version of that app.

Most of studies identified within this research question are preliminary researches and based on either manual or statistical analysis of user reviews. The researchers used either the research API and RSS Feed Generator by Apple store or some scrapers script to collect app store data. The datasets are mostly created with random sampling of all the mobile apps available and there is not any specific or common app category preferred by researchers. Since Martin et al. (2015) presented empirical evidence that indicates that the partial nature of data available on App Stores could pose an important threat to the validity of findings, the obtained results from different App Store research studies could not be compared with one another. Star rating, category and review content are most common features collected within 87.5% of the studies.

We could not obtain any data regarding average length of a review considered in the studies, however the dataset by Vasa et al. (2012) showed that user review length is highly skewed with an average of 110 characters. On the other hand, Fu et al. (2013) stated in their paper that average length of the comments is 71 characters, and median length is 47 characters. If the datasets used in the research studies would be publicly available, we will have the chance to validate these numbers. For this reason, researchers need to augment their findings with an argument to convince the reader that any sampling bias is unlikely to affect their research findings and conclusions. One of very recent studies by Gu and Kim (2015) indicated this app store sampling phenomenon as a threat to validity.

Table 4 presents the list of 9 studies with their methods, details of datasets, features and performance as a response to RQ1.

3.2. RQ2: How do the studies remedy the 'domain dependency' challenge for app store reviews?

RQ2 looked at how domain dependency affects opinion mining from reviews. A classifier trained in using opinionated words from one domain might perform poorly when applied to another domain since not only words and phrases but also language structure may differ from one domain to another.

From the primary studies it was noted that some researchers labelled data for the new domain and created their own dataset from scratch, whereas other researchers used labelled data from one domain and unlabelled data from the target domain, and then made the domain adaptation by using general opinion words (Aue and Gamon, 2005; Yang et al., 2006; Blitzer et al., 2007; Pan et al., 2010). In order to overcome the domain barrier in opinion extraction, Cosma et al. (2014) proposed a generalized methodology by considering a set of grammar rules for the identification of opinion-bearing words.

In addition, online reviews have distinctive text features, including short length, unstructured phrases and abundant information. Short reviews bring new challenges to traditional research topics in text analytics, such as text classification, information extraction and sentiment analysis. As opposed to standard texts, which include many words and phrases and their corresponding statistics, short texts consist of few phrases and sentences. Several traditional text analytics methods have been proposed to tackle the data sparseness problem:

- The first is surface representation that uses phrases in the original text from different product aspects to maintain the contextual information. However, this method fails to produce a deep understanding of the text and the method does not make use of external knowledge, which has been found useful in dealing with the semantic gap in text representation (Hu and Liu, 2012). For example, this review from the app store: "This iOS 9 update. App crashing and ugly font", does not contain any words or phrases related to the reason for the crash and possible user interface (UI) design problem, while the words 'crash' and 'font' are related to software engineering concepts. Hence, it is difficult to use bag-of-words based models and methods to build semantic connections between the review text and software characteristics.
- Another approach is to enrich the context of basic text segments by searching the external sources. Such methods have been found effective in narrowing the semantic gap for different tasks (Gabrilovich and Markovitch, 2007; Alfonso Ureña-López et al., 2001). In the app store corpus, these external sources would be app crash reports, tweets, community blogs and code repositories.

Another important characteristic of online text, particularly in online reviews, is the use of colloquial language. When composing a review, users might use abbreviations or acronyms that seldom appear in conventional text. As an example, the phrases "superb" "Good 2go" "you do not buy the guarskldj; al b bbbbbbbbb„„„„wke;" make it very difficult to identify the semantic meaning. With research question RQ2, we sought to discover how researchers tackled domain adaptation problems, how they dealt with distinctive features of the review text and what specific methods or algorithms and text features were used to improve performance. To answer this research question, we reviewed the selected studies to identify the training datasets, methods, text features and performance comparisons. The mobile app store researchers mentioned in Table 6 used their own annotated dataset rather than leveraging existing online review datasets. Since they preferred to use conventional text mining methods such as Latent Dirichlet Allocation (LDA), Aspect and Sentiment Unification Model (ASUM), Naive Bayes classifier and statistical analysis, we cannot present any new method developed for the app store corpus. No new solutions or methods were proposed for examining the text characteristics (e.g., short length, unstructured phrases and colloquial language and challenges) of app store user reviews.

As in many real-word applications, topics revealed by Latent Dirichlet Allocation (LDA) and Aspect and Sentiment Unification Model (ASUM) are needed to be verified by experts to ensure they are semantically meaningful within the domain analysis. Hence, 4 studies out of 24 leveraged truth sets to understand if the extracted features align with real app features and to minimize the threat to validity. Galvis˜Carreño and Winbladh (2013) used the manually classified data as a truth set. Since the second author is not domain expert or not involved in software development, they reported that the process is error-prone. Chen et al. (2014) collected the group truth labels of the training pool and test set according to pre-defined rules. Guzman and Maalej (2014) and Gu and Kim (2015) also used the truth set that was created with systematic assessment of review samples by human coders.

However, manual validation could dominate the time and cost of building high-quality topic models. To overcome this problem, some researchers proposed measuring topic quality with topic coherence and statistical methods (Mimno et al., 2011; Newman et al., 2009). We propose incorporating domain knowledge into Topic Modelling via Dirichlet Forest Priors (Andrzejewski et al., 2009). Dirichlet Forest Priors, when combined with LDA, allows the user to encode domain knowledge (must-links and cannot-links between words) into the prior on topic-word multi nominal P(word| topic). In this way, app store domain knowledge could be expressed by a set of Must-Links (Two words u, v have similar probability within any topic) and Cannot-Links (Two words u, v should not both have large probability within any topic).

3.3. RQ3: What criteria make a review useful?

Review quality varies from reviewer to reviewer, and low-quality reviews might not convey any useful information. App store regulators allow users to vote on the helpfulness of each review and then rank the reviews based on votes. While review helpfulness is usually assessed manually, there are automated systems that do this. For the manual review of usefulness, there are no defined criteria among users. A review that appears helpful to one user may not be helpful for others, since they might be searching for different information or have differing priorities or biases. On the other hand, standard defined criteria would be valuable to differentiate useful reviews from others. Reviews chosen in accordance with these criteria for data mining and opinion extraction studies would yield the maximum capability for information extraction. Studies examining online review helpfulness are as follows:

- Cheung et al. (2008) measured review quality in terms of completeness, timeliness, accuracy and relevance.
- Mudambi and Schuff (2010) found that review depth had a positive effect on the helpfulness of the review but product type affected the perceived helpfulness of reviews.
- Pan and Zhang (2011) analyzed a large sample of reviews from Amazon to identify what determined information helpfulness and found that review length and positive reviews had a direct correlation with review usefulness.
- Korfiatis et al. (2012) discovered that review readability and positive ratings affected the number of helpfulness votes.

Studies on automatically assessing review helpfulness:

- Kim et al. (2006) trained an SVM (Support Vector Machine) regression model to learn the helpfulness function and then applied it to rank unlabelled reviews. They found that the most important features were the length of the review, its unigrams and its product rating.
- Liu et al. (2008) also modelled the helpfulness of reviews. They showed that helpfulness of a review depends on three important factors: reviewer expertise, writing style, and timeliness.
- Ghose and Ipeirotis (2011) used a Random Forest-based classifier and examined the relative importance of three feature categories: (i) reviewer related, (ii) reviewer subjectivity, and (iii) review readability. They found that using any of the three feature category results provided the same performance as using all available features.
- Moghaddam et al. (2012) used a probabilistic graphical model based on Matrix Factorization and Tensor Factorization. These models are based on the assumption that the observed review ratings depend on latent features of the reviews, reviewers, raters and products. They reported that the latent factor models outperform state-of-the-art approaches.

Table 6
List of mobile app feature extraction studies.

No	Reference	Study Name	Method	Extracted app features	Performance/Results
1	Iacob and Harrison (2013)	Retrieving and Analyzing Mobile Apps Feature Requests from Online	LDA	positive, negative, comparative, price related, missing requirements, issue reporting, usability, customer support and versioning	N/A
2	Galvis˜Carreño and Winbladh (2013)	Analysis of User Comments: An Approach for Software Requirements Evolution	ASUM	They presented sample topics identified per application. As an example for Facebook: 'Updates', 'Developer', 'Messages', 'Photos'	For K=24 Precision: 62.5 Recall: 20.83 F-Measure: 31.44 For K=48 Precision: 86.67 Recall: 54.16 F-Measure: 66.64 K=150 Precision: 90 Recall: 75 F-Measure: 80
3	Pagano and Maalej (2013)	User Feedback in the AppStore: An Empirical Study	Statistical Analysis	Community, requirements, rating, user experience	"Rating" is revealed as most frequent them with the frequency of over 77%. Requirements-30%, Community - 13%.
4	Fu et al. (2013)	Why People Hate Your App – Making Sense of User Feedback in a Mobile App Store	Statistical Analysis	Attractiveness, stability, accuracy, compatibility, connectivity, cost, telephony, picture, media and spam	91% (Precision), 73% (Recall)
5	Oh et al. (2013)	Facilitating developer-user interactions with mobile app review digests	SVM	Functional Bug, Functional Demand, Non-functional Request	0.8981 (Precision), 0.8165 (Recall), 0.8553 (F-Measure)
6	Chen et al. (2014)	AR-Miner: Mining Informative Reviews for Developers from Mobile App Marketplace	EMNB (Expectation Maximization for Naive Bayes)	They presented sample topics identified per application. As an example for Swiftkey: 'more theme', 'swype feature', 'space bar', 'more option', 'like keyboard' and etc.	F-measure: 0.764 - SwiftKey 0.877 - Facebook 0.797 - TempleRun2 0.761 -TopFish
7	Guzman and Maalej (2014)	How Do Users Like This Feature? A Fine Grained Sentiment Analysis of App Reviews	LDA	Functionality related topics were extracted	0.59 (Precision), 0.51 (Recall)
8	McIlroy et al. (2015a)	Analyzing and Automatically Labelling the Tyes of User Issues that are Raised in Mobile App Reviews	Naïve Bayes, Decision Tree, SVM	Additional cost, functional complaint, compatibility issue, crashing, feature removal, feature request, network problem, privacy and ethical Issue, resource heavy, response time, uninteresting content, update issue, user interface	Average 59% (Accuracy), 44 percent (Exact Match), 65 percent (Precision), 64% (F-measure micro) and 56% (F-measure macro))
9	Khalid (2013)	On Identifying User Complaints of iOS Apps	Manual Tagging	Hidden Cost, Functional Error, Compatibility, App Crashing, Feature Removal, Feature Request, Network Problem, Privacy and Ethical, Resource Heavy, Unresponsive App, Uninteresting Content, Interface Design	N/A
10	Khalid et al. (2015)	What Do Mobile App Users Complain About?	Manual Tagging	App Crashing, Compatibility, Feature Removal, Feature Request, Functional Error, Hidden Cost, Interface Design, Network Problem, Privacy and Ethics, resource Heavy, Uninteresting Content, Unresponsive App	N/A
11	Vu et al. (2015)	Mining User Opinions In Mobile App Reviews	Keyword extraction, grouping and ranking	Battery, versioning, unrecoverable error, snapchat, authentication, facebook	Average: 83.11% Accuracy
12	Park et al. (2015)	Leveraging User Reviews to Improve Accuracy for Mobile App Retrieval	AppLDA	Top Topics by LDA: log, upgrade, purchase, note, account, battle, refund, support	NDCG (Normalized Discounted Cumulative Gain) @3 =0.651, @5=0.656, @7=0.627, @20=0.634
13	Panichella et al. (2015)	How Can I Improve My App? Classifying User Reviews for Software Maintenance and Evolution	Bayes, SVM, Logictic Regression, J48 and ADTree	Information Giving, Information Seeking, Feature Request, Problem Discovery, Others	Precision = 0.79, Recall = 0.719, F-Measure = 0.672, best results obtained with the features Natural Language Processing (NLP), Text Analysis (TA) and Sentiment Analysis (SA)
14	Gu and Kim (2015)	What parts of your apps are loved by users?	SUR-Miner (POS tag, Parsing Tree and Semantic Dependence Graph (SDG)	Aspect Evaluation, Praises, Feature Request, Bug Reports and Others	F1-score = 0,81

For the mobile app store corpus, we could not find any study that assessed app store review helpfulness either manually or automatically. As app store users could mark any review for any app as: 'Helpful', 'Unhelpful' and 'Spam' and the reviews could be ranked per their helpfulness at Google Play, some of App Store mining researchers such as Chen et al. (2014) and Park et al. (2015) from Table 6 preferred using only Helpful reviews or filter Unhelpful reviews out to train their models. According to Pagano and Maalej (2013)'s dataset only 67,143 (5.96%) reviews are rated by other users regarding their usefulness. From these, 38,519 (57.37%) are considered 100% helpful. Interestingly, 16,671 (24.83%) are rated completed useless. Even though we could not get enough

information about the percentage of helpful or unhelpful reviews in other datasets, the need for filtering these reviews has been apparent to maximize the information extraction capability.

3.4. RQ4: How could the spam reviews be differentiated from legitimate reviews?

As consumers increasingly rely on user reviews and ratings, there is greater incentive to create fraudulent reviews in order to boost sales and to damage competitor reputations on the market. Fraudulent reviews not only mislead customers into poor purchase decisions, but also degrade user trust in online reviews. According to a Harvard Business School study (Luca and Zervas, 2013), 20% of all online reviews on Yelp.com are fake.

Most of the earlier research focused on detecting email and web spam. As the number of online reviews increases, as well as the number of fraudsters, different studies and techniques have been proposed to detect spam reviews. Two main approaches are being used for opinion spam detection: behavioural and textual features. Behavioural features correspond to features such as review date, rating, and geo-location of the reviewer, while textual features refer to methods, such as part-of-speech patterns, word frequency, n-grams and cosine similarity.

Dellarocas (2000), the first to work on immunizing online reputation systems against unfair ratings and discriminatory behaviour, proposed a set of 'exception handling' techniques such as 'controlled anonymity' and 'cluster filtering'. Kim et al. (2006) used SVM regression on different classes of features including structural (e.g., html tags, punctuation, review length), lexical (e.g., n-grams), syntactic (e.g., percentage of verbs and nouns), semantic and metadata (e.g., star rating) features.

Jindal and Liu (2008) observed that spammers tended to create a small number of review templates and then copy them to spam a single product or several different products. To identify the replicated spam reviews, they used two-gram review content comparison method, as in Kim et al. (2006).

Lim et al. (2010) trained a linear regression model to use four different spamming behaviour models as target products and groups, general rating deviation and early rating deviation. Wang et al. (2011) proposed a heterogeneous graph model to capture relations between reviewers, reviews and stores. Sandulescu and Ester (2015) presented two methods: (i) a semantic similarity measure by extracting specific parts-of-speech (POS) patterns and (ii) an LDA model using bag-of-words and opinion phrases.

Within the corpus of mobile app stores, scammers use a great many bogus user accounts or bots in order to download applications multiple times and write fraudulent reviews. In this way, the applications begin appearing on the top charts and have greater visibility in an app store search. In addition, there are numerous sites that allow purchasing of reviews. One example such a site is Fiverr. Even though fake and opinion spam reviews are widespread and have significant manipulative effects on app store success, regulators have only recently begun to crack down on fake reviews (Clover, 2014). We found only a single app store review spam identification study in the literature:

- Chandy and Gu (2012) compared latent class graphical and decision tree models for classification of app spam and analyzed the preliminary results for clustering reviews. They used linear Gaussian parameterization on the labelled data, which achieved higher accuracy than a baseline decision tree model. As a result, they proposed a latent class model for the spam identification task. The details of this study are presented in Table 5.

3.5. RQ5: extracted application features from user reviews

App stores provide a user feedback capability that is particularly useful and interesting from the software requirements engineering point of view. User ratings and reviews are user-driven feedback that may help improve software quality and address missing features.

With regard to extracted application features from app store user reviews, Iacob and Harrison (2013) identified nine different classes of feedback: positive, negative, comparative, price related, missing requirements, issue reporting, usability, customer supports and versioning. Galvis˜Carreño and Winbladh (2013) adopted the Aspect and Sentiment Unification model (ASUM), which incorporates both topic modelling and sentiment analysis to obtain constructive feedback from user comments. They extracted various topics such as updates, features and developers from review text.

Pagano and Maalej (2013) identified topics in user app store reviews by grouping the information as follows:

- Community: References to other reviews or other applications.
- Requirements: All request types such as feature, content, improvement requests, shortcomings and bug reports.
- Rating: User intention to change his/her idea given certain improvements.
- User experience: Helpfulness in terms of application features and user interface.

In addition, they pointed out the correlation between overall app ratings and number of user reviews, app price and amount and type of feedback the application received. Fu et al. (2013) identified 10 top factors that affect the success of an app on mobile application ecosystems: attractiveness, stability, accuracy, compatibility, connectivity, cost, telephony, picture, media and spam. In addition, they identified 0.9% inconsistencies between user review texts and rating that may be caused by careless mistakes or intention to mislead.

Oh et al. (2013) developed a review digest system (SVM classifier) which was tested on 1,711,556 reviews mined from 24,000 Google Play apps. They automatically categorized user reviews into functional and non-functional requests, bug reports and produced a digest featuring the most informative reviews in each category. Chen et al. (2014) compared the Latent Dirichlet Allocation (LDA) and Aspect and Sentiment Unification Model (ASUM) and found that LDA presented many "non-informative or redundant topics" However, they validated their results on user reviews of only four Android apps, and it is not clear that the framework will attain similar good results when applied to other Android apps or other app stores.

Guzman and Maalej (2014) used topic-modelling techniques to group fine-grained explicit features into high-level features using topic modelling LDA and weighted-average techniques. In addition, they compared the relevance of the extracted features with app requirements and concluded that for the top 10 popular extracted features, the words (e.g., upload photo, file exchange – for Dropbox, board pin, time search – for Pinterest) usually described actual app features and conveyed some clues about how the app was used. McIlroy et al. (2015a) and its counterpart studies Khalid (2013) and Khalid et al. (2015) automatically labelled the types of user issues raised in mobile app reviews, such as additional cost, functional complaint, compatibility issue, crashing, feature removal request, network problem, privacy and ethical issue, resource heaviness, response time, uninteresting content, update issue and user interface. They manually labelled a statistically representative sample of user reviews from the Apple app store and Google Play.

Vu et al. (2015) pursued a keyword based approach to collect and mine user opinion from app stores by extracting, ranking and grouping keywords based on semantic similarity. In addition, they

provided a visualization tool that showed the occurrence of keywords over time and reported any unusual patterns. Park et al. (2015) developed a topic model AppLDA that is designed for use on app descriptions and user reviews. Their proposed method enables developers to inspect the reviews and find out important app features of apps.Panichella et al. (2015) presented a system for automatically classifying user reviews based on a predetermined taxonomy, in order to support software maintenance and requirement evolution. Gu and Kim (2015) proposed a SUR-Miner that is a review summarization and categorization tool, which evaluated 2000 sentences from the reviews of 17 Google Play apps. In addition to these studies, McIlroy et al. (2015b) examined this research problem from different perspective, developers' respective and observed that there are positive effects to responding the reviews (users changed their earlier ratings 37.8% of the time) with a median increase of 20% in the rating.

Table 6 presents the list of studies that apply to RQ5 and identifies the related methods, extracted app features and performance.

4. Discussion

The mobile app ecosystem and user reviews contain a wealth of information about user experience and expectations. Developers and app store regulators could leverage the information to better understand their audience. Mining app store data, and in particular user reviews, may provide valuable information for users to reach an informed decision about applications and their features; similarly, it would be valuable for developers to receive user feedback about most liked or expected features, as well as reported bugs in the applications. Mining opinions from app store reviews still requires pre-processing at the content level, including filtering out non-opinionated content and identifying the trustworthiness and genuineness of the opinion and its source. Even though, to date, there is a limited number of research studies analyzing mobile app reviews, the direction and results obtained are promising. Hence, from the perspective of software requirements engineering, with further research, it is expected that app store meta-data will provide a more accurate picture of user choices and expectations. Developers and app store regulators could leverage reviews to better understand their audience. Here we present our principal findings from the SLR.

4.1. Challenges

Challenges in mining app store reviews fall into two main categories:

- The unstructured nature of user reviews and the colloquial language used make the task of extracting application features and user issues from those reviews a challenge, albeit potentially rewarding. Even though some studies useful for data mining user opinion and application features from review texts exist in the literature, the domain and context dependency aspect of the opinion mining problem has not yet been studied for the app ecosystem. Furthermore, the relevance of extracted features has not been cross-validated with the main software engineering concepts.
- Whereas app users rely on the reviews and ratings of others to formulate an informed decision about applications and their features before downloading them, reading all the reviews is time consuming and occasionally deceptive due to misleading or spam reviews. In addition to spam reviews, some reviews do not include useful data for information extraction. Even though some automated systems have been introduced to identify fake and spam reviews and evaluate usefulness, these systems are limited and not yet mature.

4.2. Principal findings

- App store user feedback mining has begun to attract the attention of researchers. Most of the studies selected were of an exploratory nature, based on manual classification and correlation analysis. The number of high-quality app store studies was very limited: we retrieved nine app store mining studies and only one app store spam identification study.
- The automated extraction of app features in online reviews does not consider the nature of the review text. As online app reviews have distinctive features of text (including short length, unstructured phrases, colloquial language and abundant information), there is a need to develop a unique model specific for app store reviews in order to extract targeted app features rather than use conventional methods and techniques developed for different domains and contexts.
- Furthermore, the information requested by users and developers are different. Users are more interested in the opinion and experience of others about the application and which aspects/features they liked or disliked most. Developers have a different point of view when using reviews to:
  - extract usability and user experience information,
  - elicit missing requirements and define requested application features, and
  - improve software quality.
- To deal with abundant information in reviews, external sources such as app crash reports, tweets, community blogs and code repositories could be used to enrich the data. In addition, integration of text with different data sources (such as social media profiles) would be helpful to ensure context level opinion mining, since in terms of preferences and needs, opinions are specific to each person or group.
- Opinion spam or fake review detection is one of the largest problems in the domain. In addition to spam reviews, there are various kinds of user reviews, some of which do not include any useful data for information extraction. Hence, it is necessary to merge multiple criteria not only to identify suspicious reviews but also to differentiate useful reviews from others so that reviews complying with the usefulness criteria can be processed for information extraction. Even though some automated systems have been introduced to identify fake and spam reviews and to evaluate review usefulness, these systems are very limited and not yet mature.

4.3. Future research directions

Our predictions about future of mobile app stores are as following:

We envision that the scale of opinionated text data on Web and mobile app stores will increase tremendously along with other types of big data. While the volume of the big data increases, so do the complexity and relationships underneath the data. Collecting opinions requires concept or semantic level processing and filtering out non-opinionated text data. Users generally prefer to compare specific features of different products. To make such comparisons, researchers need to construct comprehensive common-knowledge bases to spot product features and text polarity. Future opinion-mining systems need broader and deeper commonsense knowledge bases.

On the other hand, the ubiquity of sentiment or opinion analysis as a service (SaaS or OaaS) will make it easy and cheap to embed a SaaS into every application, mobile device and digital experience. Opinion mining and sentiment analysis are inextricably bound to the affective sciences that understand human emotions. Hence, neuroscience and cognitive sciences will inform how opinion mining researchers should measure, analyze and report the

emotions within the text. As there will be more data about a person under one single index, opinion mining will be more specific to user's preferences and needs, predictive sentiment analysis will be another research area to denote the approach in which sentiment analysis is used to predict the changes in the phenomenon of interest.

Our predictions about future of mobile app stores are as following:

Cross-platform and Cross-device Development Creating mobile apps that work easily on multiple platforms (iOS, Android and etc) and devices is presently a challenging task that will not be allowed to persist. Although there is no "one size fits all" approach for mobile app development, we envision the rise in cross-platform mobile development tools. As HTML5 evolves and matures in last couple of year, the future of mobile app development will also make greater use of it to build hybrid mobile apps that will work well across different platforms and devices.

Mobile App Development for Internet of Things (IoT): The future of mobile app development will not be simply about mobile phones and tables, but IoT. As the example of IoT products such as the self-driving cars, the thermostats, the fridges that read the tweets and etc increases and devices start to get more interconnected, the opportunity for software to add value to these smart devices will become even greater.

Search Ads in the App Store: Apple recently began inviting developers to test the App Store's new Search ads that will come to the U.S App Store with IOS 10 in Fall 2016. The introduction of App Store Search Ads will give developers another way to have their apps appear at the top of the results through paid advertisement. This change also brings forward the concern that larger developers could bid more often and win more search ads that will lead their apps to the higher ranks than small developers " apps. However, it is apparent that app users will need others' feedback and reviews more than before to understand if the mobile app really appeals to them, since actual search results will be modified via paid search ads.

References

Andrzejewski, D., Zhu, X., Craven, M., 2009. Incorporating domain knowledge into topic modeling via dirichlet forest priors. In: Proceedings of the 26th Annual International Conference on Machine Learning. ACM, New York, NY, USA, pp. 25–32. doi:10.1145/1553374.1553378.

Aue, A., Gamon, M., 2005. Customizing sentiment classifiers to new domains: a case study. In: Submitted to RANLP-05, the International Conference on Recent Advances in Natural Language Processing, Borovets, BG. URL: http://research.microsoft.com/apps/pubs/default.aspx?id=65430.

Blitzer, J., Dredze, M., Pereira, F., 2007. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In: In ACL, pp. 187–205.

Cambria, E., Schuller, B., Xia, Y., Havasi, C., 2013. New avenues in opinion mining and sentiment analysis. IEEE Intell. Syst. 28 (2), 15–21. doi:10.1109/MIS.2013.30.

Chandy, R., Gu, H., 2012. Identifying spam in the iOS App Store. In: Proceedings of the 2Nd Joint WICOW/AIRWeb Workshop on Web Quality. ACM, New York, NY, USA, pp. 56–59. doi:10.1145/2184305.2184317.

Chen, M., Liu, X., 2011. Predicting popularity of online distributed applications: ITunes app store case analysis. In: Proceedings of the 2011 iConference. ACM, New York, NY, USA, pp. 661–663. doi:10.1145/1940761.1940859.

Chen, N., Lin, J., Hoi, S.C.H., Xiao, X., Zhang, B., 2014. AR-miner: mining Informative Reviews for Developers from Mobile App Marketplace. In: Proceedings of the 36th International Conference on Software Engineering. ACM, New York, NY, USA, pp. 767–778. doi:10.1145/2568225.2568263.

Cheung, C.M., Lee, M.K., Rabjohn, N., 2008. The impact of electronic word-of-mouth: the adoption of online opinions in online customer communities. Internet Res. 18 (3), 229–247.

Clover, J., 2014. MacRumors. URL: http://www.macrumors.com/2014/06/13/apple-fake-app-store-reviews/.

Cosma, A.C., Itu, V.-V., Suciu, D.A., Dinsoreanu, M., Potolea, R., 2014. Overcoming the domain barrier in opinion extraction. In: Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on. IEEE, pp. 289–296.

Dellarocas, C., 2000. Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior. In: Proceedings of the 2Nd ACM Conference on Electronic Commerce. ACM, New York, NY, USA, pp. 150–157. doi:10.1145/352871.352889.

Ding, X., Liu, B., Yu, P.S., 2008. A holistic lexicon-based approach to opinion mining. In: Proceedings of the 2008 International Conference on Web Search and Data Mining. ACM, New York, NY, USA, pp. 231–240. doi:10.1145/1341531.1341561.

Fu, B., Lin, J., Li, L., Faloutsos, C., Hong, J., Sadeh, N., 2013. Why people hate your app: making sense of user feedback in a mobile app store. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York, NY, USA, pp. 1276–1284. doi:10.1145/2487575.2488202.

Gabrilovich, E., Markovitch, S., 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In: Proceedings of the 20th International Joint Conference on Artifical Intelligence. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 1606–1611. URL: http://dl.acm.org/citation.cfm?id=1625275.1625535

Galvis Carreño, L.V., Winbladh, K., 2013. Analysis of user comments: an approach for software requirements evolution. In: Proceedings of the 2013 International Conference on Software Engineering. IEEE Press, Piscataway, NJ, USA, pp. 582–591. URL: http://dl.acm.org/citation.cfm?id=2486788.2486865

Ganapathibhotla, M., Liu, B., 2008. Mining opinions in comparative sentences. In: Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 241–248. URL: http://dl.acm.org/citation.cfm?id=1599081.1599112

Ghose, A., Ipeirotis, P.G., 2011. Estimating the helpfulness and economic impact of product reviews: mining text and reviewer characteristics. Knowl. Data Eng., IEEE Trans. 23 (10), 1498–1512.

Gu, X., Kim, S., 2015. "what parts of your apps are loved by users?" (T). In: 30th IEEE/ACM International Conference on Automated Software Engineering, ASE 2015, Lincoln, NE, USA, November 9–13, 2015, pp. 760–770. doi:10.1109/ASE.2015.57.

Guzman, E., Maalej, W., 2014. How do users like this feature? A fine grained sentiment analysis of app reviews.. In: Gorschek, T., Lutz, R.R. (Eds.), RE. IEEE Computer Society, pp. 153–162. URL: http://dblp.uni-trier.de/db/conf/re/re2014.html#GuzmanM14

Gómez, M., Rouvoy, R., Monperrus, M., Seinturier, L., 2015. A recommender system of buggy app checkers for app store moderators. In: Proceedings of the Second ACM International Conference on Mobile Software Engineering and Systems. IEEE Press, Piscataway, NJ, USA, pp. 1–11. URL: http://dl.acm.org/citation.cfm?id=2825041.2825043

Ha, E., Wagner, D., 2013. Do Android users write about electric sheep? Examining consumer reviews in Google Play. In: Consumer Communications and Networking Conference (CCNC), 2013 IEEE, pp. 149–157. doi:10.1109/CCNC.2013.6488439.

Harman, M., Jia, Y., Zhang, Y., 2012. App store mining and analysis: MSR for app stores. In: Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on, pp. 108–111. doi:10.1109/MSR.2012.6224306.

Hoon, L., Vasa, R., Schneider, J.-G., Mouzakis, K., 2012. A preliminary analysis of vocabulary in mobile app user reviews. In: Proceedings of the 24th Australian Computer-Human Interaction Conference. ACM, New York, NY, USA, pp. 245–248. doi:10.1145/2414536.2414578.

Hu, M., Liu, B., 2004. Mining and summarizing customer reviews. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York, NY, USA, pp. 168–177. doi:10.1145/1014052.1014073.

Hu, X., Liu, H., 2012. Text analytics in social media. In: Mining text data. Springer, pp. 385–414.

Iacob, C., Harrison, R., 2013. Retrieving and analyzing mobile apps feature requests from online reviews. In: Proceedings of the 10th Working Conference on Mining Software Repositories. IEEE Press, Piscataway, NJ, USA, pp. 41–44. URL: http://dl.acm.org/citation.cfm?id=2487085.2487094

Iacob, C., Veerappa, V., Harrison, R., 2013. What are you complaining about?: A study of online reviews of mobile applications. In: Proceedings of the 27th International BCS Human Computer Interaction Conference. British Computer Society, Swinton, UK, UK, pp. 29:1–29:6. URL: http://dl.acm.org/citation.cfm?id=2578048.2578086

Jindal, N., Liu, B., 2008. Opinion spam and analysis. In: Proceedings of the 2008 International Conference on Web Search and Data Mining. ACM, New York, NY, USA, pp. 219–230. doi:10.1145/1341531.1341560.

Keele, S., 2007. Guidelines for performing systematic literature reviews in software engineering. Technical report, Ver. 2.3 EBSE Technical Report. EBSE.

Khalid, H., 2013. On identifying user complaints of ios apps. In: Proceedings of the 2013 International Conference on Software Engineering. IEEE Press, Piscataway, NJ, USA, pp. 1474–1476. URL: http://dl.acm.org/citation.cfm?id=2486788.2487044

Khalid, H., Shihab, E., Nagappan, M., Hassan, A., 2015. What do mobile app users complain about? Software, IEEE 32 (3), 70–77. doi:10.1109/MS.2014.50.

Kim, S.-M., Pantel, P., Chklovski, T., Pennacchiotti, M., 2006. Automatically assessing review helpfulness. In: Proceedings of the 2006 Conference on empirical methods in natural language processing. Association for Computational Linguistics, pp. 423–430.

Kitchenham, B., 2004. Procedures for Performing Systematic Reviews. Keele University Technical Report TR/SE-0401. Department of Computer Science, Keele University, UK.

Korfiatis, N., García-Bariocanal, E., Sánchez-Alonso, S., 2012. Evaluating content quality and helpfulness of online product reviews: the interplay of review helpfulness vs. review content. Electron. Commer. Res. Appl. 11 (3), 205–217.

Alfonso Ureña-López, L., Buenaga, M., Gómez, J.M., 2001. Integrating linguistic resources in tc through wsd. Comput. Hum. 35 (2), 215–230. URL: http://www.jstor.org/stable/30204851

Lim, E.-P., Nguyen, V.-A., Jindal, N., Liu, B., Lauw, H.W., 2010. Detecting product review spammers using rating behaviors. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management. ACM, New York, NY, USA, pp. 939–948. doi:10.1145/1871437.1871557.

Liu, Y., Huang, X., An, A., Yu, X., 2008. Modeling and predicting the helpfulness of online reviews. In: Data Mining, 2008. ICDM '08. Eighth IEEE International Conference on, pp. 443–452. doi:10.1109/ICDM.2008.94.

Luca, M., Zervas, G., 2013. Fake It Till You Make It: Reputation, Competition, and Yelp Review Fraud. Harvard Business School Working Papers. Harvard Business School. URL: https://ideas.repec.org/p/hbs/wpaper/14-006.html

Martin, W., Harman, M., Jia, Y., Sarro, F., Zhang, Y., 2015. The app sampling problem for app store mining. In: Proceedings of the 12th Working Conference on Mining Software Repositories. IEEE Press, Piscataway, NJ, USA, pp. 123–133. URL: http://dl.acm.org/citation.cfm?id=2820518.2820535

Martin, W., Sarro, F., Jia, Y., Zhang, Y., Harman, M., 2016. A Survey of App Store Analysis for Software Engineering. Technical Report. University College London. URL: http://www.cs.ucl.ac.uk/fileadmin/UCL-CS/research/Research_Notes/RN_16_02.pdf

McIlroy, S., Ali, N., Khalid, H., E. Hassan, A., 2015. Analyzing and automatically labelling the types of user issues that are raised in mobile app reviews. Empir. Softw. Eng. 1–40. doi:10.1007/s10664-015-9375-7.

McIlroy, S., Shang, W., Ali, N., Hassan, A., 2015. Is it worth responding to reviews? A case study of the top free apps in the google play store. IEEE Softw. PP (99). doi:10.1109/MS.2015.149. 1–1

Mimno, D., Wallach, H.M., Talley, E., Leenders, M., McCallum, A., 2011. Optimizing semantic coherence in topic models. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 262–272. URL: http://dl.acm.org/citation.cfm?id=2145432.2145462

Moghaddam, S., Jamali, M., Ester, M., 2012. ETF: extended tensor factorization model for personalizing prediction of review helpfulness. In: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining. ACM, New York, NY, USA, pp. 163–172. doi:10.1145/2124295.2124316.

Mojica Ruiz, I., Nagappan, M., Adams, B., Berger, T., Dienst, S., Hassan, A., 2015. An examination of the current rating system used in mobile app stores. Software, IEEE PP (99). doi:10.1109/MS.2015.56. 1–1

Mudambi, S.M., Schuff, D., 2010. What makes a helpful online review? A study of customer reviews on amazon.com. MIS Q. 34 (1), 185–200. URL: http://dl.acm.org/citation.cfm?id=2017447.2017457

Mukherjee, A., Liu, B., 2010. Improving gender classification of blog authors. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 207–217. URL: http://dl.acm.org/citation.cfm?id=1870658.1870679

Newman, D., Karimi, S., Cavedon, L., 2009. External evaluation of topic models. In: Proc. of ADCS 2009, pp. 11–18.

Oh, J., Kim, D., Lee, U., Lee, J., Song, J., 2013. Facilitating developer-user interactions with mobile app review digests. In: 2013 ACM SIGCHI Conference on Human Factors in Computing Systems, CHI '13, Paris, France, April 27–May 2, 2013, Extended Abstracts, pp. 1809–1814. doi:10.1145/2468356.2468681.

Pagano, D., Maalej, W., 2013. User feedback in the appstore: an empirical study.. In: RE. IEEE Computer Society, pp. 125–134. URL: http://dblp.uni-trier.de/db/conf/re/re2013.html#PaganoM13

Pan, S.J., Ni, X., Sun, J.-T., Yang, Q., Chen, Z., 2010. Cross-domain sentiment classification via spectral feature alignment. In: Proceedings of the 19th International Conference on World Wide Web. ACM, New York, NY, USA, pp. 751–760. doi:10.1145/1772690.1772767.

Pan, Y., Zhang, J.Q., 2011. Born unequal: a study of the helpfulness of user-generated product reviews. J. Retailing 87 (4), 598–612. http://dx.doi.org/10.1016/j.jretai.2011.05.002.

Pang, B., Lee, L., 2008. Opinion mining and sentiment analysis. Found. Trends Inf. Retr. 2 (1–2), 1–135. doi:10.1561/1500000011.

Panichella, S., Di Sorbo, A., Guzman, E., Visaggio, C., Canfora, G., Gall, H., 2015. How can I improve my app? Classifying user reviews for software maintenance and evolution. In: Software Maintenance and Evolution (ICSME), 2015 IEEE International Conference on, pp. 281–290. doi:10.1109/ICSM.2015.7332474.

Park, D.H., Liu, M., Zhai, C., Wang, H., 2015. Leveraging user reviews to improve accuracy for mobile app retrieval. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, New York, NY, USA, pp. 533–542. doi:10.1145/2766462.2767759.

Sandulescu, V., Ester, M., 2015. Detecting singleton review spammers using semantic similarity. In: Proceedings of the 24th International Conference on World Wide Web. ACM, New York, NY, USA, pp. 971–976. doi:10.1145/2740908.2742570.

Statista, 2014. Statistics and facts about App Stores. URL: http://www.statista.com/topics/1729/app-stores/.

Tang, H., Tan, S., Cheng, X., 2009. A survey on sentiment detection of reviews. Expert Syst. Appl. 36 (7), 10760–10773. doi:10.1016/j.eswa.2009.02.063.

Tsytsarau, M., Palpanas, T., 2012. Survey on mining subjective data on the web. Data Min. Knowl. Discov. 24 (3), 478–514. doi:10.1007/s10618-011-0238-6.

Vasa, R., Hoon, L., Mouzakis, K., Noguchi, A., 2012. A preliminary analysis of mobile app user reviews. In: Proceedings of the 24th Australian Computer-Human Interaction Conference. ACM, New York, NY, USA, pp. 241–244. doi:10.1145/2414536.2414577.

Vu, P.M., Nguyen, T.T., Pham, H.V., Nguyen, T.T., 2015. Mining user opinions in mobile app reviews: a keyword-based approach. CoRR abs/1505.04657. URL: http://arxiv.org/abs/1505.04657

Wang, G., Xie, S., Liu, B., Yu, P.S., 2011. Review graph based online store review spammer detection. In: Proceedings of the 2011 IEEE 11th International Conference on Data Mining. IEEE Computer Society, Washington, DC, USA, pp. 1242–1247. doi:10.1109/ICDM.2011.124.

Wano, M., Iio, J., 2014. Relationship between reviews at app store and the categories for software. In: Network-Based Information Systems (NBiS), 2014 17th International Conference on, pp. 580–583. doi:10.1109/NBiS.2014.51.

Wohlin, C., 2014. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering. ACM, New York, NY, USA, pp. 38:1–38:10. doi:10.1145/2601248.2601268.

Yang, H., Callan, J., Si, L., 2006. Knowledge Transfer and Opinion Detection in the TREC 2006 Blog Track. In: Proceedings of the Fifteenth Text REtrieval Conference, TREC 2006, Gaithersburg, Maryland, USA, November 14–17, 2006. URL: http://trec.nist.gov/pubs/trec15/papers/cmu.blog.final.pdf

Necmiye Genc-Nayebi is currently pursuing her Ph.D. degree at the École de Technologie Supérieure (ETS) - Université du Québec, Software Engineering Department. Her research topics are Text Mining, Natural Language Processing, Machine Learning and Spam Detection.
She has 11 years of industry expertise on ***Distributed Network Architecture, Service Oriented Architecture (SOA), Virtualization and Integration Concepts (EAI)***. She has hands on experience on software application development, architecture and testing. Embedded, Mobile and Secure Software Development, Algorithm Design and Implementation are some of her major competencies.

Dr. Alain Abran is a Professor and the Director of the Software Engineering Research Laboratory at the École de Technologie Supérieure (ETS) - Université du Québec.
He is currently Co-executive editor of the Guide to the Software Engineering Body of Knowledge project. He is also actively involved in international software engineering standards and is Co-chair of the Common Software Metrics International Consortium (COSMIC).
Dr. Abran has more than 20 years of industry experience in information systems development and software engineering. The maintenance measurement program he developed and implemented at Montreal Trust, Canada, received one of the 1993 Best of the Best awards from the Quality Assurance Institute.