Identification and Classification of Requirements from App User Reviews

Hui Yang
State Key Lab of Software Engineering
School of Computer, Wuhan University, China
huiyang@whu.edu.cn

Peng Liang*
State Key Lab of Software Engineering
School of Computer, Wuhan University, China
liangp@whu.edu.cn

***Abstract*—Review function, as a feedback mechanism from users to developers and vendors, is provided by most APP distribution platforms that allow users to rate and comment an APP after using it. User reviews are recognized as a valuable source to improve APPs and increase the value for users. With the sharp increase in the amount of user reviews, how to effectively and efficiently analyze the user reviews and identify potential and critical user needs from them to improve the APPs becomes a challenge. In this paper, we propose an approach to automatically identify requirements information and further classify them into functional and non-functional requirements from user reviews, using a combination of information retrieval technique (TF-IDF) and NLP technique (regular expression) with human intervention in keywords selection for requirements identification and classification. We validated the proposed approach with the user reviews collected from a popular APP iBooks in English App Store, and further investigated the cost and return of our approach: how the size of sample reviews for keywords selection (cost) affects the classification results in precision, recall, and F-measure (return). The results show that when setting an appropriate size of sample reviews, our approach receives a relatively stable precision, recall, and F-measure of requirements classification, in particular for non-functional requirements, which is meaningful and practical for APP developers to elicit requirements from user reviews.**

Keywords-requirements identification; requirements classification; user review analysis

I. INTRODUCTION

Review function is provided by most APP distribution platforms (e.g., Apple App Store, Google Play) that allow users to rate and comment an APP after using it, which provides a feedback mechanism from users to developers and vendors of the APP. User reviews are recognized as a valuable source to improve APPs and increase the value for users [9][18], as the reviews help developers to better understand user needs as a type of collective knowledge [19]. However, existing APP platforms provide limited support for developers to systematically filter, aggregate, and classify user feedback to derive requirements [9]. User review and rating information has been investigated for technical and business purposes (e.g., APP price prediction) [11]. Pagano and Maalej collected the user reviews of the top 25 APPs from each of the 22 categories from App Store [1]. Based on the review data, they studied the content of user feedback and its impact on the user community. Chandy and Gu proposed an approach to automatically identify spam reviews in the iOS App Store [5]. However, there is little work on systematically and automatically identifying and classifying requirements information from user reviews, which will significantly improve requirements elicitation and analysis in APP development. To this end, we propose an approach to automatically identify requirements information from user reviews and further classify them into functional (FR) and non-functional requirements (NFR), which are the basic classification of software requirements. For the practical application of the proposed approach, we further analyze the cost and return of our approach: how the size of sample reviews for keywords selection (i.e., the cost, described in Section III) affects the classification results in precision, recall, and F-measure (i.e., the return, presented in Section IV.C).

In the rest of this paper: Section II provides an overview of our proposed approach and the tool support. Section III describes the principles, TF-IDF technique, and process of selecting keywords for automated requirements identification and classification. Section IV presents the experiment material (user reviews of a popular APP iBooks) and the experiment results. The implications of the results are discussed in Section V. The threats to validity are described and analyzed in Section VII. Related work is discussed in Section VI. We conclude this work with further work directions in Section VIII.

II. APPROACH AND TOOL SUPPORT

When developing and continuously updating APPs, developers (especially requirements engineers) are responsible for being very much concerned about user experience and needs (e.g., privacy requirements [20]). If the requirements information from user reviews can be automatically identified and classified, it will significantly help developers and vendors to improve the quality and satisfaction of the APPs, for example, collecting critical and missing features for APP update. To this end, we propose an automated approach with tool support for identifying and classifying requirements from user reviews (see Fig. 1). There are two components in this tool: **User Reviews Extractor** is used to extract and collect user review information from APP platforms as raw data to be further processed, and **Requirements Identifier and Classifier** is used to identify and classify requirements from user reviews into FRs and NFRs.

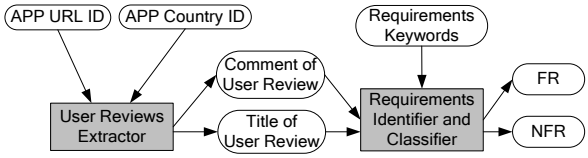

Figure 1. Proposed approach and tool architecture

* Corresponding author
This work is sponsored by the NSFC under Grant No. 61170025.
(DOI reference number: 10.18293/SEKE2015-063)

A. User Reviews Extractor

User Reviews Extractor uses *APP URL ID* and *APP Country ID* as input parameters to extract the user reviews of an APP from a specific APP platform. In the experiment of this work, we extracted and collected user reviews (including *comment* and *title* of user reviews) from APPs in Apple App Store. **User Reviews Extractor** uses the APIs provided by an open source package AppReviews[1] for accessing and retrieving the user reviews from App Store, which provides individual web portal in different countries with local languages. Each country store has its own *APP Country ID*, which allows us to access App Store for each country and retrieve the user review data of a specific APP using *APP URL ID*.

B. Requirements Identifier and Classifier

Requirements Identifier and Classifier is used to automatically identify requirements from user reviews and further classify them into FRs and NFRs. The inputs of Requirements Identifier and Classifier are the *title* and *comment* of user reviews and the extracted keywords (detailed in Section III) and the outputs are FRs and NFRs that are automatically classified. Note that some input user reviews may not contain any requirements information, which are namely spam reviews. These spam reviews[2] are roughly filtered out in **Phase 2** (i.e., pre-processing user reviews). The execution process of this component is composed of five sequential phases as shown in Fig. 2, which are further detailed in this section.

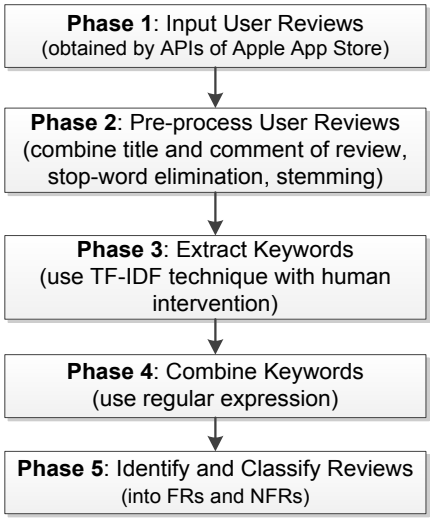

Figure 2. Processing phases of Requirements Identifier and Classifier

Phase 1: Input User Reviews to be processed: Preparing user reviews to be processed obtained by **User Reviews Extractor** as the input of **Requirements Identifier and Classifier**.

Phase 2: Pre-process User Reviews: User reviews obtained by **User Reviews Extractor** are pre-processed by automatically combining the *title* and *comment* of these reviews as the target content, followed by eliminating punctuation marks (such as "," , ".") and stop words in natural language processing, like "*a*", "*the*", and "*this*", filtering out spam reviews (e.g., the reviews less than three words), as well as word stemming [3].

Phase 3: Extract Keywords: In this phase, human experts (e.g., requirements engineers) first manually identify and classify a certain number of user reviews as NFRs or FRs, which are regarded as correct classifications, and then these classified NFRs and FRs are used as sample reviews to extract requirement keywords for automated identification and classification of NFRs and FRs respectively. These requirement keywords are automatically extracted from the sample reviews using TF-IDF technique [16] with human intervention by following the keywords extraction procedure detailed in Section III.B.

Phase 4: Combine Keywords: **Requirements Identifier and Classifier** combines the extracted keywords, the requirement keywords from each sample review (obtained from **Phase 3**), in various logical relationships (e.g., OR "|") of regular expressions (e.g., *bug|crash*). These regular expressions are used to match (identify and classify) user requirements from user reviews in **Phase 5**. For example, for FR, *^is|are*choice$*, which represents such phrases "*is ... choice*" or "*are ... choices*".

Phase 5: Identify and Classify User Reviews: User requirements are identified and classified from the pre-processed content of reviews (obtained from **Phase 2**) using the regular expressions (obtained from **Phase 4**). A user review is automatically identified as requirement and classified into a NFR (or FR) using the regular expressions if the review can match the regular expressions (obtained from **Phase 3**). Note that, the identification and classification of requirements are performed in one step.

III. KEYWORDS SELECTION

A. Sample Reviews

According to the description in [14], a functional requirement specifies "a function that a system must be able to perform", "what the product/system should do", and a non-functional requirement is restricted to a set of specific qualities other than functionality: such as usability, reliability, and security. For example, a user review: "*the loss of the bookshelf look, the boring and ugly flat design plus the stark white background make it extremely difficult to read anything on this app*." can be manually classified by domain experts as a NFR usability; another user review: "*at least give me the option of how I would prefer it to look.*" can be categorized as a FR that allows users to configure the style of UI. These manually identified and classified NFRs and FRs are used as sample reviews to extract keywords for NFR and FR identification and classification.

B. Keywords Extraction

As shown in Fig. 1, the requirement keywords are used to identify requirements information from user reviews and further to classify them into FRs and NFRs. The selection of

[1] http://www.perculasoft.com/appreviews/

[2] We are not intending to filter out all spam reviews, but only the obvious spams to improve the efficiency of subsequent processing.

the keywords is critical to the quality of the requirements identification and classification results.

In the field of information retrieval, Term Frequency - Inverse Document Frequency (TF-IDF) [16] is a statistic-based technique used to reflect how important a word is to a document in a collection or corpus. This technique has been successfully applied to text mining and classification (e.g., [15]). We use TF-IDF to calculate and evaluate the importance of a word extracted from each sample NFR (or FR) review to the set of sample NFR (or FR) reviews that are manually classified by domain experts. TF means the importance of a word extracted from each sample NFR (or FR) review to the sample review. The words that obtain a high TF-IDF score in each sample review require further checking by human experts, who judge and select the keywords which act as representative keywords of the sample review. For example, "*privacy*" is not considered as the keyword for FR, and "*feature*" is filtered out from the keywords for NFR. The selection criteria employed by human experts are very simple: for **FR**, the words which typically represent the NFR information should be excluded from the FR keywords (e.g., "*privacy*", "*security*", "*usability*", and "*crash*"); for **NFR**, the words which typically represent the FR information should also be excluded from the NFR keywords (e.g., "*feature*" and "*choice*").

The formulas for calculating the TF-IDF score of each word [16] are as follows (Formula (1) and (2) are used for calculating the TF-IDF score of NFR and FR words respectively), which are further explained below.

$$\text{Score}_{nfr}(\text{w}) = \frac{freq(\text{Rv},\text{w})}{|\text{Rv}|} \times \log\frac{|\text{R(w)}|}{Na(\text{w})} \times \frac{Nnfr(\text{w})}{Na(\text{w})} \quad (1)$$

$$\text{Score}_{fr}(\text{w}) = \frac{freq(\text{Rv},\text{w})}{|\text{Rv}|} \times \log\frac{|\text{R(w)}|}{Na(\text{w})} \times \frac{Nfr(\text{w})}{Na(\text{w})} \quad (2)$$

Each word *w* in a sample review (NFR or FR) will obtain a TF-IDF score $Score_{nfr}(w)$ or $Score_{fr}(w)$, which represents the importance of the word *w* in identifying and classifying user reviews. $freq(Rv,w)/|Rv|$ denotes the TF (term frequency) section of TF-IDF, in which $|Rv|$ refers to the quantity of all the words contained in the review *Rv* and $freq(Rv,w)$ represents the frequency of word *w* appearing in the sample review *Rv*. $log(|R(w)|/Na(w))$ represents the IDF (inverse document frequency) section of TF-IDF, in which $Na(w)$ denotes the number of sample reviews that contain the word *w*, and $|R(w)|$ denotes the number of reviews to be classified that contain the word *w*. $Nnfr(w)$ or $Nfr(w)$ represents the number of NFR or FR sample reviews that contain the word *w*. $Nnfr(w)/Na(w)$ in Formula (1) or $Nfr(w)/Na(w)$ in Formula (2) implies if the word *w* is more densely distributed in the set of sample NFR or FR reviews, the word *w* is more important (i.e., $Score(w)$ is higher) in identifying and classifying NFRs or FRs from user reviews.

According to the obtained TF-IDF score of each word, the words are extracted from each sample review as representative requirement keywords of this sample review, and they are added into the requirement keywords set (duplicated keywords are removed). When keywords are extracted from all sample reviews and added to the keywords set, the keywords selection process is finished. The requirement keywords set is then used to identify and classify requirements from user reviews. One user review can be classified as NFR or FR when it contains (can match) the requirement keywords of NFR or FR in the requirement keywords set.

IV. EXPERIMENT

A. Experiment Material

iBooks is a popular APP in the books category to read and buy books online through various Apple devices. This APP is provided for free in App Store. We decided to choose the user reviews of iBooks APP in English App Store as experiment material for the following reasons: (1) there are a large number of users of iBooks APP, which provide rich review data for the experiment; (2) the user reviews of this APP can be easily classified without the necessity of much domain knowledge, which improves the reliability of the experiment results (further discussed in Section VI); and (3) the review data in English is widely understandable which might act as benchmark data for other researchers to repeat this experiment using their own classification methods and tools.

B. Selected Keywords

As described in Section III, keywords are selected from a set of manually classified sample reviews. To investigate the cost and return of our approach, i.e., how the size of sample reviews (cost) for keywords selection affects the classification results (return), we provide increasing sizes of sample reviews as follows: 1, 3, 5, 7, 9, 10, 20, 30, 50, and 100. It is worth noting that these sets of sample reviews are independent of each other (i.e., one user review cannot exist in two sample sets). We then extracted the keywords from different size of sample reviews for identifying and classifying user requirements by following the keywords extraction procedure (in Section III.B). We first extracted the keywords from the latest "1" user review of iBooks APP, and then we iterated the keywords selection steps towards the remaining latest 3, 5, 7, 9, 10, 20, 30, 50, and 100 reviews from iBooks. Finally, all the selected keywords from each set of sample reviews are collected in an XML file and used to identify and classify requirements from the user reviews of iBooks. The requirement keywords, extracted using TF-IDF for each set of sample reviews, and further checked by human experts (see Section III.B), are available online[3].

C. Experiment Results

To investigate the effectiveness of our approach, we compare the manual classification results by experts (the two authors), which act as ground truth, with the identification and classification results. The experiment use 1000 user reviews as experiment material retrieved from iBooks APP in English App Store. For the practical application of the approach, we further analyze the cost and return of our approach as discussed in Section IV.B, i.e., the experiment results are further evaluated and compared with different sizes of sample reviews (cost) using precision, recall, and F-measure of the classification results (return). The experiment results are presented below.

[3] http://www.cs.rug.nl/search/uploads/Resources/TF-IDF-Keywords.zip

In iBooks APP from English App Store, we obtained 217 (set **B** in Fig. 3) user reviews containing FR information and 622 user reviews containing NFR information among the 1000 user reviews (some user reviews may contain both FR and NFR information) by manual classification (i.e., the ground truth). To examine that how the size of sample reviews affects the classification results, we provide the sizes of sample reviews as follows: 1, 3, 5, 7, 9, 10, 20, 30, 50, and 100, which is shown in the *x*-axis of Fig. 4 and Fig. 5.

We evaluate our approach through comparing automated classification results with manual classification results. We use F-measure which is a combination of precision and recall used in the evaluation of information retrieval systems [2] to measure the overall performance of the automated classification results.

In this work, precision means the percentage of user reviews that are correctly classified as FRs or NFRs compared to all the classified results (i.e., set **A** divided by **C** as illustrated in Fig. 3), and the recall refers to the percentage of user reviews that are correctly classified as FRs or NFRs compared to the manual classification results - the ground truth (i.e., set **A** divided by **B** in Fig. 3).

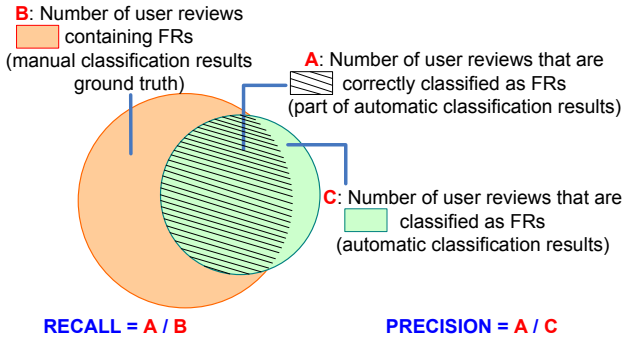

Figure 3. Recall and Precision calculation for classification results evaluation

We calculate and get the evaluation results of FR and NFR classification as shown in TABLE I and TABLE II respectively (including the size of sample reviews, the number of extracted keywords using TF-IDF, Precision, Recall, and F-measure for FR and NFR classification). Fig. 4 and Fig. 5 show the trend line of Recall, Precision, and F-measure for FR and NFR classification results on iBooks user reviews along with different sample sizes of user reviews. These two figures show that the value of F-measure (represented in blue line) is significantly increased as the size (number) of sample reviews increases, but when the size of sample reviews reaches a certain threshold, the value of F-measure tends to be stable. The possible explanation of the experiment results and their implications will be discussed in Section V.

TABLE I. RESULTS OF AUTOMATED FR CLASSIFICATION ON 1000 IBOOKS USER REVIEWS WITH DIFFERENT SIZES OF SAMPLE REVIEWS

Sample Size	No. of Keywords	Precision	Recall	F-measure
1	5	0.000	0.000	0.000
3	15	0.406	0.129	0.196
5	18	0.454	0.184	0.262
7	20	0.469	0.207	0.288
9	24	0.404	0.281	0.332
10	26	0.394	0.249	0.305
20	53	0.385	0.525	0.444
30	56	0.356	0.710	0.474
50	75	0.383	0.636	0.478
100	116	0.350	0.760	0.479

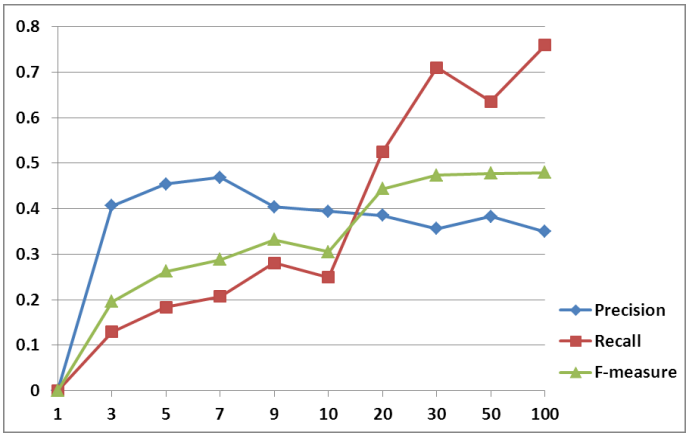

Figure 4. Trend lines of Precision, Recall, & F-measure for FR classification on 1000 iBooks user reviews with different sizes of sample reviews

TABLE II. RESULTS OF AUTOMATED NFR CLASSIFICATION ON 1000 IBOOKS USER REVIEWS WITH DIFFERENT SIZES OF SAMPLE REVIEWS

Sample Size	No. of Keywords	Precision	Recall	F-measure
1	5	0.909	0.032	0.062
3	12	0.837	0.215	0.342
5	16	0.836	0.418	0.557
7	25	0.816	0.740	0.776
9	22	0.820	0.698	0.754
10	28	0.826	0.704	0.760
20	43	0.795	0.810	0.803
30	57	0.758	0.897	0.823
50	82	0.761	0.895	0.823
100	104	0.745	0.924	0.825

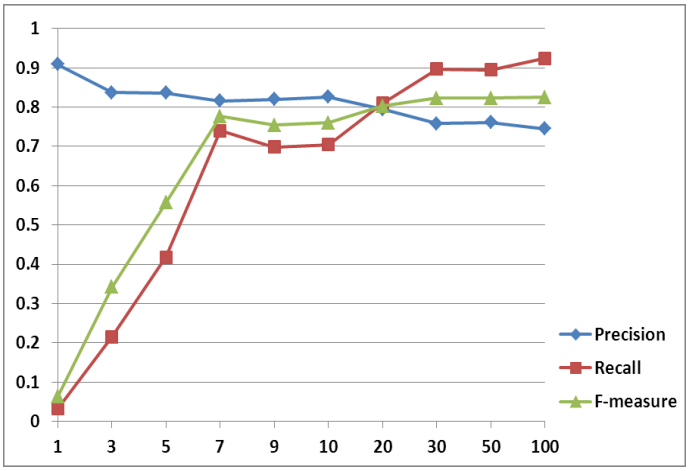

Figure 5. Trend lines of Recall, Precision, & F-measure for NFR classification on 1000 iBooks user reviews with different sizes of sample reviews

V. DISCUSSION

We explain the experiment results and discuss their implications according to the visualization in Fig. 4 and Fig. 5.

1) In both Fig. 4 and Fig. 5, the value of F-measure dramatically increases when the sample size increases initially (e.g., from 1 to 20 for FR classification, and from 1 to 7 for NFR classification), and after that size (number) the value of F-measure tends to be stable. This result implies that there is a certain threshold of sample size that can achieve a comparably good and balanced classification results without the necessity to increase the sample size unceasingly.

2) The significant difference between the thresholds of sample size for FR and NFR classification (20 vs. 7 in this experiment) implies that NFR classification requires less sample reviews to get a decent set of keywords reaching a stable F-measure than FR classification, which is reasonable since the FR keywords are more domain-dependent than the NFR keywords. This may also explain a relatively higher F-measure (e.g., when the sample size is 100) of the NFR classification results (0.825) than the FR results (0.479).

3) From both Fig. 4 and Fig. 5, it can be found that the three trend lines of Precision, Recall, and F-measure have an intersection point (e.g., a sample size (number) between 10 to 20 for FR classification, and 20 for NFR classification), and after that size (number) the value of F-measure tends to be stable. This intersection point seems providing a reliable way to decide the threshold of sample size as discussed in point (1) for a balanced (cost vs. return) classification results. But this conjecture should be validated with more experiments (APPs).

VI. THREATS TO VALIDITY

We discuss the threats to the validity by following the guidelines in [4] and how they are partially mitigated.

Construct validity: We use F-measure from information retrieval theory to evaluate the requirements classification results. The automated requirements classification with our approach is basically an information retrieval activity since both of them use keywords to get results.

Internal validity focuses on the unknown factors that may have an influence on the study results. This experiment is a study about the performance of the proposed approach (to what extend the approach can identify and classify requirements from user reviews) using descriptive statistics. In other words, we did not investigate and intend to establish any causal relationship between the identification and classification results and the factors that may impact the results in this study, and the threats to internal validity are minimal.

External validity: We applied our approach to a popular APP from the books category (application domain) in English App Store with promising results. This experiment can be repeated with APPs in other domains and languages to improve the applicability and generalizability of the proposed approach.

Reliability: The manual requirements classification results by experts are regarded as ground truth to be compared with the automated classification results for the evaluation (in Section IV.C), but the manual classification results might be different when conducted by different experts, which makes the evaluation results not reliable. We tried to mitigate the influence of this issue by three measures: (1) we selected a general APP iBooks and its reviews can be reliably classified without the need of much domain knowledge. (2) the manual classification results by the first author were checked by the second author, and any disagreement on the classification results was discussed and resolved. (3) the manual classification results were further examined by 10 master students, who major in software engineering through voting. We also set criteria (see Section III.B) to select requirement keywords by experts, and this mitigates the bias when different experts select representative keywords from the keywords obtained by TF-IDF.

VII. RELATED WORK

We summarize and discuss relevant work and their relationship to our work in this section.

Chen and his colleagues proposed a method to automatically mine informative reviews for APP developers, and further rank these informative reviews [21]. Our work aims to identify and classify the informative reviews that contain requirement information as functional and non-functional requirements.

Khalid and his colleagues [17] focused on low star-rating user reviews of free iOS APPs, and identified 12 types of complaints that users complain about. They found that functional errors, feature requests, and APP crashes are the most frequent complaints, which supports that user reviews are indeed rich source of requirements.

Pagano and Maalej presented an empirical study on user feedback in the App Store [1]. They mainly discussed the usage of user feedback by the users, the content of user feedback, and its impact on the user community in the App Store, through analyzing the App Store review data with statistical approaches. They also discussed the impact of user feedback to requirements engineering, which inspires our work.

Galvis Carreño and Winbladh focused on changing requirements and creating new requirements using the topics identified from user reviews [13], while our work is different in that we try to identify original requirements and classify them from user reviews. The outcome of our approach can act as input (identified and classified user requirements) of [13] for topics identification in requirements evolution.

Chandy and Gu proposed an approach to automatically identify spam reviews in the iOS App Store [5], which compared the performance of a baseline Decision Tree model with a novel Latent Class graphical model to the classification results of App review spam. The difference of their work to our work is that they employ data mining techniques and focus on spam identification.

Finkelstein and his colleagues [11] introduced a method to extract feature and price information of the APPs in the Blackberry App Store for an analysis that combines technical, business, and customer properties. The analysis results are further used as the input to predict the prices of APPs with

case-based reasoning, while our work focuses on the extraction of user requirements information from APP user reviews.

Sagar and Abirami investigated conceptual modeling of FRs in natural language [10]. For the purpose of visualizing the FRs, they focus on automated extraction of concepts and their relationships to create a conceptual model based on linguistic aspects of the English language. Their work could be useful to develop the conceptual model for FR identification and classification from user reviews.

The work on mining general and APP repositories focuses on analyzing the feature information among user reviews, and understanding their inter-relationships with other factors, e.g., rating, price, downloads, and code [6][7][11]. Our approach tries to combine App Store reviews mining and requirements engineering to help developers understand the trend of software products in order to improve their APPs.

VIII. CONCLUSIONS AND FUTRUE WORK

In this paper, we present an approach, which can automatically identify and classify requirements from user reviews. We validated the proposed approach with user reviews collected from a popular APP iBooks in English App Store, and further investigated the cost and return of our approach: how the size of sample reviews for keywords selection (cost) affects the classification results in precision, recall, and F-measure (return). The results show that when setting an appropriate size of sample reviews, our approach receives a relatively stable precision, recall, and F-measure of requirements classification, in particular for non-functional requirements, which is meaningful and practical for APP developers to elicit requirements from user reviews. In the next step, the approach can be improved in three promising aspects:

1) To validate our approach with user reviews of APPs from other application domains (e.g., social networking, finance) and languages (e.g., East Asian languages) and perform comparative studies with other identification and classification approaches (e.g., through data mining, machine learning techniques) in order to mitigate the threats to the external validity of the results.

2) The identified and classified requirements can be further prioritized to show their importance when hundreds-and-thousands of requirements flooding to developers [8]. The potential factors for prioritizing requirements from user reviews are different from those for general requirements prioritization, for example, rating information, length of user review, and stickiness or importance of the user who submitted the review. All these factors are expected to have an influence on prioritizing requirements from use reviews, and other potential factors should also be considered depending on the needs of requirements prioritization in context.

3) Functional and non-functional requirements are not independent of each other [14], for example, one NFR may impact many FRs, which is an important part of requirements traceability. The potential relationships between classified FRs and NFRs can be promisingly identified through their source analysis, e.g., the user-review relationships.

REFERENCES

[1] D. Pagano and W. Maalej, "User feedback in the AppStore: An empirical study," In: Proceedings of the 21st International Conference on Requirements Engineering (RE), IEEE, 2013, pp. 125-134.

[2] R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval. vol. 463. New York. ACM Press, 1999.

[3] J. B. Lovins, Development of A Stemming Algorithm. MIT Information Processing Group, Electronic Systems Laboratory, 1968.

[4] F. Shull, J. Singer, and D. I. Sjøberg (Eds.), Guide to Advanced Empirical Software Engineering. Springer, 2008.

[5] R. Chandy and Gu. H, "Identifying spam in the iOS app store," In: Proceedings of the 2nd Joint WICOW/AIRWeb Workshop on Web Quality (WebQuality), ACM, 2012, pp. 56-59.

[6] M. Harman, Y. Jia, and Y. Zhang, "App Store mining and analysis: MSR for App Stores," In: Proceedings of the 9th Working Conference on Mining Software Repositories (MSR), IEEE, 2012, pp. 108-111.

[7] A. Zaidman, B. Van Rompaey, S. Demeyer, and A. van Deursen, "Mining software repositories to study co-evolution of production & test code". In: Proceedings of the 1st International Conference on Software Testing, Verification, and Validation (ICST), IEEE, 2008, pp. 220-229.

[8] S. Gartner and K. Schneider, "A method for prioritizing end-user feedback for requirements engineering," In: Proceedings of the 5th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE), IEEE, 2012, pp. 47-49.

[9] U. Abelein, H. Sharp, and B. Paech, "Does involving users in software development really influence system success?" IEEE Software, IEEE, 30(6):17-23, 2013.

[10] V. Sagar and S. Abirami, "Conceptual modeling of natural language functional requirements," Journal of Systems and Software, Elsevier, 88(2):25-41, 2014.

[11] A. Finkelstein, M. Harman, Y. Jia, F. Sarro, and Y. Zhang, "Mining App Stores: Extracting technical, business and customer rating information for analysis and prediction," Research Note, RN/13/21, 2013.

[12] J. Cleland-Huang, R. Settimi, X. Zou, and P. Solc, "Automated classification of non-functional requirements," Requirements Engineering, Springer, 12(2):103-120, 2007.

[13] L.V. Galvis Carreño, and K. Winbladh, "Analysis of user comments: An approach for software requirements evolution," In: Proceedings of the 35th International Conference on Software Engineering (ICSE). IEEE, 2013, pp. 582-591.

[14] L. Chung, B. Nixon, E. Yu, and J. Mylopoulos, Non-functional Requirements in Software Engineering. Springer, 2000.

[15] M.K. Dalal, and M.A. Zaveri, "Automatic text classification of sports blog data", In: Proceedings of the 2012 Computing, Communications and Applications Conference (ComComAp), IEEE, pp.219-222, 2012.

[16] J. Ramos, "Using tf-idf to determine word relevance in document queries," In: Proceedings of the 1st Instructional Conference on Machine Learning (iCML), 2003, pp. 119-122.

[17] H. Khalid, E. Shihab, M. Nagappan, and A. Hassan, "What do mobile App users complain about? A study on free iOS Apps," IEEE Software, IEEE, DOI: http://dx.doi.org/10.1109/MS.2014.50, 2014.

[18] M. Bano and D. Zowghi, "User involvement in software development and system success: A systematic literature review," In: Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering (ESEM), ACM, 2013, pp. 125-130.

[19] P. Liang, P. Avgeriou, K. He, and L. Xu, "From collective knowledge to intelligence: pre-requirements analysis of large and complex systems," In: Proceedings of the 1st Workshop on Web 2.0 for Software Engineering (Web2SE), ACM, 2010, pp. 26-30.

[20] K. Thomas, A.K. Bandara, B.A. Price, and B. Nuseibeh, "Distilling privacy requirements for mobile applications," In: Proceedings of the 36th International Conference on Software Engineering (ICSE), ACM, 2014, pp. 871-882.

[21] N. Chen, J. Lin, S.C.H. Hoi, X. Xiao, and B. Zhang, "AR-Miner: mining informative reviews for developers from mobile app marketplace," In: Proceedings of the 36th International Conference on Software Engineering (ICSE), ACM, 2014, pp. 767-778.