Finding and Analyzing App Reviews Related to Specific Features: A Research Preview

Jacek Dąbrowski[1,2][0000−0003−3392−0690], Emmanuel Letier[1][0000−0002−8935−343X], Anna Perini[2][0000−0001−8818−6476], and Angelo Susi[2][0000−0002−5026−7462]

[1] University College London, London, UK
`{j.dabrowski, e.letier}@cs.ucl.ac.uk`
[2] Fondazione Bruno Kessler, Trento, Italy
`{dabrowski, perini, susi}@fbk.eu`

Abstract. [Context and motivation] App reviews can be a rich source of information for requirements engineers. Recently, many approaches have been proposed to classify app reviews as bug reports, feature requests, or to elicit requirements. **[Question/problem]** None of these approaches, however, allow requirements engineers to search for users' opinions about specific features of interest. Retrieving reviews on specific features would help requirements engineers during requirements elicitation and prioritization activities involving these features. **[Principal idea/results]** This paper presents a research preview on our tool-supported method for taking requirements engineering decisions about specific features. The tool will allow one to (i) find reviews that talk about a specific feature, (ii) identify bug reports, change requests and users' sentiment about this feature, and (iii) visualize and compare users' feedback for different features in an analytic dashboard. **[Contributions]** Our contribution is threefold: (i) we identify a new problem to address, i.e. searching for users' opinions on a specific feature, (ii) we provide a research preview on an analytics tool addressing the problem, and finally (iii) we discuss preliminary results on the searching component of the tool.

Keywords: Mining Users Reviews · Feedback Analytics Tool · Software Quality · Requirement Engineering.

1 Introduction

Developing app reviews analytics tools is an active field of research aimed at extracting useful information from the large amount of user reviews found in app stores [12]. Analytics tools exploit data mining and analysis techniques to address different software engineering problems, including requirements engineering problems. Approaches have been proposed for inferring topics referred by reviews [14], for analyzing users' sentiments [5], and for classifying reviews as either bug reports, requests for new features [9], or discussion about non-functional properties [10,12].

However, these approaches do not allow software professionals to search for users' opinions on specific features of interest. Software professional interviewed about feedback analytics tools indicate two missing features they would like to see supported in future analytics tools: the ability to group and quantify app reviews to support requirements prioritization, and the ability to associate app reviews to work items in project management and issue tracking systems [8].

Our objective is to develop automated techniques to satisfy these requests. We present an automated tool that given a short feature description (e.g. *add reservations* for a Google Trip app), finds app reviews that refer to such feature and report historical trends about users sentiments, bug reports and enhancement requests related to this feature. Knowing what users' say about a specific feature is important to understand their needs [2,13]. It may support engineers to monitor users' satisfaction on a feature and its "health condition" over time [10]. This can also help to sketch a roadmap for next release, including decisions on which features should be changed first to improve the app performance [15].

The following questions guide the development and evaluation of our tool:

RQ1. What natural language and data analysis techniques can be used to develop our tool?

RQ2. What is the effectiveness of different techniques in searching for users' opinions on a feature?

RQ3. How useful do requirements engineers find the tool?

This research preview paper motivates our research through an example, outlines our envisioned technical solution, presents preliminary results for our app reviews search engine, relates our approach to previous work, and discuss our future research plans.

2 Motivating Scenarios

Google Trip app is an app that helps its users to organize trips and manage travel-oriented documents. The app is being used by more than 27,275 users and received over 8,500 reviews on Google Play Store. These reviews concern existing or desired functionalities, reported bugs and other aspects related to quality in-use. We use this app in our initial experiment in Section 4.

Suppose the issue tracking system for this app contains requests for introducing new features and improving existing features (for example *add the ability to create day plans*, *improve the ability to add a hotel reservation*, etc.) and the project manager has to decide which requests to implement first. Finding users reviews mentioning each of these features would allow the project managers to quickly compare how often each request appears in app reviews, for how long each request has been made, and whether the frequency of each request is increasing or decreasing. This information will provide concrete evidence of the relative importance of each request from the users' perspective. Such information is not sufficient by itself to prioritize change request because the perspective of other stakeholders must also be taken into account, but it can provide useful evidence-based data to partly inform such decisions.

Suppose now that a requirements engineer and the development team have been tasked to define and implement detailed requirements for one of these feature requests. Finding users reviews that refer to the feature will allow them to quickly identify what users have been saying about the feature. This cheap elicitation technique might be sufficient in itself or it might be the starting point for additional more expensive elicitation activities involving interviews, surveys, prototyping, or observations.

3 An Approach for Analyzing Users' Feedback on Feature

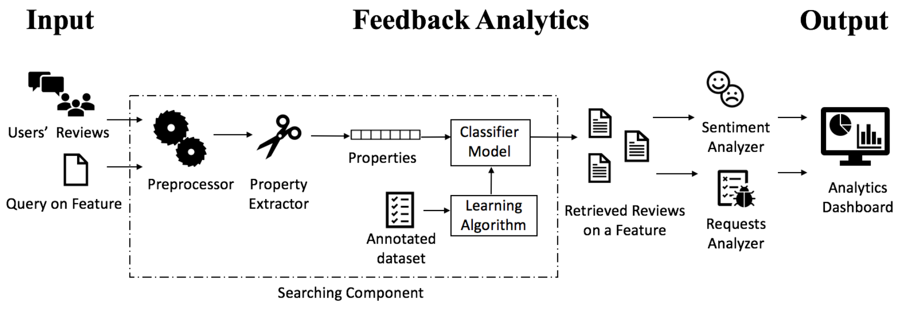

Fig. 1: Overview of our tool-based approach for finding and analysing users' opinions on a specific feature

Figure 1 outlines our approach for finding and analysing users' reviews related to specific features. The main component of our tool is the searching component that takes as input a query that describes a feature (e.g. *add reservations*) and retrieves a set of users reviews that mention that feature. An example of review retrieved for the feature query *add reservations* is *Please, improve adding reservations as it crashes and obstructs the booking process*. Sentiment analysis and classification techniques are then used to classify the retrieved reviews as expressing either positive, neutral or negative sentiments and as reporting a bug or asking for an enhancement. The results of the search for different feature queries are then presented on a dashboard.

We are exploring the following techniques to develop our tool:

Searching Component. We propose *machine-learned relevance* method to support searching for users' feedback on a feature [11]. The method exploits supervised machine learning (ML) techniques to classify reviews to be relevant or non-relevant to a query. Figure 1 illustrates details of searching component to be used for the method. The preprocessor performs standard text normalization steps of query and reviews to refine them from noisy information. The property extractor determines textual properties of filtered query and reviews, then convey them to the classifier as basis for selecting reviews to be returned.

To produce a classification model, the learning algorithm is firstly provided with a training dataset including exemplary reviews annotated with respect to test queries.

Sentiment Analyzer. Two candidate techniques could be used to analyze sentiment of users' reviews: ML and lexicon-based [6]. ML techniques treat sentiment identification as binary or multiclass classification problem. The lexicon-based methods calculate the sentiment score of a word or sentence using lexicon provided with list of positive and negative words. These methods assume that the opinion of text is determined as the sum of the sentiment score of each word. Further, we propose to analyze sentiment on the aspect-level as it allows one to determine the sentiment within a segment of text for a mentioned feature, rather than for the text as a whole [6].

Users' Requests Analyzer. We aim to use one of existing data-driven methods to classify users' request as bug report or feature request. These methods exploit supervised classification techniques such as Naive Bayes, Support Vector Machine or Logistic Regression, and proved their effectiveness for classifying users' request into requirement-related information [4].

4 Preliminary Results

To evaluate the feasibility of searching for users' opinions on a feature we conducted a preliminary experiment. We collected 200 reviews for Google Trip app and the app description from Play Store. We manually extracted feature-related phrases from the description using Part Of Speech patterns and annotated the reviews with respect to these phrases [7]. We then built a prototype of the searching component using NLTK library and Weka tool. We trained algorithms with text properties such as query-term proximity, covered query term number, cosine similarity measure and Boolean matching [11]. We evaluated our prototype using 10-fold cross-validation and obtained precision of 0.360, recall of 0.257 and F1 score of 0.300. We observed that for queries formed by two keywords (e.g. *add reservation*) and term proximity less of than three words, the approach achieve precision at the level of 0.88. Furthermore, we observed that reviews discussing a queried feature by their synonyms are not retrieved. This problem could be addressed by query expansion or word embedding techniques.

Further, we concluded that some queries (e.g. *search for attractions*) express a functional topic aggregating several real features (e.g. *search for place* or *search for restaurant*) rather than a single feature. We plan to investigate whether we could use technique for ontology inference based on app description and reviews and extend our approach by concept similarity measure.

5 Related Work

Previous work focused on inferring features in app reviews rather than finding features that talk about specific features [5,1,7]. Guzman and Maalej proposed an approach for analyzing sentiments of reviews where prospective app features are identified. The approach identifies features as frequently co-occurring keywords and extract them from users' reviews. Extracted features are associated with sentiment and then grouped using topic modelling. The authors extended the work by classifying reviews associated with extracted features into categories related to usability and user experience [1]. Similarly, Johann et al. proposed an approach for extracting app features from users' reviews and app description based on linguistic rules [7]. The approach enables comparing lists of extracted app features to identify mismatch between extracted app features from reviews and app description.

These approaches identify phrases corresponding to app features and extract them from users reviews. They are evaluated against their ability to identify whether extracted phrases from reviews are really features. In contrast, our approach aims to support software professionals to search for users' feedback on specific features of their interest. Therefore, we plan to assess our tool's ability to retrieve users' feedback on a queried feature.

Other works provide tool-based approaches to support feedback analysis [3,16,17]. The PAID approach groups frequently co-occurring keywords extracted from users' reviews and visualize them as topics by theme river [3]. The main objective of the tool is to visualize changes in topics in different versions of the app. MARK is a keyword-based tool for a semi-automated review analysis [16]. It enables one to automatically extract keywords from raw user reviews and rank them using their associations with negative sentiment. The tool provides a summary of the most relevant reviews related to the keywords and visualizes the trend of keyword occurrence. Similarly, PUMA extracts phrases from reviews which are associate with negative sentiment and visualize how sentiments evolve over a specific time period [17].

We envision our tool will use sentiment and trend analysis techniques similar to these used by previous app store analysis tools, but will perform a more fine-grained analysis on the subsets of reviews retrieved by our searching component.

6 Conclusion

In this research preview, we have presented a problem of searching for users' opinions on a specific feature. We demonstrated the problem and its relevance to support requirement engineering decisions by motivating scenarios. We proposed our tool-based approach to address the problem and analyze retrieved reviews in terms of their sentiments and users' requests. We presented preliminary results on the feasibility of the approach and technical challenges that need to be addressed.

As future work, we plan to implement remaining components of the tool and experiment with different techniques to elaborate our approach. In particular, we aim to investigate unsupervised techniques to support searching for opinionated features and analyzing associated sentiments expressed in user reviews. We plan to user and extend available datasets to evaluate our work [16]. We will select apps from different domains and app stores to investigate the generality of our approach. Further, we will use software professionals to (i) identify candidate features from app descriptions to form test queries, and to (ii) annotate users' feedback with respect to expressed users' request, opinionated feature and associated sentiment.

Finally, we will evaluate the usefulness of our tool in practice by observing and interviewing prospective users.

References

1. Bakiu, E., and Guzman, E. Which feature is unusable? detecting usability and user experience issues from user reviews. In *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)* (Sept 2017), pp. 182–187.
2. Begel, A., and Zimmermann, T. Analyze this! 145 questions for data scientists in software engineering. In *Proceedings of the 36th International Conference on Software Engineering* (New York, NY, USA, 2014), ICSE 2014, ACM, pp. 12–23.
3. Gao, C., Wang, B., He, P., Zhu, J., Zhou, Y., and Lyu, M. R. PAID: Prioritizing app issues for developers by tracking user reviews over versions. In *2015 IEEE 26th International Symposium on Software Reliability Engineering (ISSRE)* (Nov 2015), pp. 35–45.
4. Guzman, E., El-Haliby, M., and Bruegge, B. Ensemble methods for app review classification: An approach for software evolution (n). In *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)* (Nov 2015), pp. 771–776.
5. Guzman, E., and Maalej, W. How do users like this feature? a fine grained sentiment analysis of app reviews. In *2014 IEEE 22nd International Requirements Engineering Conference (RE)* (Aug 2014), pp. 153–162.
6. Hemmatian, F., and Sohrabi, M. K. A survey on classification techniques for opinion mining and sentiment analysis. *Artificial Intelligence Review* (Dec 2017).
7. Johann, T., Stanik, C., Alizadeh B., A. M., and Maalej, W. SAFE: A simple approach for feature extraction from app descriptions and app reviews. In *25th IEEE International Requirements Engineering Conference, RE 2017, Lisbon, Portugal, September 4-8, 2017* (2017), pp. 21–30.
8. Maalej, W., Kurtanović, Z., Nabil, H., and Stanik, C. On the automatic classification of app reviews. *Requirements Engineering 21*, 3 (2016), 311–331.
9. Maalej, W., and Nabil, H. Bug report, feature request, or simply praise? on automatically classifying app reviews. In *2015 IEEE 23rd International Requirements Engineering Conference (RE)* (Aug 2015), pp. 116–125.
10. Maalej, W., Nayebi, M., Johann, T., and Ruhe, G. Toward data-driven requirements engineering. *IEEE Software 33*, 1 (Jan 2016), 48–54.
11. Manning, C. D., Raghavan, P., and Schütze, H. *Introduction to Information Retrieval.* Cambridge University Press, New York, NY, USA, 2008.
12. Martin, W., Sarro, F., Jia, Y., Zhang, Y., and Harman, M. A survey of app store analysis for software engineering. *IEEE Trans. Software Eng. 43*, 9 (2017), 817–847.
13. Morales-Ramirez, I., Muñante, D., Kifetew, F., Perini, A., Susi, A., and Siena, A. Exploiting user feedback in tool-supported multi-criteria requirements prioritization. In *2017 IEEE 25th International Requirements Engineering Conference (RE)* (Sept 2017), pp. 424–429.

14. Sorbo, A. D., Panichella, S., Alexandru, C. V., Visaggio, C. A., and Canfora, G. Surf: Summarizer of user reviews feedback. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)* (May 2017), pp. 55–58.
15. Traynor, D. How to make product improvements. `https://www.intercom.com/blog/ways-to-improve-a-product/`, Aug. 2018.
16. Vu, P. M., Nguyen, T. T., Pham, H. V., and Nguyen, T. T. Mining user opinions in mobile app reviews: A keyword-based approach (t). In *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)* (Nov 2015), pp. 749–759.
17. Vu, P. M., Pham, H. V., Nguyen, T. T., and Nguyen, T. T. Phrase-based extraction of user opinions in mobile app reviews. In *2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE)* (Sept 2016), pp. 726–731.