

Relating User Feedback and Existing Requirements

Michael Anders*

Heidelberg University, Im Neuenheimer Feld 205, 69190 Heidelberg, Germany

Abstract

[Context and Motivation] Online software feedback sources such as forums, twitter or app stores have opened up new avenues for feedback collection. These large amounts of feedback give developers the opportunity to improve software by addressing users' concerns. To identify improvements from the feedback, developers need to relate it to existing requirements. **[Question/Problem]** Manual analysis of such large amounts of data is very time consuming. Automatic analysis is needed, using natural language processing, machine learning, and deep learning algorithms. However, users and developers have distinct languages due to their different knowledge. As a result the concepts users use in their feedback might not fit the concepts used in the requirements, making direct relation difficult. **[Principal ideas/results]** The proposed solution to the automatic classification problem utilizes a requirements framework in order to classify the user feedback into the requirement categories addressed therein. This is a way of pre-filtering the feedback into requirements categories before classifying which specific requirement it addresses. This offers potential for both, better automatic classification and semi-automatic recommendation of related requirements. **[Contribution]** This paper discusses the research problem of relating feedback statements to specific requirements. It then introduces a potential solution to that problem via a pre-filtering approach. It also discusses the used research method, related work, the research plan and progress so far.

Keywords

User Feedback, Requirements, Natural Language Processing, Automatic Classification

1. Problem

Online user feedback sources such as user forums, tweets or app reviews have allowed developers to collect huge amounts of natural language statements about their software. Analysing this data is advantageous as it can offer insights into users' experiences and requirements [1]. The feedback data however, comes in such large quantities, that analysis is often too time intensive to be done manually. To properly address the issues raised by feedback about existing software, these user statements need to be brought into relation with specific requirements. Thus, there is a need for automation of this process via classification algorithms using natural language processing (NLP), machine learning (ML) and deep learning (DL) algorithms (as defined in [2]).

In: A. Ferrari, B. Penzenstadler, I. Hadar, S. Oyedeji, S. Abualhaija, A. Vogelsang, G. Deshpande, A. Rachmann, J. Gulden, A. Wohlgemuth, A. Hess, S. Fricker, R. Guizzardi, J. Horkoff, A. Perini, A. Susi, O. Karras, A. Moreira, F. Dalpiaz, P. Spoletini, D. Amyot. *Joint Proceedings of REFSQ-2023 Workshops, Doctoral Symposium, Posters & Tools Track, and Journal Early Feedback Track. Co-located with REFSQ 2023. Barcelona, Catalunya, Spain, April 17, 2023.*

*Corresponding author.


✉ michael.anders@informatik.uni-heidelberg.de (M. Anders)

🌐 https://se.ifi.uni-heidelberg.de/people/michael_anders.html/ (M. Anders)

🆔 0000-0001-6932-6146 (M. Anders)



© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

There are numerous classification categories for feedback like user requests, NFRs and issues [3], that are already being researched. The relation of feedback to specific requirements, however requires a more fine grained analysis.

Directly relating feedback to requirements through the concepts within them, for example via similarity classification, is challenging because of the very different knowledge users and developers have [4]. The concepts users express in their feedback might not be present in the requirements written by developers.

A feedback statement taken from the reviews of a hiking application provides an example of this: *"The App often tells me to go right in 100 meters without that actually being possible"*. A human reader is able to infer that this comment addresses some form of navigation problem. Automatic classification could struggle to properly identify the issue because of the vague concepts expressed in the statement. It is unlikely that similar wording is used in the actual requirements specification. For this statement to be processed automatically and then related to a specific requirement a classifier would need to more closely categorize the actions ("the App tells me", "to go right") expressed therein. These actions represent aspects of the software in the form of software interactions ("the App tells me") and user activities ("to go right"). These software aspects can be traced back to requirements categories. Software interactions for example are specified in system functions. This means that direct relation of these user statements to requirements only needs to consider system function or user activity related requirements that deal with outputs to the user and activities that include movement.

As sketched in the example, the relation problem can potentially be reduced by pre-filtering the feedback before analysing the actual concepts therein. This dissertation analyses the potential of this process via classification of feedback statements according to software aspects. In this approach, feedback statements are assigned a requirements category according to their discussed software aspect. Afterwards they can be related to specific requirements according to the prior classification. Software aspects can concern specific system functionalities, interactions with the software or UI elements.

The specific software aspect categories used for the pre-filtering can potentially be defined by individual users themselves. As a proof of concept, in this dissertation the TORE framework [5] is used for the software aspect classification. TORE has been used in various development projects in the past to support requirements engineers in their communication and decision making. TORE consists of decision points (TORE categories) that capture decisions made during software development. For each decision point, a portion of the requirements is specified. The decision points are grouped into three levels of abstraction (TORE level) as shown in figure 1.

2. Research Method

This dissertation's research method follows a version of Wieringa's Design Science Methodology [6]. It is made up of a *design goal* which is achieved by following the *problem investigation*, *treatment design* and *treatment validation* phases.

Design goal: *"Design and evaluate a process to relate user feedback to specific requirements using automatic TORE software aspect classification"*.

Problem investigation: *"Identify classification algorithms in existing fine grained user feed-*

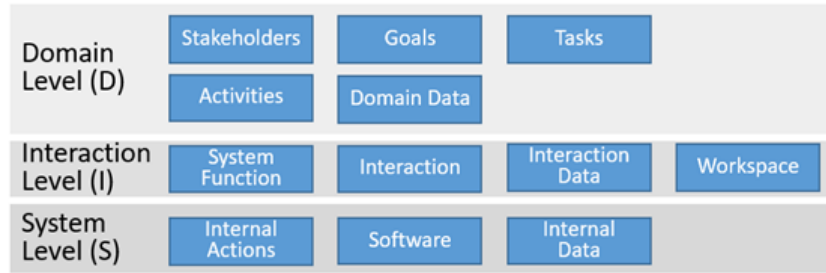


Figure 1: TORE Model For User Feedback Classification

back classification approaches to evaluate their applicability to feedback-requirements relation in a systematic mapping study". The mapping study [7] conducted during this phase, will identify NLP, ML, and DL algorithms which have already been employed to classify user feedback into fine grained classes.

Treatment design: "Automatic TORE software aspect classification for later feedback-requirements relation". A selection of the found algorithms is used to develop new, adapted algorithms for the purpose of TORE software aspect classification and feedback-requirements relation. For the training of classifiers, data from different user feedback sources is collected and adequately labelled to function as training and testing data during development of the algorithms. These sources include online forums, app reviews and online surveys.

Treatment validation: "Evaluation of the algorithms with real user feedback data and existing requirements". The data for this is gathered during the course of a healthy-ageing related project called SmartAge [8]. It will specifically be useful, as the requirements of this project are already specified according to the TORE framework.

3. Proposed Solution

The proposed solution to the problem described in section 1 is to develop a process for the relation of feedback to existing requirements using pre-filtering of the feedback. Thus, this process consists of two steps; namely (1) pre-filtering the feedback according to TORE software aspects discussed therein and (2) relating the feedback to specific requirements relevant to those TORE software aspects.

For the purpose of analysing feedback with the TORE framework, it is simplified from 18 to 12 categories as can be seen in figure 1. The adaptation of TORE for user feedback analysis and the definition for individual categories is further described in [9]. Software aspects (represented by TORE categories) will be analysed in user feedback by encoding natural language data sets. Here, individual text segments, as opposed to whole sentences or complete feedback statements, are assigned a corresponding TORE category as a code. If the requirements of a project have been developed using the TORE framework, these segments can then be traced back to a specific requirements category. This process will be realised by using either a multi-class classification approach or a multi-level, multi-class classification approach [10]. The multi-class approach uses one classifier to assign each text segment one specific TORE category (or no category at

all). The multi-level, multi-class approach first categorizes the feedback segments into one of the three TORE levels (or no category at all) before a second classifier then categorizes the statements into one of the corresponding categories of that level.

Having assigned the text segments a specific TORE category and thus a requirements category, the second step of relating the feedback to a specific requirement will be performed by pre-trained language models like BERT [11]. Depending on the accuracy of the algorithms, a simpler recommender system might also be designed. Then, developers ultimately decide which specific requirement the feedback belongs to, given a set of recommendations.

4. Related Work

In order to automate the classification of user feedback into software aspect categories, a fine grained analysis of the statements is needed. As part of the mapping study, an analysis of the works discussed in different literature reviews in the fields of crowd sourced requirements and user feedback analysis (from review sources such as app stores, Twitter and online forums) [12], [13], [14], [15], [16], and [3] is being conducted. It shows that of 341 papers cited, only 16 introduce fine grained user feedback analysis approaches. Some of these approaches will be briefly discussed in the following paragraph. Ciurumelea et al. [17] introduce a high and low level taxonomy for app review classification with categories such as *compatibility*, *usage*, *resources* etc. They apply a Regression Tree model for the automatic classification. The categories introduced by Guzman et al. [18] aim to classify feedback into a software evolution taxonomy. The categories include classes such as *feature strength* and *usage scenario*. For the classification they apply Naive Bayes, Support Vector Machine (SVM), Logistic Regression and neural networks. Lu et al. [19] aim to classify non-functional requirement categories; namely *reliability*, *usability*, *portability* and *performance*. For this they use an adaptation of Bag-of-Words with a machine learning classifier. McIlroy et al. [20] introduce a complex taxonomy of 14 categories for a multi-label classification of user feedback into categories such as *functional complaint*, *privacy and ethical issue* or *user interface*. They apply SVM, J48, Naive Bayes and Random Forest for the classification problem. Khan et al. [21] extract user rationale from feedback statements by using categories like, *claim-supporting* and *claim-attacking*. They use Naïve Bayes multinomial, Random Forest, SVM and Logistic Regression as classifiers.

It is noteworthy that none of the 341 works specifically target which aspects of a software users are referring to in their feedback. However, these existing approaches still offer valuable insights into the applicability of existing NLP, ML and DL algorithms for the classification problem.

Algorithms for the second step of the process will be identified in a separate literature review of existing pre-trained language models approaches (e.g. BERT [11]). Insights will also be drawn from the papers collected during the mapping study. Dabrowski et al. [22] performed a replication study on approaches which identify feature-related reviews. These could be of interest for the feedback-requirements relation step. However, they found that these approaches achieve lower effectiveness than reported originally. They also found that existing searching tools like Lucene could be useful for similar tasks. These will also be considered going forward.

5. Research Plan & Progress

The current progress and future plans of the dissertation can be summarized into five main milestones: (1) Adapt TORE for feedback classification, (2) Create training and evaluation data sets, (3) Perform a mapping study of fine grained user feedback classification approaches, (4) TORE classification algorithm development and (5) Feedback-requirements relation algorithm development.

(1) The TORE framework chosen for the pre-filtering of user feedback was, as previously mentioned, originally developed solely for requirements engineering. As such, adaptation and simplification of the model was necessary in order to facilitate its use on feedback. These adaptations are described in [9]. Overall, the number of individual categories was reduced from 18 to 12 with some closely related categories being combined. Coding rules for manual category assignment were then developed. This milestone has largely been completed, but the developed framework might require future adaptation.

(2) The development of milestones (4) and (5) will require sufficiently large data sets for training and evaluation of automatic algorithms. For the purpose of milestone (4) multiple data sets are manually labelled according to the previously mentioned rules. Currently, the largest of these data sets consists of 1146 sentences of both user feedback and users' descriptions of a hiking application called Komoot. This data was gathered from an online survey. Another data set of 847 sentences from posts on the online forum Reddit regarding Komoot, Chrome and VLC Video Player has also been created. Future data sets include one based on app store reviews and one consisting of user feedback on health related software during the previously mentioned SmartAge project [8]. The latter will present the main evaluation data set for milestones (4) and (5). This data set creation will be continuously worked on during milestones (3)-(5).

(3) As mentioned in section 4, a mapping study is currently being conducted to gather previous works related to automatic fine grained user feedback analysis. Existing literature reviews were used as a starting point for the study. This was mainly done because of the inconsistent nature of term usage across papers dealing with such fine grained classification, making a search term based approach possibly unreliable. Of the 341 works cited in the reviews, 214 deal with user feedback as a source, 105 of those tackle automatic classification problems. 66 of those are related to software aspect classification of which 51 handle pre-defined classes. Only 16 of those tackle classifications of fine granularity. From these 16 works snowballing will be performed to find additional works. Once the mapping study is completed a selection of NLP, ML, and DL algorithms will be chosen according to predefined criteria such as their precision and recall during the evaluation. Completion of this milestone is planned within the next three months.

(4) With the data sets manually labelled in milestone (2) and the findings of the mapping study, it is possible to implement and evaluate algorithms for the automation of the TORE classification process. In parallel to the mapping study, two different algorithms have already been implemented as a first step towards investigating the potential of automatic TORE classification. A Bi-LSTM based deep learning algorithm (LSTM) based on the work of Li et al. [23] and a machine learning named-entity-recognition algorithm based on the work of Finkel et al. [24] (NER). These were chosen as they offered classification of individual text segments and had previously been used to classify user feedback. Both algorithms were first trained and tested on the Reddit data set and showed similar results. The data set is inherently unbalanced towards

some categories as they are much more prevalent in users' statements. For example, while the category *Interaction* represents 23% of all occurrences, the category *Activity* only represented 3.5% of all occurrences. Categories with a high number of occurrences showed F1 scores as high as 0.71 (LSTM) and 0.82 (NER). Less represented categories had F1 scores ranging as low as 0.15 (LSTM) and 0.31 (NER). With the recent completion of the Komoot data set, the LSTM classifier has been retrained and evaluated on both the Reddit and Komoot data sets (around 2000 sentences). Evaluation shows improvements for the lower F1 scores (lowest 0.33 compared to 0.15 before). However, frequently used categories show no improvement. Retraining of the NER algorithm on the larger data set is planned for the near future. While the automatic classification using the LSTM and NER algorithms showed promising first results, further development of classification algorithms is necessary. Once the mapping study is completed, new algorithms will be developed and tested on the above mentioned data sets. It is difficult to define specific values for precision and recall that are required of these algorithms. A focus will however be put on developing algorithms with high recall, as this provides more use in a recommender system [25]. In addition, methods such as SMOTE [26] will be used to handle the high imbalance in the data sets. Completion of this milestone is planned for the the first half of 2024.

(5) The last milestone concerns early exploration of the possibility of automating the relation of user feedback, previously classified into software aspects, to specific requirements. Language models such as BERT [11] for example will be evaluated for their capability of relating feedback to specific requirements based on linguistic similarity. These evaluations will be performed both with and without previous filtering through TORE classification. This serves the discovery of possible problems with the direct relation of feedback and requirements. It also allows evaluation of whether pre-filtering of feedback statements alleviates these problems. Completion of this milestone is planned for beginning of 2025.

In parallel to all above mentioned milestones an open source software tool called FEED.UVL is developed to provide functionalities for data set generation, manual coding of data sets, as well as classification automation and evaluation. This tool serves as a support for the research by simplifying many manual labour steps. It will also serve to possibly incorporate the relation of feedback to specific requirements into the development process more easily, by functioning as a prototype for an interface between feedback source and requirements documentation.

References

- [1] D. Pagano, W. Maalej, User feedback in the appstore: An empirical study, in: RE Conference, IEEE, 2013, pp. 125–134.
- [2] L. Zhao, W. Alhoshan, A. Ferrari, et al., Natural language processing for requirements engineering: A systematic mapping study, *ACM Comput. Surv.* 54 (2021).
- [3] J. Dąbrowski, E. Letier, A. Perini, A. Susi, Analysing app reviews for software engineering: a systematic literature review, *Empirical Software Engineering* 27 (2022).
- [4] B. Paech, K. Schneider, How do users talk about software? searching for common ground, in: REthics Workshop, IEEE, 2020, pp. 11–14.
- [5] B. Paech, K. Kohler, Task-driven requirements in object-oriented development, *Perspectives on software requirements* (2004) 45–67.

- [6] R. J. Wieringa, Design science methodology for information systems and software engineering, Springer, 2014.
- [7] B. Kitchenham, S. Charters, Guidelines for Performing Systematic Literature Reviews in Software Engineering, Technical Report, Keele University and Durham University, 2007.
- [8] L. Radeck, B. Paech, F. Kramer-Gmeiner, et al., Understanding it-related well-being, aging and health needs of older adults with crowd-re, in: REW, IEEE, 2022, pp. 57–64.
- [9] M. Anders, M. Obaidi, B. Paech, K. Schneider, A study on the mental models of users concerning existing software", in: REFSQ Conference, Springer, 2022, pp. 235–250.
- [10] G. Tsoumakas, I. Katakis, Multi-label classification: An overview, International Journal of Data Warehousing and Mining (IJDWM) 3 (2007) 1–13.
- [11] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).
- [12] C. Wang, M. Daneva, M. van Sinderen, et al., A systematic mapping study on crowdsourced requirements engineering using user feedback, Evolution and Process 31 (2019).
- [13] T. Zhang, L. Ruan, The challenge of data-driven requirements elicitation techniques, master thesis, Blekinge Institute of Technology, 2020.
- [14] R. Santos, E. C. Groen, K. Villela, An overview of user feedback classification approaches., in: REFSQ Workshops, volume 3, Springer, 2019, pp. 357–369.
- [15] S. Lim, A. Henriksson, J. Zdravkovic, Data-driven requirements elicitation: A systematic literature review, SN Computer Science 2 (2021) 1–35.
- [16] J. A. Khan, L. Liu, L. Wen, et al., Crowd intelligence in requirements engineering: Current status and future directions, in: REFSQ Conference, Springer, 2019, pp. 245–261.
- [17] A. Ciurumelea, A. Schaufelbühl, S. Panichella, H. C. Gall, Analyzing reviews and code of mobile apps for better release planning, in: SANER Conference, IEEE, 2017, pp. 91–102.
- [18] E. Guzman, M. El-Haliby, B. Bruegge, Ensemble methods for app review classification: An approach for software evolution (n), in: ASE Conference, IEEE, 2015, pp. 771–776.
- [19] M. Lu, P. Liang, Automatic classification of non-functional requirements from augmented app user reviews, in: EASE Conference, ACM, 2017, pp. 344–353.
- [20] S. McIlroy, N. Ali, H. Khalid, et al., Analyzing and automatically labelling the types of user issues that are raised in mobile app reviews, Empirical SE 21 (2016) 1067–1106.
- [21] J. Ali Khan, L. Liu, L. Wen, Requirements knowledge acquisition from online user forums, Iet Software 14 (2020) 242–253.
- [22] J. Dąbrowski, E. Letier, A. Perini, A. Susi, Mining and searching app reviews for requirements engineering: Evaluation and replication studies, Information Systems (2023).
- [23] N. Li, L. Zheng, Y. Wang, B. Wang, Feature-specific named entity recognition in software development social content, in: SmartIoT Conference, IEEE, 2019, pp. 175–182.
- [24] J. R. Finkel, T. Grenager, C. D. Manning, Incorporating non-local information into information extraction systems by gibbs sampling, in: ACL Conference, Association for Computational Linguistics, 2005, pp. 363–370.
- [25] D. M. Berry, Evaluation of tools for hairy requirements and software engineering tasks, in: RE Workshop, IEEE, 2017, pp. 284–291.
- [26] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, Smote: synthetic minority over-sampling technique, Journal of artificial intelligence research 16 (2002) 321–357.