Research on NLP for RE at the University of Hamburg: a Report

Davide Fucci
HITeC/University of Hamburg
Hamburg, Germany
fucci@informatik.uni-hamburg.de

Christoph Stanik
HITeC/University of Hamburg
Hamburg, Germany
stanik@informatik.uni-hamburg.de

Lloyd Montgomery
HITeC/University of Hamburg
Hamburg, Germany
montgomery@informatik.uni-hamburg.de

Zijad Kurtanović
University of Hamburg
Hamburg, Germany
kurtanovic@informatik.uni-hamburg.de

Timo Johann
University of Hamburg
Hamburg, Germany
johann@informatik.uni-hamburg.de

Walid Maalej
University of Hamburg
Hamburg, Germany
maaleej@informatik.uni-hamburg.de

Abstract

The Mobile Applied Software Technology (MAST) group at the University of Hamburg focuses its research on context-aware adaptive systems and the social side of software engineering. In the context of natural language processing for requirements engineering, the group has mostly focused on mining app stores reviews. Currently, the group is involved in the OpenReq project where natural language processing is being used to recommend requirements from diverse sources (e.g., social media, issue trackers), and to improve the structural quality of existing requirements.

1 Research Group Overview

The MAST group at the University of Hamburg[1] concentrates its research effort on context-aware adaptive systems (CAAS) and social software engineering (SSE) with a particular focus on the mobile service domain.

In particular, *CAAS* observe their users and environments to create a context and automatically adjust and optimize their behavior to it. We are interested in context-aware recommender systems for supporting individuals as well as groups in accessing information, sharing information, and taking collective decisions in software engineering and management scenarios. In our research, we aim to support different stakeholders—for example, requirement engineers in getting recommendations for release planning, or software developer for whom recommender systems can suggest useful documentation.

[1] `https://mast.informatik.uni-hamburg.de/`

SSE pertains the social and human aspects of software engineering as well as the engineering of social software. Within SSE, we recognize the importance of *software socialness*—the systematic involvement of end-users and their communities in the software life cycle, from authoring documentation to even development and integration tasks.

There are several synergies between these research topics investigated by the group and NLP for RE. With the advent of app stores, this is especially the case in mobile services domain. Users produce large, complex, yet information-rich textual data on app stores which can be analyzed, using NLP approaches, to extract requirements. At the same time, recommender systems leverage structured and semi-structured data to support the work of requirements engineers (e.g., requirements elicitation) together with other stakeholders (e.g., requirements negotiation).

2 Past Research on NLP for RE

This section summarizes, in ascending chronological order, the work done by the MAST research group, which focuses on NLP to advance the state-of-the-art in requirements engineering.

Our investigations focus on user-driven requirements engineering. In particular, our NLP studies target user-generated textual content in review systems such as app stores (e.g., Google Play, Apple App Store, Amazon Appstore).

2.1 App Reviews

The data, scripts, and tools for the paper described in this subsection are available at research group website[2].

How Do Users Like this Feature? A Fine Grained Sentiment Analysis of App Reviews
Guzman and Maalej [GM14] use NLP to extract app features from app reviews and analyze the sentiment users show when discussing these features. For the feature extraction, they perform ordinary text preprocessing steps such as stop-word removal, lemmatization, and part-of-speech filtering. After the preprocessing, collocations are used to find app features in the reviews. The collocation process ignores the word order, takes a word window of three words, and is only considered if it appears in at least three reviews. Then collocations, also with similar words, are grouped. Finally, the most frequent collocation within each group was selected as the representative name for that feature.

Moreover, a sentiment analysis was performed using SentiStrength [TBP+10]. This analysis shows how users express their opinion about specific features, or in the whole review. SentiStrength calculates a positive and a negative score for a given text, as both types of expressions can be part of a single text.

As a result of this work, we can extract app features with an average f1-score of 55% and show how these features are perceived (e.g., either positively or negatively) by the users.

On the automatic classification of app reviews
Maalej et al.[MKNS16] paper on the classification of app reviews is an extended version of the previously submitted work of Maalej and Nabil [MN15], which focuses on automatically classifying app reviews as bug report, feature request, user experience, and rating. This paper approaches the classification problem by analyzing which classifier achieves better results and by trying different combinations of machine learning features. In the paper, we consider metadata and NLP based information as machine learning features. For the classification, the results are reported by using only reviews metadata, or only NLP-based machine learning features, or with the combination of both. The data used in the approach are app reviews from the Google Play Store and the Apple App Store. The classification benchmark shows promising results with f1-scores ranging from 89% to 99% for the four classes.

Besides the classification, we developed a prototype of an analytics tool that aggregates the information retrieved from the classification. The tool shows, for example, how the number of bugs evolved, the distribution of the four classes for an app in different app stores, and gives deeper insight by showing concrete reviews in each class. Finally, the tool was evaluated by interviews with nine practitioners, such as software developers and analysts. The interviews show that most practitioners have a need for filtering app reviews that do not contain useful information, such as “great app”, or “I hate it”.

[2] `https://mast.informatik.uni-hamburg.de/app-review-analysis/`

SAFE: A Simple Approach for Feature Extraction from App Descriptions and App Reviews In this paper, Johann et al. [JSM+17] describe a uniform approach (SAFE) that can extract app features from app descriptions, app reviews, and matches both together. SAFE extracts app features without prior machine learning training to analyze what features the app developer provide and to understand how the users talk about it. To extract app features, we use NLP to analyze the structure of sentences. Through qualitative analysis, we found that there are 18 common part-of-speech patterns and four common sentence structures that describe app features. The extraction from app descriptions achieved an average f1-score of 46% while the extraction from reviews had an average f-score of 35%.

After SAFE extracted the app features from the app description and reviews, the final step was to match which features were mentioned in both sources. This information provides insights about the app, such as the identification of (un)popular features, feature requests, and bug reports. The matching was performed in three steps. First SAFE checks if the terms contained in both sources (i.e., app description and app reviews) are identical. Second, we tackle language ambiguity using WordNet to compare the synonyms of each word of the app feature. Third, SAFE extracts the semantic similarity of the app features and calculates the cosine similarity to find a match. The matching procedure achieved and accuracy of 87%.

2.2 Mining User Rationale from Software Reviews

Kurtanović and Maalej [KM17b] introduce user rationale for requirements engineering. Motivated by the amount of data available in social media, user forums, and app stores, software vendors started to give these channels increasing attention. Software vendors want to easily access users' input to make better decisions about software design, its development, and the evolution. This work focuses on the identification of design- and user rationale, which can be valuable for software and requirements engineering. In this work, we found, among others, that rationale, alternatives, criteria, and decisions often co-occur in user comments and that in 21% to 70% of the cases they contain justifications.

In this work, we studied 32,414 reviews for 52 software applications in the Amazon Store. To identify user rationale, we employ a supervised machine learning approach using text, metadata, sentiments, and syntactic features and compare these results between three classification algorithms (Naive Bayes, Support Vector Machine, and Logistic Regression). The classification is tested with different configurations and predicts user rationale at comment and sentence level. The precision and recall for all considered user rationale concepts range between 80%-99% at a comment level and between 69%-98% at a sentence level.

2.3 Other

In this section, we report our experience with topics, other than the application of NLP to app reviews, which we deem interesting for the community.

Toward Data-Driven Requirements Engineering In this paper [MNJR16], we suggest a shift in the requirements engineering community to include user feedback to enable user-centered, data-driven identification, prioritization, and management of software requirements. We show the importance of user feedback and explain what research has achieved so far. These achievements are scoped to the area of analytics of user feedback, such as classifying user feedback into bug reports and feature requests, the classification of stakeholders, and the summarization of user reviews. One primary focus of the paper is to show how these topics are addressed using NLP-based approaches.

Automatically Classifying Functional and Non-Functional Requirements Using Supervised Machine Learning Kurtanović and Maalej [KM17a] use the supervised machine learning classifier Support Vector Machine to classify functional (FR) and non-functional requirements (NFR) automatically using metadata, lexical features, and syntactical features of the requirement text. We show how to classify fine-grained NFRs, such as Usability, Security, Operational, and Performance. From a methodological perspective, one contribution of this paper is the use of under- and over-sampling strategies to handle imbalanced data in the different NFR classes. The classification of FRs and NFRs achieved an f1-score of up to 93%. The classification results of more specific NFRs achieved f1-scores ranging between 51% and 82%.

3 Research Plan on NLP for RE

Currently, the group is involved in the H2020-funded project OpenReq.[3]

The goal of the project is to research, develop, and evaluate intelligent recommendation and decision technologies that will support communities and individual stakeholders in the gathering and management of software requirements.

In particular, OpenReq wants to bridge the gap between the development and usage of software products and services. To that end, the project aim is to take into account the user community as part of the innovation process and continuously observe and involve stakeholders and end users in the decision-making process. OpenReq use cases will cover open-source development, telecommunications, and railways bidding.

In the context of the project, the group will apply NLP to two specific activities related to requirements engineering, i) derive/improve requirements from unstructured text, and ii) improve the quality of existing requirements.

Activity i) is currently under development. It consists of collecting explicit user feedback from public channels, such as social media, reviews system, ticketing systems, and discussion forums, and then aggregating and analyzing this large amount data to facilitate stakeholders understanding of users needs and help them to react quickly.

From an NLP perspective, we are using such data to tackle four tasks:

- provide features, based on statistical language processing (e.g., tf-idf, GloVe), for machine learning classifiers. Here, we want to differentiate between relevant and irrelevant feedback, as well as further categorize relevant one—e.g., understand whether the feedback contains a request for a new feature, a complaint about an existing one, or both
- perform sentiment analysis, to assess, for example, the user base reception of a new feature and allow stakeholders to act accordingly,
- perform summarization and facilitate the access to this significant amount of data to stakeholders and decision makers; in this regard, we are interested in visualization techniques to support this task,
- perform Named-Entity Recognition (NER) and topic recognition to understand what are the specific areas in which the previous tasks can be applied.

The above points are particularly interesting from a research point-of-view, as the language used in these texts is not only English but also Italian. Moreover, since much of the data is collected from channels such as Twitter, the text tends to be short and colloquial.

Activity ii) is currently in a preliminary phase. Here, we will analyze requirement documents—either structured (e.g., user stories) or not (e.g., free-form text). NLP techniques will be used to build a recommender system for improving structural properties of the requirements text.

In particular, we expect to focus on the following tasks:

- Word sense disambiguation and coreference resolution to identify ambiguous passages in the requirement text and suggest corrective actions.
- Chunking and relationship extraction to assess (and eventually correct) conformance to templates, such as user stories.
- Semantic role labeling and textual entailment to assess the completeness of a requirement text concerning several concerns (e.g., risk).

As these documents contain domain-specific knowledge, we are investigating the possibility to support the NLP approaches with ontologies and glossaries.

Acknowledgment

We would like to acknowledge the H2020 EU research project OpenReq (ID 732463).

[3] http://openreq.eu/

References

[GM14] Emitza Guzman and Walid Maalej. How do users like this feature? a fine grained sentiment analysis of app reviews. In *Requirements Engineering Conference (RE), 2014 IEEE 22nd International*, pages 153–162. IEEE, 2014.

[JSM+17] Timo Johann, Christoph Stanik, Walid Maalej, et al. Safe: A simple approach for feature extraction from app descriptions and app reviews. In *Requirements Engineering Conference (RE), 2017 IEEE 25th International*, pages 21–30. IEEE, 2017.

[KM17a] Zijad Kurtanović and Walid Maalej. Automatically classifying functional and non-functional requirements using supervised machine learning. In *Requirements Engineering Conference (RE), 2017 IEEE 25th International*, pages 490–495. IEEE, 2017.

[KM17b] Zijad Kurtanović and Walid Maalej. Mining user rationale from software reviews. In *Requirements Engineering Conference (RE), 2017 IEEE 25th International*, pages 61–70. IEEE, 2017.

[MKNS16] Walid Maalej, Zijad Kurtanović, Hadeer Nabil, and Christoph Stanik. On the automatic classification of app reviews. *Requirements Engineering*, 21(3):311–331, 2016.

[MN15] Walid Maalej and Hadeer Nabil. Bug report, feature request, or simply praise? on automatically classifying app reviews. In *Requirements Engineering Conference (RE), 2015 IEEE 23rd International*, pages 116–125. IEEE, 2015.

[MNJR16] Walid Maalej, Maleknaz Nayebi, Timo Johann, and Guenther Ruhe. Toward data-driven requirements engineering. *IEEE Software*, 33(1):48–54, 2016.

[TBP+10] Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. Sentiment strength detection in short informal text. *Journal of the Association for Information Science and Technology*, 61(12):2544–2558, 2010.