Requirements-Collector: Automating Requirements Specification from Elicitation Sessions and User Feedback

Sebastiano Panichella and Marcela Ruiz
Zurich University of Applied Sciences - Winterthur, Switzerland
Institute of Applied Information Technologies (InIT)
Email: {sebastiano.panichella, marcela.ruiz}@zhaw.ch

***Abstract*—In the context of digital transformation, speeding up the time-to-market of high-quality software products is a big challenge. Software quality correlates with the success of requirements engineering (RE) sessions and the ability to collect feedback from end-users in an efficient, dynamic way. Thus, software analysts are tasked to collect all relevant material of RE sessions and user feedback, usually specified on written notes, flip charts, pictures, and user reviews. Afterward comprehensible requirements need to be specified for software implementation and testing. These activities are mostly performed manually, which causes process delays, with a negative effect on software quality attributes such as reliability, usability, comprehensibility. This paper presents *Requirements-Collector*, a tool for automating the tasks of requirements specification and user feedback analysis. Our tool involves machine learning (ML) and deep learning (DL) computational mechanisms enabling the automated classification of requirements discussed in RE meetings (stored in the form of audio recordings) and textual feedback in the form of user reviews. We use such techniques as they demonstrated to be quite effective in text classification problems. We argue that *Requirements-Collector* has the potential to renovate the role of software analysts, which can experience a substantial reduction of manual tasks, more efficient communication, dedication to more analytical tasks, and assurance of software quality from conception phases. The results of this work have shown that our tool is able to classify RE specifications and user review feedback with reliable accuracy.**

***Keywords*-Requirement Analysis, User Stories, User Reviews Feedback.**

I. Introduction

Software quality maintenance correlates with the success of requirements engineering (RE) sessions [6], and the ability to efficiently analyse and select useful end-user feedback [2]. Software requirements are collected during RE sessions in the shape of pictures, flip chart notes, documentation etc. This information is digitalised in order to specify a set of comprehensible user stories to be used during development phases [7]. Digitalisation of discussed requirements demands extra effort that has to be undertaken by software analysts [8]. Once software is deployed, development teams spend considerable effort in collecting and exploiting user feedback to improve user satisfaction and needs. Previous work [4], [5] has shown that approximately one third of the information contained in user reviews is helpful for developers, and these reviews tend to consist in *unstructured* text that is difficult to parse and analyze. All this complexity is magnified by contemporary software development, which take place in the different geographical locations, posing big challenges for collaborative requirements engineering [9].

This demo paper presents *Requirements-Collector*, a machine and deep learning based tool for automating the tasks of requirements specification. In the next sections we discuss our tool in the context of the state-of-the-art, describing the main components composing *Requirements-Collector*. Finally, we summarize our results and directions for future work.

II. Related work

Various approaches have been proposed to extract requirements specifications from different written origins [10], [11], with approaches able to automatically classify user reviews information [4]. This work advances the mentioned state-of-the-art by providing an unified solution enabling the automated classification of requirements discussed in RE meetings (stored in form of audio recordings) and textual user review feedback [1], [2], [3]. To the best of our knowledge, this is *the first research tool that supports the combined software requirements elicitation from RE sessions and end-users feedback.*

III. Designing a Requirements Collector Tool

The main objective of our research is *to reduce the time-to-market of software products by automating the task of requirements specification and user feedback collection*. We designed our tool around the vision (see Figure 1) of a requirements engineering room where participants discuss requirements that are automatically specified in the shape of user stories, and user reviews are automatically classified for analysis. In this room, we incorporate virtual reality tools like double robots for embodiment of remote participants, and interactive boards with collaborative tools as they have demonstrated to facilitate access and real-time edition of discussed requirements [8]. User stories and classified user reviews will be generated on the fly during the session, and virtual reality systems will allow efficient communication. To support an automatic requirements collector tool, we have implemented transcript (generated from the requirements engineering session) and user reviews (obtained from end-user reviewing software in practice) classifiers which are presented below.

Transcript & User Reviews classifiers. *Requirement-Collector* is composed by two main components:

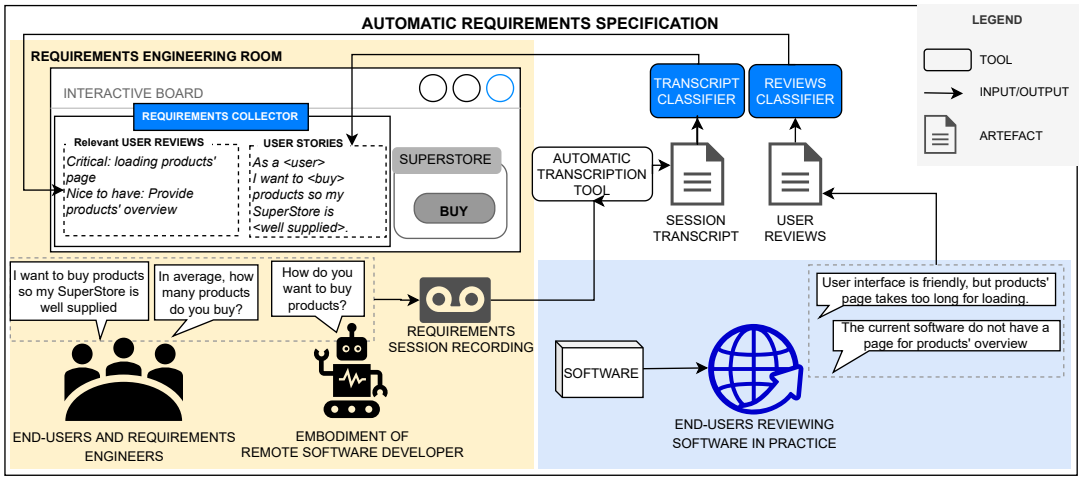


Fig. 1. The requirements collector tool in an automatic requirements specification environment

`Requirement-Collector-DL-Component` and `Requirement-Collector-ML-Component`. To classify transcripts and user reviews these two components leverage deep learning (unsupervised) and machine learning (supervised) strategies, respectively. To train the DL and ML models, these two components extract and pre-process text data from the original datasets of previous work concerning requirements analysis of transcripts [1] and user reviews feedback [2], [3]. Hence, they classify functional and non-functional requirements discussed during requirements elicitation sessions by leveraging machine learning or deep learning strategies. In classifying user reviews, *Requirement-Collector* leverages mainly ML strategies, we plan to experiment with DL strategies for future work.

Evaluation. We used *Requirement-Collector* to experiment with over ten ML (supervised) models (*J48, PART, NaiveBayes, IBk, OneR, SMO, Logistic, AdaBoostM1, LogitBoost, DecisionStump, LinearRegression, RegressionByDiscretization*) and a DL method, to evaluate its accuracy. Our preliminary results suggests that for classifying transcripts is more appropriate to leverage DL strategies (we achieved an average F-measure of 33% with DL and average 5% with ML models). Moreover, when the task concern the classification of user feedback from user reviews, the best performing ML models (i.e., the SMO) achieves an F-Measure of 77%.

Appendix and Annex. We provide Github-based versions as extended Appendix [1], [2], with video demonstrations of the two implemented Requirement-Collector components.

IV. Conclusions

This research paper introduces Requirements-collector, a tool that exploits machine learning and deep learning for automatic classification of requirements from elicitation sessions and user feedback. Our preliminary results highlight its accuracy for requirements extraction. Results provide hints to asses how ideal ML and DL models are to achieve high accuracy. For future work, we focus on leveraging extracted requirements for prioritization of maintenance activities (generate traceability links among requirements coming from elicitation sessions and user reviews) and improve the effectiveness of testing tasks. For classifying requirements from elicitation sessions, we plan to investigate to what extent a structured session might lead to better quality and completeness of extracted user stories. The final output of our requirements collector need to be validated by means of use cases.

[1] https://github.com/lmruizcar/Requirements-Collector-DL-Component
[2] https://github.com/spanichella/Requirement-Collector-ML-Component

Acknowledgment

This research project is supported by ZHAW Digital and the Digitalisation Initiative of Zürich Universities DIZH.

References

[1] M. Ruiz and B. Hasselmann, "Can We Design Software as We Talk: A research idea, International working conference on Exploring Modeling Methods for Systems Analysis and Development, 2020
[2] S. Panichella, A. Di Sorbo, E. Guzman, C. A. Visaggio, G. Canfora, and H. C. Gall, "Ardoc: App reviews development oriented classifier," International Symposium Foundations of Software Engineering, 2016
[3] S. Panichella, A. Di Sorbo, E. Guzman, C. A. Visaggio, G. Canfora, and H. C. Gall, "How can I improve my app? Classifying user reviews for software maintenance and evolution," International Conference on Software Maintenance and Evolution, 2015
[4] N. Chen, J. Lin, S. C. H. Hoi, X. Xiao, and B. Zhang,"Ar-miner: Mining informative reviews for developers from mobile app marketplace," International Conference on Software Engineering, 2014
[5] D. Pagano, and W. Maalej *User Feedback in the AppStore: An Empirical Study*. International Requirements Engineering Conference (RE), 2013.
[6] Mund, J., Femmer, H., Mendez, D., and Eckhardt, J.: "Does Quality of Requirements Specifications matter? Combined Results of Two Empirical Studies," (2017). [Online]. Available: https://arxiv.org/pdf/1702.07656.pdf
[7] Dalpiaz F. and Brinkkemper, S. "Agile Requirements Engineering with User Stories". International Requirements Engineering Conference, 2018
[8] Wüest, D., and Seyff, N., and Glinz,. "FlexiSketch: a lightweight sketching and metamodeling approach for end-users. In Software & Systems Modeling, no 2, vol. 18 pages 1513–1541, (2017).
[9] Damian, D., Zowghi, D. "Requirements Engineering challenges in multi-site software development organizations". Requirements Engineering Journal. Vol 8. Pages 149-160, (2003)
[10] Rodeghero, P., Jiang, S., Armaly, A., und McMillan, C. "Detecting User Story Information in Developer-Client Conversations to Generate Extractive Summaries", International Conference on Software Engineering, 2017
[11] Abad, Z. S. H., Gervasi, V., Zowghi, D., und Barker, K. "ELICA: An Automated Tool for Dynamic Extraction of Requirements Relevant Information", in International Workshop on Artificial Intelligence for Requirements Engineering, 2018

V. Annex

This appendix describes how this demo paper will be presented at the RE20 conference in Zürich.

VI. Interaction potential

For the RE20 Posters and Tool Demos session, the objective is to showcase live demos and present how the requirements collector works. We will go into the details describing the two main tool components: `Requirement-Collector-DL-Component` and `Requirement-Collector-ML-Component`. Both components are available on our GitHub repositories [3] [4]. Important, we experimented the usage of the two components on the dataset available from previous work, concerning the classification of requirements discussed in RE meetings (stored in form of audio recordings) and textual user review feedback [1], [2], [3].

Figure 2 presents our GitHub repository for the Requirements-Collector-DL-Component. It describes the component's project structure, technical requirements, and how to get started with the deep learning component. A screenshot with the expected output from running the code has been also made available for verification (see Figure 3).

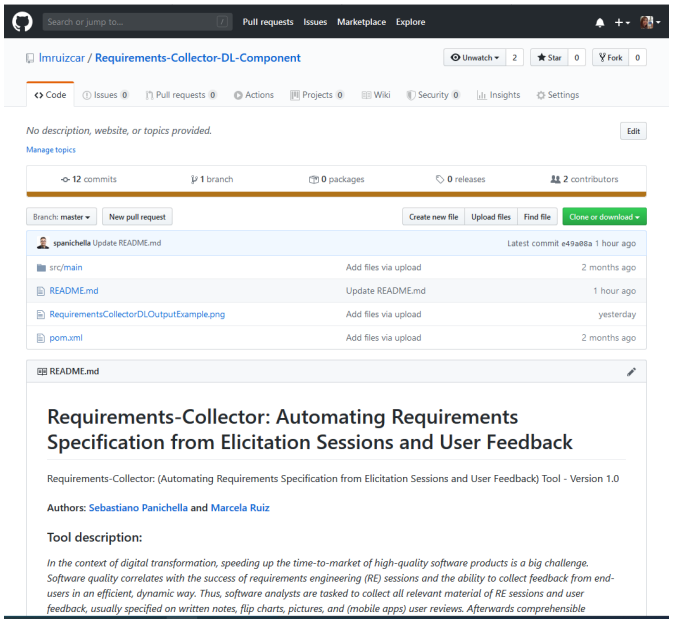


Fig. 2. Requirements-Collector-DL-Component on GitHub

Posters and tool demos participants will experiment how available requirements session transcripts are processed by using machine learning. In addition, the code and technicalities of the tool will be explored. During the session, we will answer questions regarding tool's architecture and potential improvements. Since the source code is available on GitHub,

[3]https://github.com/lmruizcar/Requirements-Collector-DL-Component
[4]https://github.com/spanichella/Requirement-Collector-ML-Component

```
Examples labeled as 0 classified by model as 0: 1 times
Examples labeled as 0 classified by model as 2: 4 times
Examples labeled as 1 classified by model as 2: 5 times
Examples labeled as 2 classified by model as 0: 1 times
Examples labeled as 2 classified by model as 2: 4 times

Warning: 1 class was never predicted by the model and was excluded from average precision
Classes excluded from average precision: [1]

========================Scores========================================
 # of classes:    3
 Accuracy:        0.3333
 Precision:       0.4038       (1 class excluded from average)
 Recall:          0.3333
 F1 Score:        0.3651       (1 class excluded from average)
Precision, recall & F1: macro-averaged (equally weighted avg. of 3 classes)
======================================================================
```

Fig. 3. Output from executing the Requirements-Collector-DL-Component

participants are free to download the code and execute the components on their own IDEs.

Figure 4 presents our GitHub repository for the Requirements-Collector-ML-Component. The page describes how to get started with the machine learning component, necessary requirements and packages that need to be installed in advance to be able to execute the tool. Participants during the RE session will experiment first hand with different machine learning algorithms we have implemented for requirements and reviews classification. For this component, we also provide two demonstration examples to run the tool. We provide a configuration files example and also a command line statements (see Figure 5). It is important to highlight that provided demonstration examples are just available for Unix/Linux/MacOS operative systems.

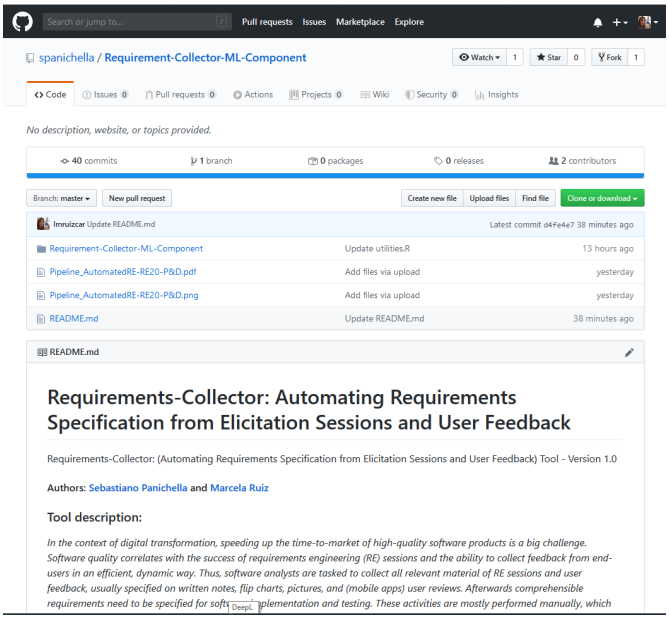
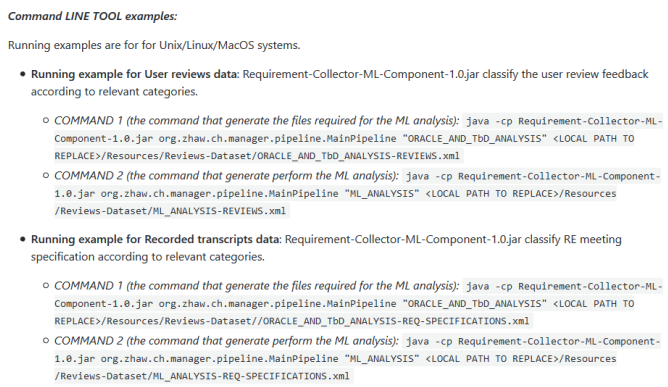


Fig. 4. Requirements-Collector-ML-Component on GitHub

During the posters and tool demos at RE20, we will run a parallel execution of the two main components of the requirements collector. As a use case, we plan to analyse the same requirements transcripts by using our implemented machine and deep learning algorithms. This would allow us to discuss with the participants the different advantages and



Command LINE TOOL examples:

Running examples are for for Unix/Linux/MacOS systems.

- **Running example for User reviews data**: Requirement-Collector-ML-Component-1.0.jar classify the user review feedback according to relevant categories.
  - *COMMAND 1 (the command that generate the files required for the ML analysis):* `java -cp Requirement-Collector-ML-Component-1.0.jar org.zhaw.ch.manager.pipeline.MainPipeline "ORACLE_AND_TbD_ANALYSIS" <LOCAL PATH TO REPLACE>/Resources/Reviews-Dataset/ORACLE_AND_TbD_ANALYSIS-REVIEWS.xml`
  - *COMMAND 2 (the command that generate perform the ML analysis):* `java -cp Requirement-Collector-ML-Component-1.0.jar org.zhaw.ch.manager.pipeline.MainPipeline "ML_ANALYSIS" <LOCAL PATH TO REPLACE>/Resources /Reviews-Dataset/ML_ANALYSIS-REVIEWS.xml`
- **Running example for Recorded transcripts data**: Requirement-Collector-ML-Component-1.0.jar classify RE meeting specification according to relevant categories.
  - *COMMAND 1 (the command that generate the files required for the ML analysis):* `java -cp Requirement-Collector-ML-Component-1.0.jar org.zhaw.ch.manager.pipeline.MainPipeline "ORACLE_AND_TbD_ANALYSIS" <LOCAL PATH TO REPLACE>/Resources/Reviews-Dataset//ORACLE_AND_TbD_ANALYSIS-REQ-SPECIFICATIONS.xml`
  - *COMMAND 2 (the command that generate perform the ML analysis):* `java -cp Requirement-Collector-ML-Component-1.0.jar org.zhaw.ch.manager.pipeline.MainPipeline "ML_ANALYSIS" <LOCAL PATH TO REPLACE>/Resources /Reviews-Dataset/ML_ANALYSIS-REQ-SPECIFICATIONS.xml`

Fig. 5. Requirements-Collector-ML-Component on GitHub

TABLE I
EXAMPLES OF SENTENCES CLASSIFIED BY THE
Requirement-Collector-ML-Component

Sentence	Category
They just need to update the layout I fill like everything is hidden I want a better task bar.	feature request
Please restore a way to open pin links in external browser or let us save photos.	feature request
App crashes when new power up notice pops up.	problem discovery
Please fix the syncing issues with the iPad app.	problem discovery
It's already possible to rearrange boards why not the pins on a single board?	Information seeking
Overall it is fun and provides a lot of good info.	Information giving
This app runs so smoothly and I rarely have issues with it anymore.	Information giving

disadvantages of both techniques. Since we have implemented different machine learning algorithms for supporting user reviews classification, this give us further room to explore how we can find a good combination of machine learning algorithms to reach optimal results. We will discuss how the use of machine learning and deep learning can be exploited for the use case we bring to the session.

Examples. Table I shows examples of sentences and the related user reviews classified by *Requirement-Collector-ML-Component*, according to maintenance and evolution tasks [3].

We expect that this experience at the posters and tool demos session at RE20 will be very fruitful for the participants and us as researchers. Since the requirements collector tool would greatly help requirements engineers in the future, we find it appropriate to be able to share this tool with the community. We expect to have very interesting discussions that give us further hints for the evolution of our current progress.