TraceLab Components for Generating Extractive Summaries of User Stories

Rrezarta Krasniqi, Siyuan Jiang, and Collin McMillan
Dept. of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN, USA
{rkrasniq, sjiang1, cmc}@nd.edu

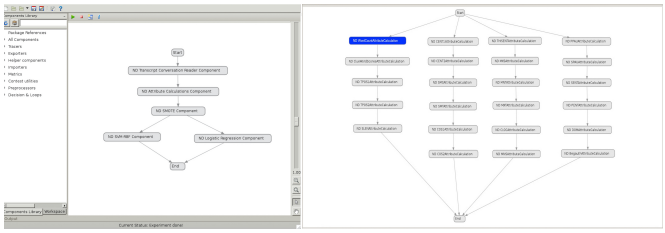
***Abstract*—This artifact is a reproducibility package for experiments in user stories summarization. We implemented and packaged the artifact as a set of reusable TraceLab components. The existing implementation of the artifact was relatively difficult to use because it required the user to coordinate several different programming languages and dependencies. This artifact, available via our online appendix, provides the components, a detailed tutorial with screenshots that show exactly where to click and what to enter, and an example virtual machine image.**

I. Introduction / Background

User stories are short descriptions of software features told from the user perspective. In general, a user story consists of a role, function, and rationale, in the format "As a <role>, I want to <function>, so that I can <rationale>." In work published at ICSE 2017, we presented a technique that aims to generate user stories automatically from recorded transcripts of conversations between customers and developers [2]. The input to our approach is a transcript of a customer-developer conversation, and the output is a list of "speech turns" in the conversation that include role, function, and rationale data (a speech turn is a section of speech by one member of the conversation, during which others are quiet). The automatic generation of extractive summaries for user stories is a new research area, and working implementations are rare. This artifact is a set of reproducible TraceLab components that implement our technique. The benefit to using TraceLab components is that our approach is not only reusable but relatively easy to modify, so that future researchers can build improvements and test them against our approach.

II. The Artifact

In this artifact we include two "`teml`" files and 28 "components." The `teml` file describes an experiment in the TraceLab workbench[1]. This experiment provides a prepackaged working implementation of a research artifact and describes the input/output to the artifact. The implementation is composed from a set of executable components and decision nodes all of which are managed by the *teml* file. In Tracelab, a user can retrieve an existing experiment or create one from scratch. A new experiment can be created in three steps: (1) load a `teml` file, (2) configure the input/output parameters of each component and (3) execute the `teml` file. The results are then displayed in the Tracelab's workspace. The teml file that we built, implements Rodeghero et al. [2] extractive technique. We also include an alternative `teml` file for the "attribute components." The attribute components represent the details of developer/customer conversations, such as: their turns, the duration of conversation including structure and lexical information. In the first `teml` file, "*AttributeCalculations*" component considers all attributes *combined*. The second `teml` file, is more flexible in that, it gives the researchers the ability to pick attributes. We did so to accomodate custom researcher requirements. While the existing implementation of the Rodeghero et al. [2] is available, it is relatively difficult to use because the user has to refer to various languages and manage their dependencies. We improve on this approach by implementing our artifact on one language (Java) and managing its dependencies through TraceLab. The figures below, show the experiment and the collocated 28 components in TraceLab:



On the left are the five components of the experiment [2]:
Transc. Conv.Reader. loads the transcripts into the database.
Attribute Calc. calculates the attributes of each turn in a conv.
SMOTE detemines the path to either of the two training models.
SVM-RBF trains *LIB-SVM* model.
Logistic Regression trains *LIB-LINEAR* model.

On the right are the *23* components, the `teml` file extension of *AttributeCalculations* component.

The experiment, components, detailed tutorial with screenshots, and an example virtual machine image are available via:
http://www3.nd.edu/~rkrasniq/tracelab/icsme17/

Acknowledgment: This work is supported in part by the NSF CCF-1452959 and CNS-1510329 grants. Any opinions, findings, and conclusions expressed herein are the authors and do not necessarily reflect those of the sponsors.

References

[1] J. Cleland-Huang. Tracelab. http://www.coest.org.
[2] P. Rodeghero, S. Jiang, A. Armaly, and C. McMillan. Detecting user story information in developer-client conversations to generate extractive summaries. In *Proc. of the 39th ACM/IEEE International Conference on Software Eng*, volume 1, Buenos Aires, Argentina, May 20-28 2017.