Predicting How to Test Requirements: An Automated Approach

Jonas Winkler[1] Jannis Grönberg[2] Andreas Vogelsang[3]

Abstract: An important task in requirements engineering is to identify and determine how to verify a requirement (e.g., by manual review, testing, or simulation; also called *potential verification method*). This information is required to effectively create test cases and verification plans for requirements. In this paper, we propose an automatic approach to classify natural language requirements with respect to their potential verification methods (PVM). Our approach uses a convolutional neural network architecture to implement a multiclass and multilabel classifier that assigns probabilities to a predefined set of six possible verification methods, which we derived from an industrial guideline. Additionally, we implemented a backtracing approach to analyze and visualize the reasons for the network's decisions. In a 10-fold cross validation on a set of about 27,000 industrial requirements, our approach achieved a macro averaged F_1 score of 0.79 across all labels. The results show that our approach might help to increase the quality of requirements specifications with respect to the PVM attribute and guide engineers in effectively deriving test cases and verification plans.

Keywords: Requirements Engineering, Requirements Validation, Test Engineering, Machine Learning, Natural Language Processing, Neural Networks

1 Introduction

Verifiability is a quality characteristic for requirements that is mentioned in many normative quality standards such as ISO 29148 and the IREB. One of our industry partners has introduced an explicit requirements attribute called *Potential Verification Method (PVM)* that specifies in which ways a requirement must be verified. Possible PVMs are Review, Simulation, Formal Verification, Process Audit, System Test and Production Control. Setting values for this attribute is a manual, time-consuming, and error-prone task.

We propose an automatic approach to classify natural language requirements with respect to their potential verification methods [WGV19]. Additionally we visualize the importance of parts of the input sentence for the classification decision. We conclude that our automated approach helps to increase the quality of requirements by detecting misclassified PVM attributes or automatically generating classification proposals for unlabeled requirements. In this paper, we present a brief overview of the technique and its applications.

[1] Technische Universität Berlin, DCAITI, jonas.winkler@tu-berlin.de
[2] Technische Universität Berlin, DCAITI, jannis.r.groenberg@campus.tu-berlin.de
[3] Technische Universität Berlin, DCAITI, andreas.vogelsang@tu-berlin.de

doi:10.18420/SE2020_43

2 Classifying requirements with respect to the Potential Verification Method

Our classifier takes a requirement and uses a convolutional neural network (CNN) to assign labels to it, which represent different PVM values. First an input sentence is transformed into vector representation (i.e. word embedding). The first layer in the CNN applies a set of filters by moving them as a sliding window over the sentence vector, producing single values at each position (i.e. convolution). The most important features are concatenated and form a feature vector that is connected to the output layer from which a probability between 0 and 1 is derived for each of the six corresponding possible PVM values.

Classifier	Accuracy	Perfect Match Ratio	Macro-F_1	Micro-F_1
ZeroR baseline	0.8477	0.8293	0.1553	0.8323
CNN	**0.9310**	0.9115	**0.7904**	**0.9368**
Example Sentence (System Test)	The actuators and switches must be activated separately within the control unit.			

Tab. 1: Results for the CNN based PVM classifier

We compared the classifier against a ZeroR baseline. Since 84% of the requirements in our dataset contain the class *System Test*, the ZeroR baseline has high values for *Accuracy*, *Perfect Match Ratio* and *Micro-F1*. Table 1 reveals that our CNN-based classifier performs substantially better than the baseline. We traced back the probabilities in our CNN and derived important key phrases for each class. The highlighting for the class *System Test* in the table shows individual words that are especially important for the classification process of the CNN in this particular sentence.

3 Applications & Conclusions

Our classifier can be used for detecting misclassified PVM attributes or automatically generating classification proposals for unlabeled requirements. As shown in previous studies [WV18], integrating a classifier into a tool may provide certain benefits such as shorter review time and increased number of errors fixed.

Literaturverzeichnis

[WGV19] Winkler, J. P.; Grönberg, J.; Vogelsang, A.: Predicting How to Test Requirements: An Automated Approach. In: 2019 IEEE 27th International Requirements Engineering Conference (RE). S. 120–130, Sep. 2019.

[WV18] Winkler, Jonas P.; Vogelsang, Andreas: Using Tools to Assist Identification of Non-Requirements in Requirements Specifications - A Controlled Experiment. In: Requirements Engineering: Foundation for Software Quality (REFSQ). 2018.