2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER) | 978-1-7281-9630-5/20/$31.00 ©2021 IEEE | DOI: 10.1109/SANER50967.2021.00076

Will Data Influence the Experiment Results?: A Replication Study of Automatic Identification of Decisions

Liming Fu, Peng Liang*, Xueying Li
School of Computer Science
Wuhan University, Wuhan, China
{limingfu, liangp, xueyingli}@whu.edu.cn

Chen Yang
IBO Technology (Shenzhen) Co., Ltd.
Shenzhen, China
c.yang@ibotech.com.cn

***Abstract*—Decisions are an important type of artifacts in software development and maintenance, while decisions are not well-documented in projects due to limited human resources and budget. To this end, many studies focus on using automatic approaches to identify decisions from textual artifacts, e.g., mailing lists, issue tracking systems. In this paper, we present a replication study of our previous work (EASE2020), which conducted experiments to automatically identify decisions from the Hibernate developer mailing list. In addition, we utilized different datasets in the experiment with the aim of exploring the impact of the proprieties of dataset (i.e., the quality of positive samples, different negative samples in the dataset, and the size of the dataset) on classification results of decisions. The results show that (1) improving the quality of positive samples in the dataset can decently improve the classification results; (2) different negative samples in the dataset have an impact on the classification results; and (3) before the dataset size reaches 1200, increasing the size will improve the classification results.**

***Index Terms*—Decision, Automatic Identification, Dataset, Hibernate, Replication Study**

I. Introduction

Decision refers to a choice made after thinking or discussing about what is the best thing to do. Decisions run through the whole software development life cycle [1] and understanding these decisions is essential for development and maintenance [2]. Within the evolution process, the past decisions may be challenged [3]. They need to be reviewed, adapted, or even withdrawn since the system shall be kept aligned with the changing context. Hence, existing decisions should be captured and documented to be reconsidered. On the one hand, it helps stakeholders review, analyze, improve, and reuse decisions that have been made, and guides them while making new decisions in a similar context [4]. On the other hand, it encourages stakeholders to organize development knowledge and reduce its vaporization [5]. While in practice decisions are not well-documented due to limited human resources and budget [6], therefore, decisions are a typical type of vaporized knowledge in development [2], which leads to e.g., architecture drift and erosion [7].

* Corresponding author.

This work was funded by the National Key R&D Program of China with Grant No. 2018YFB1402800 and IBO Technology (Shenzhen) Co., Ltd.

Many researchers focused on identifying decisions from available history artifacts, e.g., mailing list, issue tracker, version control repository, etc. For instance, Shahbazian *et al.* [2] developed an automatic technique RecovAr to recover architectural design decisions from issue trackers. Bhat *et al.* [4] also proposed a supervised Machine Learning (ML) approach to detect and classify design decisions from issue tracking systems. In our previous work (EASE2020) [8], we experimented with 160 different configurations regarding text preprocessing, feature extraction techniques, and ML algorithms to automatically identify decisions from mailing lists.

While many researches focuses on automatic identification of decisions, limited work was conducted to investigate the influence of the dataset on the identification results. To this end, we conducted a replication study of our previous work [8] to understand the impact of the proprieties of dataset. There are three motivations: (1) explore the impact of the quality of positive samples in the dataset on the classification results; (2) study the influence of different negative samples in the dataset on the classification results; and (3) research the impact of the size of the dataset on the classification results. Based on the EASE2020 dataset in our previous work [8] that contains 650 *decision* sentences and 650 *non-decision* sentences, we constructed two new datasets [9] (i.e., a new *decision* dataset that contains 844 relabelled *decision* sentences and a new *non-decision* dataset that contains 750 *assumption* sentences extracted by Li *et al.* [10]). We took the aforementioned *decisions* as the positive samples and *non-decisions* as the negative samples, and then conducted experiments to identify decisions using the best approach in our previous work [8]. The experiment results show that (1) improving the quality of positive samples in the dataset can decently improve the classification results; (2) different negative samples in the dataset have an impact on the classification results; and (3) before the dataset size reaches 1200, increasing the size will improve the classification results.

The contribution of this work is twofold: (1) a dataset of decisions in software development, which contains *decision* sentences and *non-decision* sentences, and (2) a first study focusing on exploring the impact of properties of dataset on

978-1-7281-9630-5/21/$31.00 ©2021 IEEE
DOI 10.1109/SANER50967.2021.00076

the results of automatic identification of decisions.

II. Research Design

A. Research Questions

RQ1. Does the quality of the positive samples in the dataset impact the classification results?

RQ2. Do the negative samples in the dataset impact the classification results?

RQ3. Does the size of the dataset impact the classification results?

B. Automatic Identification Approach

In our EASE2020 work, we experimented with 160 configurations regarding text preprocessing, feature extraction techniques, and ML algorithms when automatically identifying decisions [8]. Table I provides an overview of the 160 configurations. The EASE2020 results show that the SVM algorithm with Including Stop Words, No Stemming and Lemmatization, Filtering Out Sentences by Length, and BoW achieves the best performance. In this replication experiment, since we focused on the impact of dataset proprieties on the experiment results rather than different classification approaches, we conducted the experiment with the best identification approach (i.e., the aforementioned best configuration).

TABLE I: The Methods, Techniques, and Algorithms Used for Decisions Identification

	Decisions Identification Methods, Techniques, and Algorithms
Text preprocessing methods	Include Stop Words vs. Remove Stop Words
	Stemming and Lemmatization vs. No Stemming and Lemmatization
	Filtering Out Sentences by Length vs. No Filtering Out Sentences by Length
Feature extraction techniques	BoW vs. TF-IDF vs. CBOW vs. Skip-gram
ML classification algorithms	NB vs. LR vs. DT vs. SVM vs. RF

III. Experiment

A. Data Collection

In our recent study on decisions [1], we manually labelled and analyzed 9006 posts from the Hibernate developer mailing list, and a total of 980 *decision* sentences were extracted from these posts. We then randomly selected 650 *decision* sentences from the 980 *decision* sentences and also labelled 650 *non-decision* sentences to conduct experiments of decisions identification [8]. In this replication study, we rechecked the aforementioned 980 decisions with three researchers, and found that they include debatable and mislabeled decisions. To improve the quality of *decision* sentences in our dataset, we relabelled the 980 decision sentences. Finally, we got a relabelled dataset that contains 844 *decision* sentences. In addition, to explore the impact of different negative samples (i.e., *non-decision* sentences) in dataset on the experiment results, we used 750 *assumption* sentences from [10] as part of negative samples in our dataset. Assumptions, as another type of artifacts, are highly related to but different from decisions. Therefore, we used the 750 *assumption* sentences as part of the negative samples in our dataset. Table II provides an overview of the experiment dataset, which has been provided at [9].

TABLE II: Different Datasets Used for Decisions Identification

Dataset	**Data Description**
Decisions from EASE2020	650 decisions from EASE2020
Decisions from Relabelled	844 relabelled decisions in this work
Non-Decisions from EASE2020	650 non-decisions from EASE2020
Non-Decisions from Assumptions	750 assumptions from [10]

B. Text Preprocessing

As elaborated in Section II-B, we conducted this replication experiment with the best approach in our EASE2020 work [8], and the text preprocessing process is composed of two steps:

1) Eliminating Useless Textual Characters: Due to the fact that Hibernate developers may use natural language to communicate with other developers through mails, there may be a lot of useless characters in the decision sentences, such as ":)" and "*". Therefore, we eliminated these redundant characters and only retained valid terms.

2) Filtering Out Sentences by Length: Li *et al.* pointed out that sentences that contain less than four words have limited information, and they can be considered as invalid features [10]. Therefore, we set a threshold to filter out the sentences with less than four words.

C. Feature Extraction and Classifier Training

We conducted a vectorization process and transformed *decisions* and *non-decisions* into vectors that could represent their features through feature extraction techniques. As illustrated in Section II-B, BoW is exploited to extract textual features in this experiment. BoW builds a vocabulary composed of all unique terms in the sentences of our dataset and then represents each sentence by counting the term frequency of every term in the vocabulary. Then the vector value of each sentence would be the input to train the ML classifier (i.e., SVM). The hyperparameters of SVM remain the same with our previous work. We used scikit-learn toolbox to implement the technique and ML classifier.

D. Evaluation

Precision, recall, and F1-score were adopted as standard metrics to evaluate the performance of automatic approach. In our context, precision is the ratio of sentences correctly identified as decision sentences to all sentences identified as decision sentences, while recall is the fraction of all decision sentences correctly demarcated. F1-score is the harmonic mean of precision and recall. While our previous experiment evaluated the approach through **.90-split** which describes a single stratified 90% train and 10% test split of the dataset, we used a stratified 10-fold cross-validation referred to as **10-fold** in this experiment, which is common in machine learning as it produces less biased accuracy estimations for dataset with small sample sizes [11].

IV. Results and Analysis

A. RQ1: The impact of improved positive samples

For answering RQ1, we used **Non-Decisions from EASE2020** as the negative samples, and compared the classification results when separately using **Decisions from EASE2020** and **Decisions from Relabelled** as the positive samples in the dataset. To avoid the influence of size of dataset, we randomly selected 650 *decisions* from 844 *decisions* in **Decisions from Relabelled**. In addition, the procedure of random selection is repeated ten times and we calculated the average results obtained by all times (see Table III). The calculation results show that the automatic approach achieves better performance when using our relabelled *decisions* as positive samples in the dataset (with a precision of 0.678, a recall of 0.756, and an F1-score of 0.714). Compared with the original positive samples (i.e., **Decisions from EASE2020**), the precision, recall, and F1-score are increased by 7.4%, 9.6%, and 8.5% respectively, which indicates that the quality of positive samples in the dataset indeed has an impact on the classification results. We also note that the results shown in Table III when using both **Decisions from EASE2020** and **Non-Decisions from EASE2020** are different from the reported EASE2020 results [8]. The reason is that we used different evaluation methods with the same identification approach in this experiment (see Section III-D). In summary, **the quality of positive samples in the dataset has an impact on the classification results when automatically identifying decisions from the Hibernate developer mailing list, and improving the quality of positive samples can decently improve the classification results.**

TABLE III: Average Classification Results of Automatic Identification Approach for Different Positive Samples

Dataset	Non-Decisions from EASE2020		
	precision	recall	F1-score
Decisions from EASE2020	0.631	0.690	0.658
Decisions from Relabelled	0.678	0.756	0.714

B. RQ2: The impact of negative samples

For answering RQ2, we used **Decisions from Relabelled** as the positive samples since we draw conclusion in RQ1 that positive samples with high quality can improve the classification results. Then we compared the classification results when separately using **Non-Decisions from EASE2020** and **Non-Decisions from Assumptions** as the negative samples in the dataset. Similarly to RQ1, we also randomly selected 650 *assumptions* from 750 *assumptions* in **Non-Decisions from Assumptions** and repeated this random selection ten times. The average results are calculated and listed in Table IV. From Table IV, we can find that when using **Non-Decisions from Assumptions** as negative samples, compared with **Non-Decisions from EASE2020**, the precision is fairly increased by 7.8% whereas the recall is slightly decreased. Overall, the automatic approach achieves better performance when using *assumptions* as negative samples. It shows that **negative samples in the dataset have an impact on the classification results when automatically identifying decisions from the Hibernate developer mailing list.** Therefore, it indicates that the properties of negative samples should also be considered in automatic identification tasks in order to make the approach valuable in practice.

TABLE IV: Average Classification Results of Automatic Identification Approach for Different Negative Samples

Dataset	Decisions from Relabelled		
	precision	recall	F1-score
Non-Decisions from EASE2020	0.678	0.756	0.714
Non-Decisions from Assumptions	0.731	0.749	0.738

C. RQ3: The impact of size of dataset

For answering RQ3, we chose seven sizes (i.e., 400, 600, 800, 1000, 1200, 1400, and 1600) of dataset to investigate the trend of performance of the supervised ML classifier (i.e., SVM) on different sizes of the dataset. To avoid the effect of imbalanced dataset, the number of selected *decisions* is equal to the number of *non-decisions* in the dataset (e.g., 500 *decisions* and 500 *non-decisions* when the dataset size is 1000). Especially, due to the limited size of **Non-Decisions from EASE2020** and **Non-Decisions from Assumptions**, we combined these two datasets and randomly selected *non-decisions* from the combined dataset. For each size of dataset, we also repeated the random selection ten times and calculated the average classification results as shown in Table V. To show the trend of the performance of SVM-based classifier, we also plotted the results in a line chart (as shown in Fig. 1). According to the Table V and Fig. 1, we find that the precision increases with the size of the dataset, and the recall increases before the size of dataset reaches 1200 and then slightly decreases. We can also see that the F1-score shows a roughly similar trend as the recall. The F1-score achieves the highest value (i.e., 0.72) when the size of dataset reaches 1200 and then slightly fluctuates around 0.72. Considering the training time of supervised ML classifiers, we argue that 1200 is a sufficient number of size of dataset for training a well-performed classifier when automatically identifying decisions as larger size of dataset consumes more time to train. In summary, **the size of dataset has an impact on the classification results. Before the size reaches 1200, increasing the size of dataset will improve the classification results when automatically identifying decisions from the Hibernate developer mailing list.**

TABLE V: Average Classification Results of Automatic Identification Approach for Different Size of Dataset

Dataset Size	Precision	Recall	F1-score
400	0.669	0.708	0.683
600	0.678	0.710	0.691
800	0.695	0.727	0.708
1000	0.704	0.730	0.715
1200	0.708	0.737	0.721
1400	0.709	0.733	0.720
1600	0.713	0.733	0.721

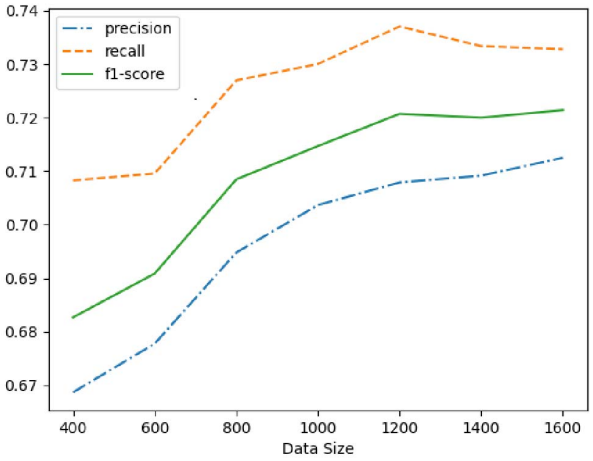

Fig. 1: Trend lines of Precision, Recall, and F1-score for the SVM-based classifier on different sizes of the dataset

V. Threats to Validity

The threats to the validity are discussed by following the guidelines *et al.* [12], and internal validity was not discussed since we did not study causality.

Construct Validity reflects on the extent of consistency between the operational measures of the study and the RQs [12]. A potential threat in this study involves whether the *decision* sentences used for the experiments were labelled correctly by the researchers. To mitigate this threat, we labelled the decisions by two researchers. Any disagreements were discussed and resolved with another researcher. Another potential threat is whether the dataset for the experiments is sufficient enough to obtain reasonable conclusions. To mitigate this threat, we used a random sample of collected decisions and non-decisions, and repeated this random selection ten times in our experiment. In addition, Manning *et al.* argued that the classification performance will be improved when increasing the sample size of the dataset [13]. Therefore, we conjecture that we can obtain more accurate conclusions and mitigate this threat by increasing the sample size of the dataset.

External Validity refers to the degree to which our study results and findings can be generalized in other cases with similar characteristics (i.e., the developer mailing lists of other OSS projects). We collected *decisions* and *non-decisions* from the developer mailing list of Hibernate, one of the most popular OSS projects and is widely used in the industry. Considering its popularity and robustness, we argue that Hibernate can be a representative in many OSS projects and can increase the generalizability of our experiment results.

Reliability refers to whether the experiment yields the same results when it is replicated by other researchers. To mitigate this threat, the research protocol was discussed and confirmed by all the researchers iteratively, and we also made the dataset and the source code of our experiments available online with the aim of facilitating other researchers replicating our experiments easily [9]. We believe that this measure can partially alleviate the threat.

VI. Conclusions and Future Work

In this work, we replicated our EASE2020 work with the best approach for automatically identifying decisions from the Hibernate developer mailing list [8], and we focused on exploring the impact of the proprieties of dataset on the classification results. Besides the EASE2020 dataset in our previous work [8], we also utilized a relabelled *decision* dataset and a new *non-decision* dataset from assumptions to conduct this replication experiment. The results show that (1) improving the quality of positive samples in the dataset can decently improve the classification results; (2) different negative samples in the dataset have an impact on the classification results; and (3) before the size reaches 1200, increasing the size of dataset will improve the classification results.

Given the importance of decisions in development, our future work is to: (1) collect *decisions* and *non-decisions* from other OSS projects or resources, and validate our conclusions on these projects, (2) use automatic approaches to mine diverse information related to decisions (e.g., rationale of decisions) in order to enrich the documentation of decisions.

References

[1] X. Li, P. Liang, and T. Liu, "Decisions and their making in oss development: An exploratory study using the hibernate developer mailing list," in *Proceedings of the 26th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 2019, pp. 323–330.

[2] A. Shahbazian, Y. K. Lee, D. Le, Y. Brun, and N. Medvidovic, "Recovering architectural design decisions," in *Proceedings of IEEE International Conference on Software Architecture (ICSA)*. IEEE, 2018, pp. 95–104.

[3] T.-M. Hesse, B. Paech, T. Roehm, and B. Bruegge, "How to improve decision documentation in software evolution?" in *Software Engineering (Workshops)*, 2014, pp. 14–15.

[4] M. Bhat, K. Shumaiev, A. Biesdorf, U. Hohenstein, and F. Matthes, "Automatic extraction of design decisions from issue management systems: A machine learning based approach," in *Proceedings of the 11th European Conference on Software Architecture (ECSA)*. Springer, 2017, pp. 138–154.

[5] J. S. van der Ven, A. G. Jansen, J. A. Nijhuis, and J. Bosch, "Design Decisions: The Bridge between Rationale and Architecture," in *Rationale Management in Software Engineering*. Springer, 2006, pp. 329–348.

[6] A. Tang, M. A. Babar, I. Gorton, and J. Han, "A survey of architecture design rationale," *Journal of Systems and Software*, vol. 79, pp. 1792–1804, 2006.

[7] A. Jansen and J. Bosch, "Software architecture as a set of architectural design decisions," in *Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture (WICSA)*. IEEE, 2005, pp. 109–120.

[8] X. Li, P. Liang, and Z. Li, "Automatic identification of decisions from the hibernate developer mailing list," in *Proceedings of the 24th International Conference on Evaluation and Assessment in Software Engineering (EASE)*. ACM, 2020, pp. 51–60.

[9] L. Fu, P. Liang, X. Li, and C. Yang, "Replication package for the paper: Will data influence the experiment results?: A replication study of automatic identification of decisions." http://doi.org/10.5281/zenodo.4412131, 2020.

[10] R. Li, P. Liang, C. Yang, G. Digkas, A. Chatzigeorgiou, and Z. Xiong, "Automatic identification of assumptions from the hibernate developer mailing list," in *Proceedings of the 26th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 2019, pp. 394–401.

[11] P. Refaeilzadeh, L. Tang, and H. Liu, "Cross-validation." *Encyclopedia of database systems*, vol. 5, pp. 532–538, 2009.

[12] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 2, pp. 131–164, 2009.

[13] C. D. Manning, C. D. Manning, and H. Schütze, *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.