Non-Functional Requirement Detection Using Machine Learning and Natural Language Processing

Hazlina Shariff[1], Mar Yah Said[*2]

[1,2]Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43400, Serdang, Selangor, Malaysia
ein_4657@yahoo.com[1], maryah@upm.edu.my[*2]

Article History: Received: 10 November 2020; Revised: 12 January 2021; Accepted: 27 January 2021; Published online: 05 April 2021

Abstract: A key aspect of software quality is when the software has been operated functionally and meets user needs. A primary concern with non-functional requirements is that they always being neglected because their information is hidden in the documents. NFR is a tacit knowledge about the system and as a human, a user usually hardly know how to describe NFR. Hence, affect the NFR to be absent during the elicitation process. The software engineer has to act proactively to demand the software quality criteria from the user so the objective of requirements can be achieved. In order to overcome these problems, we use machine learning to detect the indicator term of NFR in textual requirements so we can remind the software engineer to elicit the missing NFR. We developed a prototype tool to support our approach to classify the textual requirements and using supervised machine learning algorithms. Survey was done to evaluate the effectiveness of the prototype tool in detecting the NFR.

Keywords: non-functional requirement, software requirement, machine learning, natural language processing.

1. Introduction

Requirements elicitation is an important activity in the systems analysis and design process. Elicitation must focus on the creation of requirements in order to adequately address users' concerns and not just the developers' needs. Prior research (Berk, 2016) shows that the requirements elicitation process is fraught with poor communication, lack of stakeholder involvement and cooperation, conflict, as well as stress. During software development, a software engineer must clearly document the functional and non-functional requirements (NFR) in order to make a clear decision on the architecture and planning quality assurance. To successfully address non-functional characteristics in these phases, it is essential to elicit and capture the NFR during the requirements engineering phase. It is normal sometimes when the user doesn't know to describe things they know. NFR is a tacit knowledge that user always faces the problem to express it especially during an elicitation process. Even when they know, user stories tend to be unclear, not precise and ambiguous and may lead the software developer to interpret in many ways because of the unfamiliarity of the NFR aspect.

Information about NFR frequently hidden inside notes and therefore they are frequently missed or forgotten (Feng et. al., 2017). Elicitation of NFR is more technical than functional requirements, so users are not aware of the technical part of the system during the elicitation process (P. Maragathavalli, et al, 2020). The NFR should be treated as important as a functional requirement so the quality of the system can be determined in early stage. In reality, real problems concern on non-functionally more than functionally (Chung et al., 2009).

This work focuses on the non-functional requirements of usability, security and performance. A prototype tool has been developed meant for software engineers. The tool consists of functions to upload a list of textual requirements and then by using machine learning (ML) and natural language processing (NLP), the tool detects and reports the NFR presence. The remainder of this paper is organized as follows. Section 2 discusses on the existing materials related to the area of study and methods that were employed in the course of this study. Section 3 presents the results of this study and discussion on the results. Section 4 discusses on the conclusion of this study.

2. Materials and Methods

2.1 Literature Review

Software requirements are characterized into process and product. Process is based on cost, time and organization, while product is considering functional and non-functional requirements. Functional requirements are viewed from user side (user requirements) and developer side while non-functional requirements such as are considered under the responsibility of software engineer (Cleland-Huang et al., 2006); (Nathan et al, 2016). The software engineer will need to decide NFR because the NFR are quality of the system that only technical person will understand the terms used.

There are many definitions of NFR given by the researchers. However, in general, they agree that NFR are very important in software (Glinz, 2007). NFR should be present at early software development phase to avoid developing the wrong system and will affect the increase of the cost (Farhat et. al., 2009). Besides discovering missing or wrong requirement late in development, this may cause schedule delays, missed expectation or even project cancellation. This quality aspect also often neglected and taken as "fix-it-later" approach.

Many NLP processes can be adapted to provide an effective analysis of the requirements. The main processes are (Verspoor et. al., 2013) normalization, remove stop words, tokenization, N-Gram, name entity recognition (NER), part of speech (POS) and stemming. Text normalization is the process of transforming the text into a single canonical form that it might not have before. Remove stop words is to excludes the connecting words like 'and, 'the' and 'has' (Bdour & Gharaibeh, 2013). Tokenization (Motik et al., 2011) divides the character sequence based on the whitespace position or other punctuation marks between words in the sentence. N-gram method makes single or sequence of words based on a given statement (Jian et al., 2010). NER determines elements in a given text as objects. POS looks for patterns of sentences from texts. Stemming forms root word by removing suffixes and prefixes in a word.

Support Vector Machine (SVM) and Naïve Bayes (NB) is a supervised machine learning algorithm which best known in text classification (Bhavani et. al, 2012). Recent studies on the ML algorithm found that the combination of SVM and NB can increase the accuracy level of the classifier to classify the data by considering advantages for both SVM and NB (Feng et al., 2017). For this reason, we use SVM and NB in our tool.

There are many works on NFR using ML and NLP approaches. Cleland-Huang et al., (2006) found 15 indicator terms for some NFR and have been used in many NFR studies by other researchers. They successfully detect and classify NFRs from 15 SRS documents. By using the assumption of different kind of indicator terms for each NFR, they work on classifying NFR in each SRS accordingly. Zhang et al., (2011) performed a study on the effect of different features, including original words, N-grams, and phrases in detecting NFRs. Portugal et al., (2018) use key mining based on syntactic analysis of the requirements texts with automated support using NLP and text-mining techniques. Cleland et al.(2007) introduced an information retrieval approach on classifying NFR from the requirements specification and a free-form text. The classifier of NFR was used to evaluate the requirements and it has an ability to trawl through large free-form datasets of requirements during the elicitation process. The classifiers then parse the requirements and extract them into different types of non-functional requirement. Toth & Vidacs (2018) has conducted an experiment in identifying appropriate machine learning methods can be used for requirement classification task to support business analysts in their elicitation process. They used a small database containing labelled examples to train the classifiers. They employed under- and over-sampling strategies to handle the imbalanced classes in the dataset and cross-validated the classifiers based on the Support Vector Machine classifier algorithm. Kurtanovic & Maalej (2017) worked on the automatic classification on the requirement to FR and NFR using SVM and lexical features.

2.2 Methodology

This work has employs the steps of preliminary study, designed framework, implementation, and evaluation as its methodology. During the preliminary study stage, a comprehensive literature search was conducted using the databases. A wide range of databases such as IEEE, Research Gate, Springer, Google Scholar and Association for Computing Machinery Digital Library (ACM) were utilized for the literature search. The search terms include a non-functional requirement, machine learning, quality requirement, supervised algorithm, requirement engineering, text classification and text processing. Also, additional searches were extended from articles' citations. Papers discussing empirical research on machine learning algorithm are given less emphasis while papers providing on text classification were given more weight. Study text classification and its process involved are studied as to decide the algorithm used and why they used such algorithms.

Next we design a framework to get a clear picture of the elements of the implemented NFR detection tool. The framework consists of four elements which are input, text pre-processing, term-indicator and classifier. Subsequently, the development of NFR Detection Tool as a web application was done. Basically, this tool has three modules: login, training, and classification. The NB and SVM algorithms are applied into the classification module and the design of the interface allows users to upload CSV documents that consist of the list of requirements. This work has applied the Natural Language Processing (NLP) to process the requirements so that they are in appropriate format before the classification process. Our work has used the techniques of tokenization, stop-words removal and stemming.

We evaluated the tool by doing closed-ended questionnaires with software engineers to validate the developed tool's ability to detect the NFR. Respondents are given chance to hands-on the tool before participating in an evaluation survey. Besides, evaluation is also performed to examine the effectiveness of the tools to label the requirements keyed-in by the user whether each is classified under *Usability*, *Performance*, *Security* or *Not Labelled*. *Not Labelled* means the text classified is either the functional requirement or other non-functional requirement that were not covered in the scope of study.

2.3 Tool Design and Evaluation

Figure 1 shows the framework of the tool. User Requirement Specification (URS) is used as an input for the machine learning training. Pre-processing is a process transforms input data into a format that is easy and effective for processing computationally. F(X) GET TEXT is used to capture the data from the input and F(X) CLEAN is a function to clean captured data in order to transform the noisy data into clean ones. Next step, F(X) REMOVE STR is to remove a string from the text. Then all the words will be transformed into lower case. Indicator terms are terms that characterize each of the NFR supported in this work. Classifier of the tool has applied the SVM and NB, the supervised machine learning algorithms.

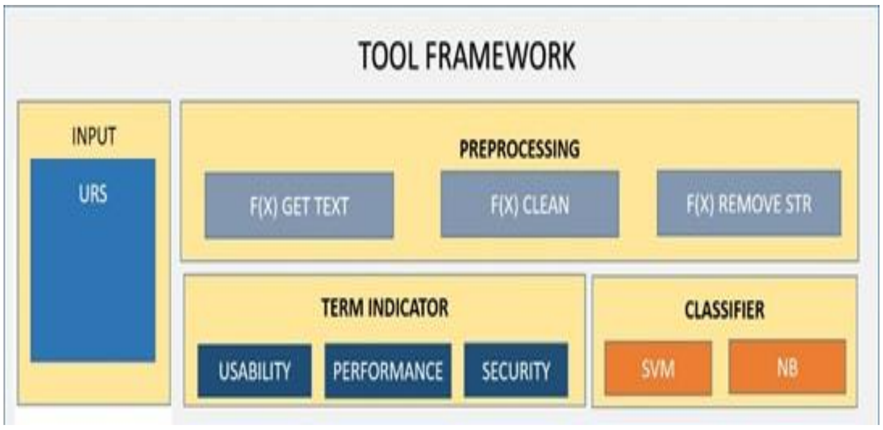


Figure 1. Tool Framework

Figure 2 shows the flow of the NFR detection tool which consists of the process that involves main modules: Text Pre-processing, Classifier training and Text Classification.

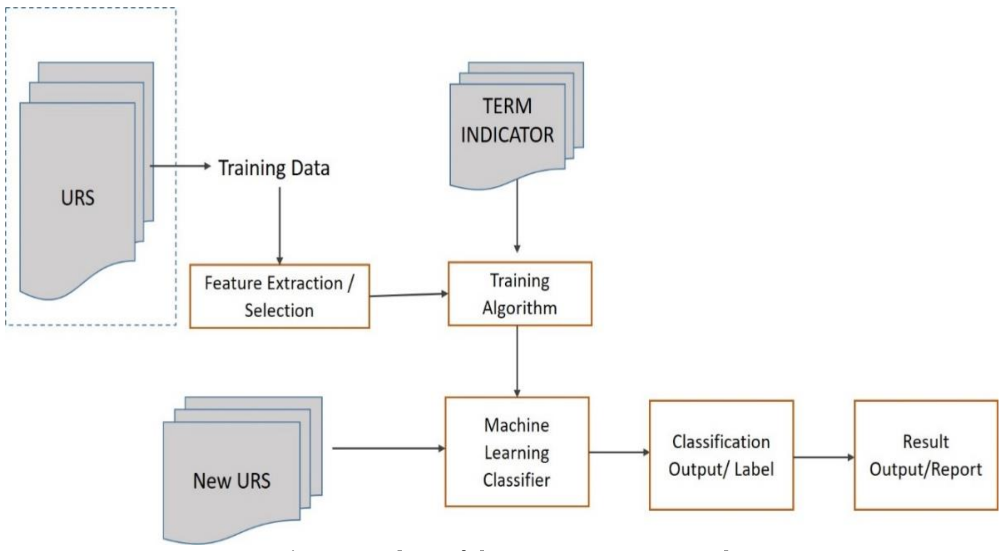
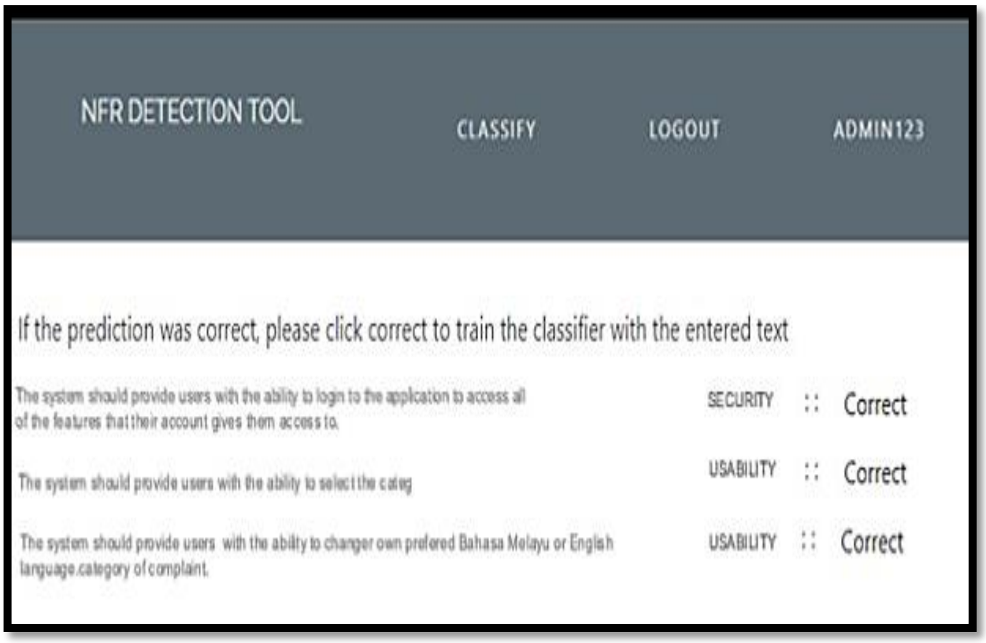

URS
Training Data
TERM INDICATOR
Feature Extraction / Selection
Training Algorithm
New URS
Machine Learning Classifier
Classification Output/ Label
Result Output/Report

Figure 2. Flow of the NFR Detection Tool

The inputs from the User Requirement Specification (URS) are used in the classifier training module. The pre-processing module takes control of the text by executing text processing, text cleaning and removing string in the text so the words are executable by the machine. The output of the pre-processing is the features that are used in the training algorithm. The user then will give input for new requirements and the pre-processing modules will then happen to extract the features from the data given.

SVM and NB classifiers are the learning model which needs to be learned with a labelled quantity. In the training phase, the requirements are labelled in advance, as unseen document and then their categories are estimated according to the generated features. The terms indicator are used to classify the new requirements in the database using the function where every term will be weight with regard to a specific NFR type. A single requirement could be classified into more than one NFR type but the highest classification value will be chosen as the final type. This terms indicator have been introduced by Cleland-Huang et al. (2006).

NFR Detection tool receive input in .csv format from the user produce a classification result as an output. The pre-processing and classification processes are done in the back-end of the tool. The output for this tool will be the classification result for the CSV formatted document uploaded. Figure 3 depicted the result from the classification component. User needs to verify the result by clicking the 'Correct' button.



If the prediction was correct, please click correct to train the classifier with the entered text

The system should provide users with the ability to login to the application to access all of the features that their account gives them access to.	SECURITY	::	Correct
The system should provide users with the ability to select the categ	USABILITY	::	Correct
The system should provide users with the abilty to changer own prefered Bahasa Melayu or English language.category of complaint.	USABILITY	::	Correct

Figure 3. Classification Result

Evaluation of the NFR Detection tool was performed through a survey study to evaluate the ability of the tool to classify the requirements into Security, Usability, Performance or Others. The survey was conducted by following the good practice suggested by Kelley, K. (2003) and Yue et al. (2018). They provide the guidance of good practice for novice researcher in order to produce a high-quality survey from planning until data analysis.

3. Results And discussion

In this study, we have presented an approach to NFR detection in order to support the requirement elicitation process by using ML and natural language processing. We designed a framework and implement the tool that classifies the NFRs from the CSV file uploaded by the user. In this study, we only focus on usability, security and performance requirement.

The classifiers we used are NB and SVM that are the most popular classifiers in text classification research. The dataset we collected from government agencies were split into a training dataset and test dataset. As the validation part, we did an evaluation survey among the system analysts in the government agencies that involve in requirement elicitation for the Centralized Complaints Management System project. By using the tool, the elicitation of NFR is improved by detecting the NFR from the requirement text. For this, a requirement documentation that is established may ensure that no NFR is neglected.

For the evaluation survey, system analysts are required to have experience on eliciting functional and non-functional requirements. They need to know the difference between functional and non-functional requirements to verify the output. From the observation, the more experience the person is, the faster he can verify the output. The tool may help the novice with least experience to understand and easily capture the NFR in their requirements. Also, the tool assists system analyst to detect the NFR from the functional requirement elicited.

The non-random survey was distributed to 15 system analysts to manually determine the results of classification. System analysts that experiences in the requirement engineering process were the respondent for this survey including the novice. They were familiar with the software requirements process and they have experience in eliciting the non-functional requirements. The survey is done face to face and the respondents need to use the tool before answering the survey. The respondents involved in Centralized Complaints Management System development project. The survey uses a Likert Scale which weighted average via a weighting scheme of rating: 1 = Disagree, 2 = Neutral, 3 = Agree, 4 = Strongly agree. Table 1 shows the result of the evaluation. All the participants agree that the tool achieves the objectives. In terms of tool abilities to give the expected output, most of the participants agree. The tool also can support the user in the elicitation process. The tool can detect the NFR from the functional requirements. With this tool, user can improve the quality of the elicitation process. They agree that this tool is suitable for novice system analyst. Some of the participants just prefer to choose neutral when it comes to the last criteria.

Table 1. Result of the evaluation on the effectiveness of the NFR detection tool

		Strongly agree	**agree**	**neutral**
1	**Ability**	27%	73%	-
2	**Support the elicitation process**	7%	80%	13%
3	**Detect NFR from functional requirements**	-	67%	33%
4	**Improve the elicitation process**	60%	40%	-
5	[illegible]	[illegible]	[illegible]	
6	**User friendly**	-	80%	20%

4. Conclusion

This study contributes to requirement engineering domains where we adapted machine learning and natural language processing techniques in the NFR detection tool. The tool may reduces cases where NFR were neglected and discovered in later stages in development. The tool may help novice system analyst to identify any NFR in their requirements artifacts.

Result from this study shows that non-functional requirements are often neglected and are considered towards the end of the system developed. These will cost money and other problems (Farhat et. al. 2009). This study reviewed selected related work and found out that combination of SVM and NB increase the accuracy of detection tool. Based on that, NFR detection tool framework is designed, developed and evaluated. From the evaluation result, we conclude that the tool may assist the software developers in considering and detecting NFR in the requirement engineering phase.

5. Acknowledgement

The authors would like to express appreciation for the support of the Faculty of Computer Science and Information Technology, Universiti Putra Malaysia.

References

1. Berk R.A. (2017). Support Vector Machines. In: Statistical Learning from a Regression Perspective. Second Edition. Springer Texts in Statistics. Springer, Switzerland.
2. Chung, L., & do Prado Leite, J. C. S. (2009). On non-functional requirements in software engineering. In Conceptual modelling: Foundations and applications (pp. 363-379). Springer Berlin Heidelberg.
3. Cleland-huang, J., Settimi, R., Zou, X., & Solc, P. (2006). The Detection and Classification of Non-Functional Requirements with Application to Early Aspects. In the 14th IEEE International Requirements Engineering Conference (RE'06), Minneapolis/St. Paul, MN, 2006, pp. 39-48, doi: 10.1109/RE.2006.65.
4. Dasari, B., Rao and V. G.. K, Durga. (2012). Text Categorization and Machine Learning Methods: Current State Of The Art. Global Journal of Computer Science and Technology, 12(11). ISSN 0975-4172.
5. Farhat, G. S., and Mitropoulos, F. I. (2009). Refining and reasoning about nonfunctional requirements. In the 47th Annual Southeast Regional Conference (ACM-SE 47),2009, pp. 1-5.
6. Feng, W., Sun, J., Zhang, L., Cao, C., & Yang, Q. (2016). A support vector machine based naive bayes algorithm for spam filtering. In the IEEE 35th International Performance Computing and Communications Conference, IPCCC 2016, Las Vegas, USA.

https://doi.org/10.1109/PCCC.2016.7820655

7. Huang, J., Gao, J., Miao, J., Li, X., Wang, K. and Behr, F. (2010). Exploring web-scale language models for search query processing. In the 19th International World Wide Web Conference (WWW-2010), Raleigh, NC.
8. Kelley, K. (2003). Good practice in the conduct and reporting of survey research. International Journal for Quality in Health Care, 15(3), 261–266. doi:10.1093/intqhc/mzg031.
9. Kurtanovic, Z., & Maalej, W. (2017). Automatically Classifying Functional and Non-functional Requirements Using Supervised Machine Learning. In the IEEE 25th International Requirements Engineering Conference (RE), Lisbon, Portugal.
10. M. Glinz. (2007). On non-functional requirements. In the 15th IEEE International Requirements Engineering Conference, 2007, New Delhi, India.
11. Nathan, S.S., Hussain, A., Hashim, N.L. (2016). Studies on deaf mobile application: Need for functionalities and requirements. Journal of Telecommunication, Electronic and Computer Engineering, 8 (8), pp. 47-50.
12. Motik, Nifierola, A. B. (2011). (12) Patent Application Publication (10) Pub. No. : US 2011/ 0320187 A1 Map The Natural Language Question into One Or More Compute One Or More Result Sets of the Natural Perform An intent Detection Of One Or More Semantic Hyper graphs 145 By Transforming One Or More Semantic Hyper graphs into One Or - /, 1(19).
13. Portugal, R., Li, T., Silva, L., Almentero, E. and Leite, J. (2018). NFR Finder: A Knowledge Based Strategy for Mining Non-Functional Requirements. in the XXXII Brazilian Symposium on Software Engineering Conference, São Carlos, São Paulo, Brasil.pp. 102–111. 10.1145/3266237.3266269.
14. P. Maragathavalli, B. Tamilarasi, R. Nivetha & S. Anjali. 2020. Machine Learning Algorithms to Detect Suspicious Domain Names in Internet Security. IIRJET, V-5, I-3, IT-55 - IT-64.
15. Tóth L. andVidács L. (2018) Study of Various Classifiers for Identification and Classification of Non-functional Requirements. In: Gervasi O. et al. (eds) Computational Science and Its Applications – ICCSA 2018. ICCSA 2018. Lecture Notes in Computer Science, vol 10964. Springer, Cham.
16. Verspoor, K. and Cohen, K. (2013). Natural Language Processing. 10.1007/978-1-4419-9863-7_158.
17. Bdour, W. N. and Gharaibeh, N. K. (2013) Development of Yes/No Arabic Question Answering System. International Journal of Artificial Intelligence & Applications (IJAIA), 4(1), January 2013.
18. Yue, W.S., Chye, K.K., Hoy, C.W., Hussain, A. (2018). Strategy model in bus tracking and information application (BTA) towards smart mobility in urban spaces. Journal of Telecommunication, Electronic and Computer Engineering, 10 (1-10), pp. 109-114.
19. Zhang, W., Ye, Y., Qing, W. and Fengdi, S. (2011). An empirical study on classification of non-functional requirements. In the Twenty-Third International Conference on Software Engineering and Knowledge Engineering (SEKE 11),Eden Roc Renaissance Miami Beach, USA, pp. 190-195.