Automatic Classification of Software Requirements in Vietnamese Based on Machine Learning Techniques

Thi Nham Cao[1], Dai Tho Dang[2], Thi My Hanh Le[3], Nguyen Thanh Binh[2]

[1]University Of Economics, The University Of Danang, Danang
[2]Vietnam - Korea University of Information and Communication Technology, The University of Danang, Danang
[3]University of Science and Technology, The University of Danang, Danang

Correspondence: Thanh Binh Nguyen, Email: ntbinh@vku.udn.vn
Communication: received 01/05/2022, revised 01/06/2022, accepted 30/06/2022
DOI: 10.32913/mic-ict-research.v2022.n1.1051

***Abstract*: Requirements engineering is often the first stage in the software process to understand the problem statement. Finding mistakes earlier in requirements helps reduce the development cost. One activity contributing to defining clear, complete and precise requirements is classifying requirement items in the specification. This paper presents a classification approach of functional and non-functional requirements in Vietnamese using different supervised machine learning techniques. Five supervised machine learning algorithms, including Naïve Bayes (NB), Support Vector Machine (SVM), Logistics Regression (LR), Multi-layer Perceptron Neural Network (MLP), and FastText, are implemented, trained, tested and compared using a dataset. The experimental results show that NB is the best model in terms of accuracy.**

Keywords: *Software engineering, requirements engineering, requirements classification, machine learning.*

I. Introduction

Requirement Engineering (RE) is a process of eliciting imprecise, incomplete needs and wishes of the potential users of the software, analyzing, validating, and translating them into complete, precise, and formal specifications. This activity is generally the first step in the software process. Misunderstanding requirements could be very serious. Because fixing a requirement mistake after delivery may cost up to 100 times the cost of fixing an implementation mistake. Therefore, the importance of RE is enormous to develop software and reduce errors at a very early stage of the development process.

Software requirements are often classified into two types: functional and non-functional. Functional requirements are statements of services or functions of software, while non-functional requirements are constraints on services or functions, such as performance, usability, operating platform, and software process... This classification impacts how requirements are processed during analysis, and validation activities. The requirement classification is often done manually in the software process. Thus it could require many developers' efforts, and they may fail to identify the critical requirements.

Recently, several studies have been conducted in this area to reduce efforts and help to improve the quality of the software requirements. One of them is automatically distinguishing functional from non-functional requirements [1-5, 13] or identifying non-functional [9, 12] or categorizing non-functional requirements [6, 7, 8] or prioritizing requirements [10, 11]. These works concentrate mostly on studying requirement specifications in English by using various machine learning techniques. The results are quite interesting. However, automatically classifying software requirements in Vietnamese has not been focused in the literature. Besides, no dataset for this problem has been built yet.

That is why we are motivated to study the application of machine learning algorithms in classifying software requirements in Vietnamese into functional and non-functional categories and evaluating their performance. In this paper, we first build a dataset consisting of functional and non-functional requirements, then propose a methodology for classifying requirements, select different machine learning algorithms for experimentation, and evaluate the experimental results.

The paper is organized as follows. Section I introduces the motivation of the study. The related works are presented in Section II. The methodology is described in Section III. Section IV is reserved for experimentation. The results are discussed in Section V. Finally, the paper finishes with the

conclusion and future works.

II. Related works

The requirements classification is often manually done late in the software process. Thus, automatic classification of software requirements into functional and non-functional requirements is helpful. This task-based on machine learning has been examined in English requirements.

In [1], the authors used the SVM for requirements classification. The study combines SVM and linguistic features as functional and non-functional requirements. It reaches a recall and precision of about 92% for both classes. This approach was developed and expanded to 17 linguistic features [3].

Another approach for preprocessing requirements was proposed in [2]. It standardizes and normalizes requirements before applying classification algorithms. The experimental results show that using this approach improves the efficiency of algorithms Latent Dirichlet Allocation, Biterm Topic Modeling, K-means, and Naïve Bayes in classifying requirements.

Software requirements usually vary in wording and style. Thus, the efficiency of machine learning algorithms reduces in the case used for unseen projects. To deal with the problem, using the transfer learning capabilities of BERT for requirements classification is proposed in [4]. This approach was used for different tasks in the domain of requirements classification. Results are similar or better (F1-scores up to 94%) on both seen and unseen projects for classifying functional and non-functional requirements.

In [5], the authors examined the problem of combining feature extraction techniques and machine learning algorithms for classifying software requirements. The feature extraction techniques are Bag of Words and Term Frequency - Inverse Document. The algorithms used for classification are LR, SVM, NB, and k-Nearest Neighbors. The experimental results show that the combination of Term Frequency - Inverse Document and LR has the best efficiency, with an F-measure of 91%.

In Vietnam, the software industry has been developing quickly in recent years. The total revenue of the software industry reached 5.44 billion U.S. dollars in 2020 [14]. However, the automatic classification of Vietnamese software requirements has not been investigated in the literature.

III. Methodology

This section presents the details of our proposed methodology for automatically classifying Vietnamese software requirements. The classification of requirements is performed in four phases. As illustrated in Figure 1, software requirements are pre-processed, and then features are extracted. Finally, five machine learning algorithms are performed and evaluated.

1. Data pre-processing

The most important phase in any data-driven project is obtaining quality data. Without these pre-processing steps, the results of a project can easily be biased or wholly misunderstood. This phase is the process of normalizing data and removing elements that are not meant for text classification. In this paper, we realize the following works for data pre-processing:

- *Removing HTML tags.* To build the dataset for this study, requirements were collected from websites. They may contain HTML tags; thus, it is necessary to remove these tags. Because HTML tags could lead to a bad result in text classification, we use regex in Python to remove these tags.
- *Normalizing Unicode encoding.* In Vietnamese, there are several forms of Vietnamese encoding, but the most popular ones are the combination Unicode and precomposed Unicode. Using different encoding standards in the same texts can lead to a problem: when putting a word into a training model, it will interpret that word as different words even though humans see it as the same. Our solution for this case is converting texts to the precomposed Unicode standard.
- *Standardizing typing styles.* Different typing styles could lead to different words in text processing. For instance, “òa” and “oà” are different. We develop a Python function to convert texts to the same typing style.
- *Converting to lower texts.* This work is essential because this feature does not contribute to the text classification result. Converting texts to lower cases also reduces features and increases the accuracy of the model.
- *Deleting special characters in texts.* Punctuations, numbers, and other special characters do not contribute to the result of text classification, so we remove them.
- *Removing stop words.* These words do not contribute to the classification result, thus we also develop a Python function to remove them from texts.
- *Tokenizing words.* There are currently several Vietnamese natural language processing toolkits such as VnCoreNLP, underthesea,... In this study, we use Python programming languages to develop models, therefore underthesea library, which is a suite of open-source Python modules, was used to tokenize the Vietnamese words.

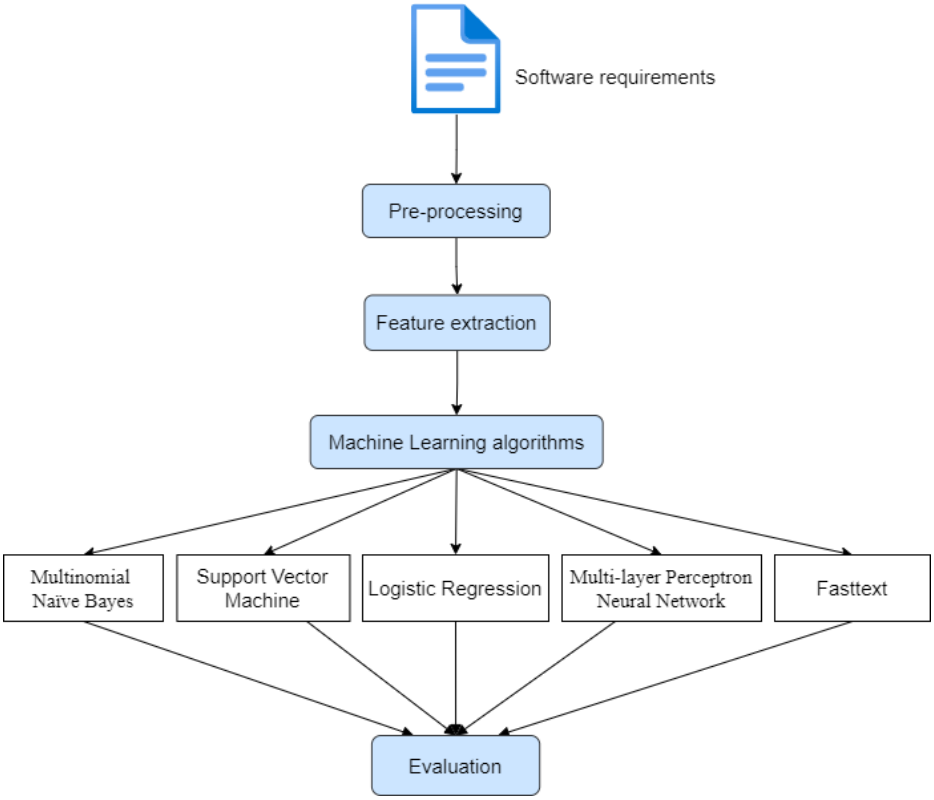


Figure 1. Schema of classifying software requirements in Vietnamese

2. Feature extraction

Machine learning techniques work on numeric features; therefore, we need to present them in numbers to be understood by the algorithms to deal with textual data. There are currently three popular ways to extract features from text documents: Bag of Words, Term Frequency - Inverse Document Frequency, and Word Counts. This uses word counts techniques with CountVectorizer of Scikitlearn library to convert texts to vector representations.

3. Classification

Supervised machine learning algorithms are used to classify software requirements into functional or non-functional categories. This study applies five algorithms as follows.

- *Naïve Bayes.* It is a probabilistic learning strategy often utilized in natural language processing. NB computes each tag's probability and gives the tag the highest probability as the outcome.
- *Support Vector Machine.* SVM is a supervised machine learning algorithm used for classification and regression challenges. Each data item is plotted as a point in n-dimensional. The value of each feature is the value of a specific coordinate. Then, the category is done by finding the hyperplane that differentiates well the two classes.
- *Logistics Regression.* LR is a classification algorithm that assigns observations to a discrete set of classes. It is the baseline supervised machine learning algorithm close to neural networks.
- *Multi-layer Perceptron Neural Network.* MLP includes at least three layers: an input layer, a hidden layer, and an output layer. MLP uses a supervised learning technique named backpropagation for training. Its multiple layers and nonlinear activation distinguish MLP from a linear perceptron.
- *FastText.* FastText is a method for encoding words as numeric vectors. It is much faster and easier to maintain than deep neural networks like BERT. Pretrained FastText embedding is utilized for solve problems, for example, named entity recognition or text classification.

4. Evaluation

To evaluate the ability to distinguish functional requirements and non-functional requirements, we use the Precision, Recall, and F1-score metrics. Before we get into precision and recall, we present True positive, True

TABLE I
EXPERIMENTAL RESULTS OF FIVE ALGORITHMS

Algorithms	Precision	Recall	F1-score	Training time (seconds)
Naïve Bayes	0.929	0.935	0.932	0.0172
Logistics regression	0.907	0.935	0.932	0.0259
Support Vector Machine	0.895	0.903	0.888	0.0168
Multi-layer Perceptron Neural Network	0.578	0.578	0.578	0.0153
FastText	0.755	0.755	0.755	1.5640

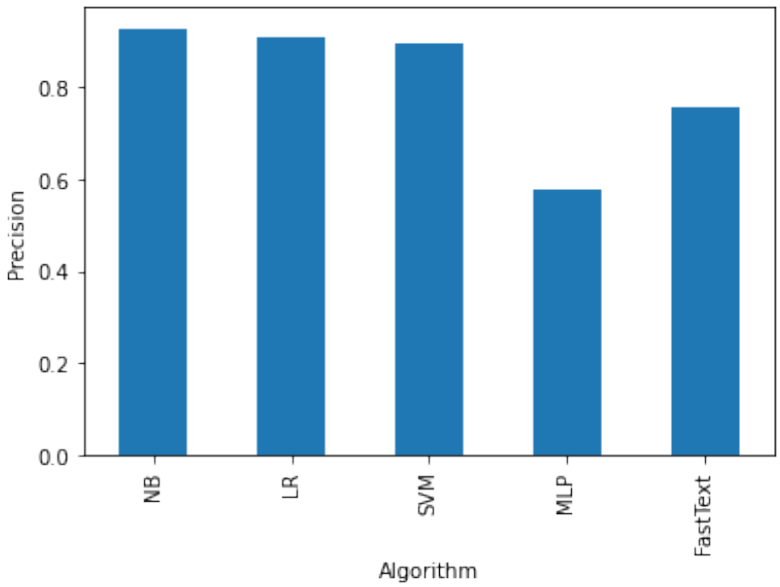


Figure 2. Algorithms comparison in terms of precision

negative, False positive, and False negative outcome classifications. True positives are understood positive results that the algorithm predicted correctly. True negatives are negative results that the algorithm predicted correctly. False positives are positive results that the algorithm predicted incorrectly. False negatives are negative results that the algorithm predicted incorrectly. By TP, TN, FP, and FN, we denote True positives, True negatives, False positives, and False negatives, respectively.

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. It is determined as follows:

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

Recall is the ratio of correctly predicted positive observations to all observations in an actual class. Recall is computed as the following:

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

F1-score is the weighted average of Precision and Recall. It is calculated as follows:

$$F_1 - Score = \frac{2 * Recall * Precision}{Recall + Precision} \tag{3}$$

IV. EXPERIMENT AND ANALYSIS

1. Dataset

Since there has not been an available software requirement dataset in Vietnamese yet, we firstly built a testing sample dataset. It contains 222 requirements collected from four different software projects on the Internet. The requirements are in the form of a sentence or a short paragraph, such as "Sản phẩm phải có bảng màu và phông chữ nhất quán". In this dataset, we labeled 131 requirements as functional and 91 requirements as non-functional.

2. Experimental setups

This study conducts experiments with the requirements classification in Vietnamese with algorithms NB, SVM,

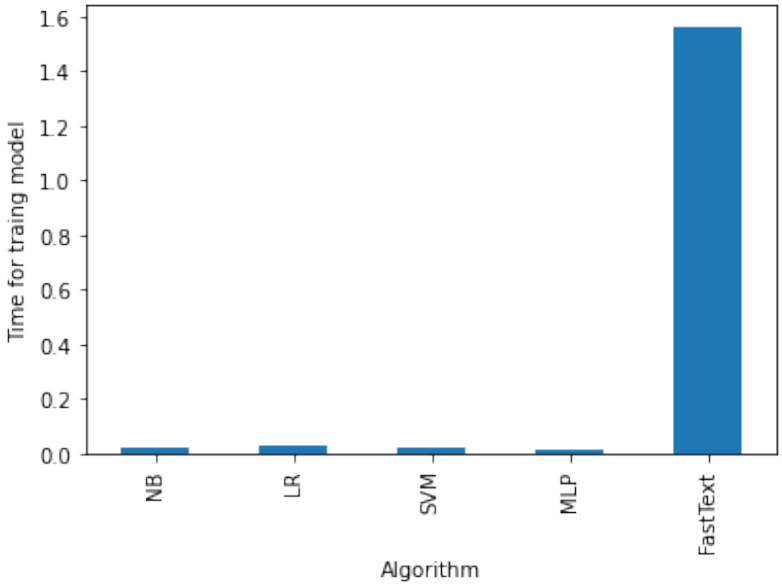


Figure 3. Algorithms comparison in terms of time for training model

LR, MLP, and FastText. These five algorithms all execute on the same data set that we built. We used the underthesea library to pre-process Vietnamese texts and the Scikit-learn library for five algorithms.

3. Experimental results

In this section, we evaluate the experimental results of five algorithms in the classification of requirements in Vietnamese. Table 1 shows the performance of each classification method.

Figure 2 shows a significant difference in precision value between the algorithms. NB has the best performance with 92.9% precision, and MLP is the worst with only 57.8%.

In terms of time for training models with the same training dataset size, FastText takes much more time than the other models, with 1.0629 seconds, while those of the others are only 0.01 – 0.03 seconds. We can see this difference clearly in Figure 3. Besides, the precision of FastText is only 75.5%.

We consider the remaining four models. The training time of MLP is the smallest with 0.0172 seconds, the second is SVM with 0.0168 seconds, the third is NB with 0.0172 seconds, and the fourth is LR with 0.0259 seconds. The difference between these fourth training times is not significant. Besides, NB has the best performance in all algorithms with 92.9% precision. We can assume that NB is best in classifying software requirements written in Vietnamese in both performance and time of model training.

V. Discussion

This section provides a discussion of the results in terms of the performance and how the results of this study can be useful for industry practitioners. In this era of software development, requirement analysis could generate a large amount of unstructured software specification documents during the elicitation process. The work in this paper is concerned with the development of models to classify functional and non-functional requirements in Vietnamese. The results of this study could help researchers as well as practitioners from the software industry find the type of requirements based on their description in the early phases of software development. It is a factor that contributes to software quality.

The experimental results show that NB and LR are the best supervised machine learning methods for classifying software requirements in Vietnamese into functional and non-functional categories. The efficiency of these two algorithms can be further improved by more suitable data representation.

Determining whether these two algorithms are efficient for Vietnamese texts in other fields requires much research with datasets in each field. It is also an issue that we will delve into in future research.

VI. Conclusion

Classification of software requirements into functional and non-functional is critical in the software process. In this study, we have presented a machine learning approach to classify functional and non-functional requirements in

Vietnamese in order to support the requirement elicitation process. We first built a dataset containing functional and non-functional requirements and then labeled them. After pre-processing the requirements, we performed five machine learning algorithms, including NB, SVM, LR, MLP, and FastText for the experiment. The experimental results show that the NB algorithm gives the best performance for distinguishing the requirements.

REFERENCES

[1] Zijad Kurtanovic, Walid Maalej, "Automatically Classifying Functional and Non-Functional Requirements Using Supervised Machine Learning," *IEEE International Conference on Requirements Engineering*, 2017, pp. 490-495.

[2] Zahra Shakeri Hossein Abad, Oliver Karras, Parisa Ghazi, Martin Glinz, Guenther Ruhe, Kurt Schneider, "What Works Better? A Study of Classifying Requirements," *25th IEEE International Requirements Engineering Conference*, 2017, pp. 496-501.

[3] Fabiano Dalpiaz, Davide Dell'Anna, Fatma Başak Aydemir, Sercan Çevikol, "Requirements Classification with Interpretable Machine Learning and Dependency Parsing," *IEEE International Conference on Requirements Engineering*, 2019, pp. 142-152.

[4] Tobias Hey, Jan Keim, Anne Koziolek, Walter F. Tichy, "NoRBERT- Transfer Learning for Requirements Classification," *IEEE 28th International Requirements Engineering Conference*, 2020.

[5] Edna Dias Canedo, Bruno Cordeiro Mendes, "Software Requirements Classification Using Machine Learning Algorithms," *Entropy*, vol. 22, 2020.

[6] Rajesh Kumar Gnanasekaran, Suranjan Chakraborty, Josh Dehlinger, Lin Deng, "Using recurrent neural networks for classification of natural language-based non-functional requirements," *Proceedings of REFSQ-2021 Workshops*, 2021.

[7] Rajni Jindal, Ruchika Malhotra, Abha Jain, Ankita Bansal, "Mining Non-Functional Requirements using Machine Learning Techniques," *e-Informatica Software Engineering Journal*, vol. 15, 2021.

[8] Nouf Rahimi, Fathy Eassa, Lamiaa Elrefaei, "An Ensemble Machine Learning Technique for Functional Requirement Classification," *Symmetry*, vol. 12 (10), 2019.

[9] Agustin Casamayor, Daniela Godoy, Marcelo Campo, "Identification of non-functional requirements in textual specifications: A semi-supervised learning approach," *Information and Software Technology*, vol. 52 (4), 2010.

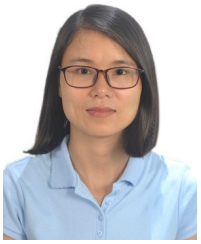
[10] Anna Perini, Angelo Susi, Paolo Avesani, "A Machine Learning Approach to Software Requirements Prioritization," *IEEE Transactions on Software Engineering*, vol. 9 (4), 2013.

[11] Lorijn van Rooijen, Frederik Simon Baumer, Marie Christin Platenius,Michaela Geierhos, Heiko Hamann, Gregor Engels, "From User Demand to Software Service: Using Machine Learning to Automate the Requirements Specification Process," *IEEE 25th International Requirements Engineering Conference Workshops*, 2017, pp. 379-385.

[12] Tetsuo Tamai, Taichi Anzai, "Quality Requirements Analysis with Machine Learning," *Proceedings of the 13th International Conference on Evaluation of Novel Approaches to Software Engineering*, 2018, pp. 241-248.

[13] Law Foong Li, Nicholas Chia Jin-An, Zarinah Mohd Kasirun, "An Empirical Comparison of Machine Learning Algorithms for Classification of Software Requirements," *International Journal of Advanced Computer Science and Applications*, vol. 10, 2019.

[14] Minh-Ngoc Nguyen, "Revenue of software industry in Vietnam 2016-2020", *Statista*, Jan 24, 2022 https://www.statista.com/statistics/1193528/vietnam-software-industry-revenue/



Thi Nham Cao graduated master program of software engineering at VNU University of Engineering and Technology in 2010. Now she is working at Faculty of Statistics and Informatics of Danang University of Economics. Her reseach interests are Machine Learning, Explainable Artificial Intelligence.

Dai Tho Dang received the Master of Computer Sciences from the Nice Sophia Antipolis, France, in 2010, and the Ph.D. degree from Yeungnam University, Republic of Korea, in cooperation with the Wrocław University of Science and Technology, Poland, in 2020. His research interests include Collective Intelligence, Evolutionary Computation, Deep Learning, and NLP. He is an active reviewer of IEEE Transactions on Cybernetics and Applied Intelligence.

Le Thi My Hanh is currently a lecturer of the Information Technology Faculty, University of Science and Technology, Danang, Vietnam. She gained M.Sc. degree in 2004 and the Ph.D. degree in Computer Science at the University of Danang in 2016. Her research interests are about software engineering and more generally application of heuristic techniques to problems in software engineering.
Email: ltmhanh@dut.udn.vn

Thanh Binh Nguyen graduated in Information Technology from the University of Danang - University of Science and Technology in 1997. He received PhD. degree in Information Technology at Grenoble Institute of Technology, France in 2004. He has been qualified as Associate Professor since 2013. He is currently working at the University of Danang - Vietnam-Korea University of Information and Communication Technology. His research interests include software engineering and software quality.
Email: ntbinh@vku.udn.vn