Multi-label Classifier to Deal with Misclassification in Non-functional Requirements

Maliha Sabir [0000-0001-5657-9929] ✉ ,Christos Chrysoulas ✉ , and Ebad Banissi ✉

[sabirm3@lsbu.ac.uk], [chrysouc@lsbu.ac.uk], [banisse@lsbu.ac.uk]
London South Bank University, 103 Borough Road,
London SE1 0AA, UK
www.lsbu.ac.uk

Abstract. Automatic classification of software requirements is an active research area; it can alleviate the tedious task of manual labeling and improves transparency in the requirements engineering process. Several attempts have been made towards the identification and classification by type of functional requirements (FRs) as well as non-functional requirements (NFRs). Previous work in this area suffers from misclassification. This study investigates issues with NFRs in particular the limitations of existing methods in the classification of NFRs. The goal of this work is to minimize misclassification and help stakeholders consider NFRs in early phases of development through automatically classifying requirements. In this study, we have proposed an improved requirement detection and classification technique. The following summarizes the proposed approach:

A newly created labeled corpus.
Textual semantics to augment user requirements by word2vec for automatically extracting features, and
A convolution neural network-based multi-label requirement classifier that classifies NFRs into five classes: reliability, efficiency, portability, usability, and maintainability.

Keywords: Non-functional requirements · Requirement engineering · Machine learning approaches · Natural language processing · Neural networks

1 Introduction

Requirements originate from stakeholders in natural language [1]. Non-functional requirements (NFRs) are always associated with functional requirements (FRs) [2]. While FRs describe the functions, tasks or behavior that a system must support [3], NFRs represent the particular qualities the systems must have. For a successful implementation, it is crucial to have comprehensive and transparent requirements for the identification of relevant constraints, assumptions, risks, and dependencies [4]. Fulfillment of a single FR can lead to the achievement of one or more NFRs [5]. NFRs are significant for the success of a software system [4]. However, literature suffers from both terminological and theoretical conflict [6]. Given the diversity of definitions of NFRs, stakeholders tend to disagree on important NFR attributes [7]. Each NFR attribute is treated differently and requires specialized expertise [8], but due to lack of resources and engineering skills, these have not been effectively addressed, resulting in higher structural defects and project failure rates [9, 10]. The complexity of NFR concepts has led to the development of numerous automated and semi-automated classification methods based on natural language processing (NLP), rule-based (RB) approaches, and machine learning (ML) approaches [11]. However, these tools are naïve and exhibit misclassification [12]. The literature highlights concerns in this domain, described below.

Limited in Generalizability

The RB approach has limited generalizability [13]. Rules need to be manually crafted and enhanced all the time. Moreover, the system can become overly complicated with some rules beginning to contradict each other [14]. The use of keywords or certain vocabulary is correlated with specific NFR attributes [15]. Furthermore, the RB approach is limited to defining rules for handling security, usability, and maintainability [14].

Limitation of Feature Selection Techniques

The fit criteria for NFR identification and ML training lacks clarity [16]. Mostly, syntactic feature part-of-speech (POS) tagging is used [13, 17–19]. Abad found that pre-processing improved the performance of an existing classification method [16]. The second most common technique, Bag of Word (BOW) [17, 18, 20, 21], fails to maintain the sentence order and cannot deal with polysemy [22]. In comparison, n-grams feature can consider the word order in a close contest to its neighboring words [20, 23], but it also suffers from data scarcity [24]. However, it is observed that the semantic knowledge of sentences based on word2vec reach better classification performance [5, 25]. In ML approach reliability, portability [26] and usability [16] have not been adequately addressed.

Limited NFR Ontology

Most of classification refers to functionality, availability, fault tolerance, legal, look and feel, maintainability, operational, performance, portability, scalability, security and usability. The NFR attributes chosen to be a part of existing studies are not the critical NFR attributes to be representative of its whole domain [27], leading to misclassification [5, 12].

Lack of Corpus

To a large extent, ML techniques are dependent on training data [28]. They require a representative and balanced corpus which is currently lacking [17, 26]. The PROMISE dataset used in previous studies has misconceptions about requirements. A corpus with an unbalanced number of examples of a specific linguistic construction can cause an algorithm to apply a specific label more frequently than others, resulting in a bias in the classification results [29].

Limitation to Handle Multi-label Classification

Peng [25] argues that one requirement could exhibit the characteristics of more than one attribute. In such a case, the classification should be made based on more than one label [29]. However, this has not been addressed.

This study proposes a novel approach for handling the misclassification of NFRs based on multiple labels using deep learning techniques. The paper is organized as follows: Section 2 describes work related to NFRs and their classification; Section 3 illustrates a proposed conceptual framework; and finally, Section 4 details recommendations for further research.

2 The Classification of NFRs

This section presents work related to the detection and classification of NFRs.

Cleland designed an NFR classifier which uses a fixed set of keywords to identify NFRs [30]. In another study by Cleland, first he describes the classification algorithm and then evaluates its effectiveness through a series of experiments and against a large requirement document obtained from Siemens Logistics and Automotive Organization [21].

Hussain proposes a supervised automatic classifier with the use of syntactic and keyword features for information retrieval for classifying NFRs. The experiment was performed for binary as well as multi-class NFR classification with the hybrid training set [19].

Vlas and Robinson developed RCNL, a multilevel ontology in which the lower levels apply generic English grammar-based concepts while the upper, abstract levels apply OSS requirements-based concepts [13].

Ormandjieva proposes a supervised learning approach based on a support vector machine (SVM) for the classification of the requirement into ISO/IEC 9126 ontology classes using Wen ontology language (OWL). The ontology contains dependency relationship between FRs and NFRs, further an association of NFRs with subcategories has been highlighted. They also proposed gold standard annotated NFR corpus [31].

Slankas proposes an NFR locator to locate and classify 14 NFRs using k-nearest neighbors (K-nn), SVM and naïve Bayes algorithms. The tool resulted in various misclassification and suffered from generalizability issues. Slankas found that certain features are associated with specific NFR attributes [15].

Sharma suggests an approach to identify NFRs based on rules extracting multiple syntactic and semantic features and relationship among them through a domain-specific language. The approach was practiced on a publicly available requirement corpus [14].

Singh proposes automated identification and classification of NFRs, and subclasses based on the ISO 9126 quality model. The study proposes an RB classifier using the thematic role as well as identified the priority of the extracted NFRs according to their occurrence. The results were analyzed on two corpora. The PROMISE corpus contains higher-ranked sentences than the Concordia Corpus [32].

Casamayor suggests a semi-supervised approach based on user feedback, iteratively collecting data using an expectation maximization strategy to enable learning an initial classifier for NFRs. It uses a multinomial naïve Bayes classifier with expectation maximization (EM) for the initial training of requirement documents labeled manually to train the binary classifier [18].

Zhang uses three kinds of index terms at different levels of linguistic semantics, as n-grams, individual words, and multi-word expressions (MWE) are used in the representation of NFRs. Then, an SVM with the linear kernel is used as the classifier [23].

Sunner proposes a supervised learning-based cluster technique requirement mining and classification of FR and NFRs in agile software development. In the study, k-means was used with the UPGMA classifier model, SVM was used with RBF kernel, and a neural network used with a genetic algorithm. The results show that a cluster neuro-genetic approach provides better results than the SVM and RBF kernel [17].

Mahmoud A. proposes an unsupervised approach that exploits the textual semantics of FRs to identify potential NFRs with a clustering algorithm. The experiment was performed on datasets from SafeDrink, SmartTrip, and BlueWallet [33].

Kurtanovic develops and evaluates a supervised machine learning approach employing meta-data, lexical, and syntactical features—in particular, usability, security, operational, and performance requirements [26].

Abad uses various feature extraction and feature selection techniques to maximize the accuracy of classification algorithms. This study was performed on the tera-PROMISE repository and shows that pre-processing improved the performance of an existing classification method [16].

Lu proposes an approach that uses textual semantics by word2vec for automatically classifying user reviews from two popular apps, iBooks and WhatsApp. The approach is used for classifying user reviews into NFRs with each user review sentence labeled as one type [25].

Younas came up with an automated semi-supervised semantic similarity-based approach that uses an application of the word2vec model and popular keywords for the identification of NFRs. The study as performed on the PROMISE-NFR dataset [5].

Baker made a first attempt to use a convolution neural network (CNN) and artificial neural network (ANN) for the classification of NFRs into five classes—maintainability, operability, performance, security, and usability—using an unsupervised approach. The experiment was performed on the PROMISE dataset. The resulting tool shows inexperience and lacks practicality [27].

3 Handling the Misclassification of NFRs

We have adopted a multi-label classification approach for handling misclassification. The proposed approach uses a CNN, which has proven to be successful for single-label classification [27]. A CNN has a clear advantage over traditional machine learning algorithms as it can learn and generate dense vectors for word representation. Traditional machine learning algorithms tend to divide the multi-label problem into multiple binary classifications, whereas a CNN has the ability to learn multiple labels in one classifier. The experiment is a two-step procedure which is described in Fig. 1

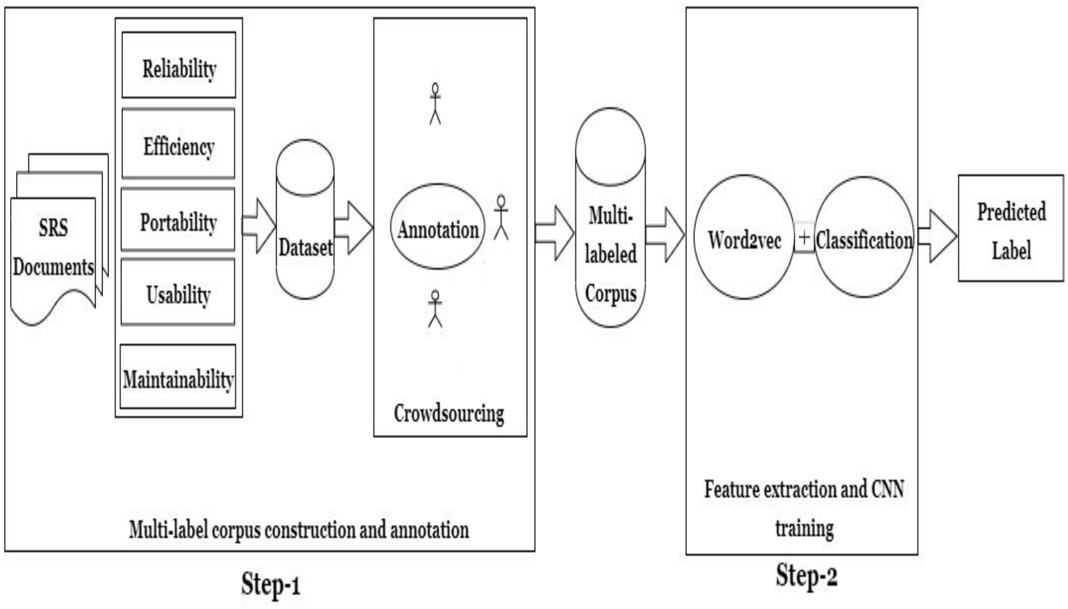

Fig. 1. -Conceptual framework for multi-label classification

3.1 Step 1: Corpus Construction and Annotation

In Fig. 1, the first step describes the multi-label corpus construction and annotation. The procedure starts with the identification of the critical NFR attributes—reliability, efficiency, portability, usability, and maintainability—based on the software quality model: McCall's, Boehm's, Dromey's, ISO 9125/25010 and FURPS/FURPS+ [34]. Various Software requirement specification (SRS) documents has been collected. The researcher manually extracts requirements related to the selected NFR attributes and records them in the form of a dataset.

To overcome the limitations of the existing corpus and construction of a new corpus, we have followed a set of guidelines through the MATTER-MAMA framework proposed by Stubs [35]. We aim to have a representative and balanced corpus which makes the corpus to be generalized to the domain it belongs to and contains roughly equal numbers of training examples of all attributes selected to be a part of this ontology.

To have a labeled corpus, all the requirements related to the selected categories will be re-assessed by a crowd. For instance, a requirement related to efficiency may also be referred to usability. So, It will be labeled for both NFR classes. This minimizes the chances of mislabeling. Three annotators perform specification to the given dataset based on the guidelines and instruction [36].

The selection of annotators, annotation tool, and annotation scheme have all been performed through the MATER-MAMA framework under the supervision of the researcher. Step 1 delivers the outcome in the form of the multi-labeled corpus.

3.2 Step 2: Feature Extraction and CNN Training

Step 2 involves designing a CNN for feature extraction and classification. This step takes the labeled corpus from the previous step, extracts features on word2vec, and trains a word-level CNN for multi-label text classification that uses libraries like TensorFlow and Keras. These libraries provide a set of tools for building neural network architecture.

Whereas, the use of word2vec for word embeddings finds the semantic dependency and relatedness of the words in a requirement to determine class label(s). The CNN learns and generates dense vectors for word representation [37], which assumes that words with similar meanings occur in similar contexts [38]. Hence, predicts the labels with improved fit criteria for classification.

4 Conclusion and Future Work

This study provides a good starting point for handling misclassification with regards to NFRs. In this work, we have proposed a CNN based multi-label classifier to automatically classify stakeholder requirements into five classes of NFRs: reliability, usability, portability, maintainability, and efficiency.

The adoption of the proposed tool will aid in manual effort and time invested in the management of NFRs. It will also result in practical benefits to stakeholders. The identification of hidden requirements, the reduction of misclassification, and timely detection of NFRs dependencies and conflicts will benefit stakeholders in the prioritization of requirements and management of resources.

In a prospective course of action, we will bring the design and implementation details of the following: (1) a labeled corpus for NFRs and (2) a multi-label CNN based classifier trained on a representative and balanced corpus for improved training of the classifier on the basis of a broad ontology of NFR domain.

5 Bibliography

1. Ibrahim, N., Wan Kadir, W.M.N., Deris, S.: Documenting requirements specifications using natural language requirements boilerplates. In: 2014 8th. Malaysian Software Engineering Conference (MySEC). pp. 19–24. IEEE, Langkawi, Malaysia (2014)
2. Khatter, K., Kalia, A.: Impact of Non-functional Requirements on Requirements Evolution. In: 2013 6th International Conference on Emerging Trends in Engineering and Technology. pp. 61–68. IEEE, Nagpur, India (2013)
3. Mijanur Rahman, Md., Ripon, S.: Elicitation and Modeling Non-Functional Requirements – A POS Case Study. International Journal of Future Computer and Communication. 485–489 (2013). https://doi.org/10.7763/IJFCC.2013.V2.211
4. Hussain, A., Mkpojiogu, E.O.C., Kamal, F.M.: The Role of Requirements in the Success or Failure of Software Projects. 6, 7 (2016)
5. Younas, M., Wakil, K., N., D., Arif, M., Mustafa, A.: An Automated Approach for Identification of Non-Functional Requirements using Word2Vec Model. International Journal of Advanced Computer Science and Applications. 10, (2019). https://doi.org/10.14569/IJACSA.2019.0100871
6. Glinz, M.: On Non-Functional Requirements. In: 15th IEEE International Requirements Engineering Conference (RE 2007). pp. 21–26. IEEE, Delhi (2007)
7. Berntsson Svensson, R., Gorschek, T., Regnell, B.: Quality Requirements in Practice: An Interview Study in Requirements Engineering for Embedded Systems. In: Proceedings of the 15th International Working Conference on Requirements Engineering: Foundation for Software Quality. pp. 218–232. Springer-Verlag, Berlin, Heidelberg (2009)
8. Zhang, M.-L., Zhou, Z.-H.: A Review on Multi-Label Learning Algorithms. IEEE Transactions on Knowledge and Data Engineering. 26, 1819–1837 (2014). https://doi.org/10.1109/TKDE.2013.39
9. Helmy, W., Kamel, A., Hegazy, O.: Requirements Engineering Methodology in Agile Environment. 9, 8 (2012)
10. Khan, F., Jan, S.R., Tahir, M., Khan, S., Ullah, F.: Survey: Dealing Non-Functional Requirements at Architecture Level. VFAST Transactions on Software Engineering. 9, 7 (2016). https://doi.org/10.21015/vtse.v9i2.410
11. Binkhonain, M., Zhao, L.: A review of machine learning algorithms for identification and classification of non-functional requirements. Expert Systems with Applications: X. 1, 100001 (2019). https://doi.org/10.1016/j.eswax.2019.100001
12. Gries, S.Th., Berez, A.L.: Linguistic Annotation in/for Corpus Linguistics. In: Ide, N. and Pustejovsky, J. (eds.) Handbook of Linguistic Annotation. pp. 379–409. Springer Netherlands, Dordrecht (2017)

13. Robinson, W.N.: Two Rule-Based Natural Language Strategies for Requirements Discovery and Classification in Open Source Software ... Two Rule-Based Natural Language Strategies for Requirements Discovery and Classification in Open Source Software Development Projects. (2012). https://doi.org/10.2753/MIS0742-1222280402
14. Sharma, V.S., Ramnani, R.R., Sengupta, S.: A framework for identifying and analyzing non- functional requirements from text A Framework for Identifying and Analyzing Non-functional Requirements from Text. (2014). https://doi.org/10.1145/2593861.2593862
15. Slankas, J., Williams, L.: Automated extraction of non-functional requirements in available documentation. 2013 1st International Workshop on Natural Language Analysis in Software Engineering, NaturaLiSE 2013 - Proceedings. 9–16 (2013). https://doi.org/10.1109/NAturaLiSE.2013.6611715
16. Abad, Z.S.H., Karras, O., Ghazi, P., Glinz, M., Ruhe, G., Schneider, K.: What Works Better? A Study of Classifying Requirements. In: 2017 IEEE 25th International Requirements Engineering Conference (RE). pp. 496–501. IEEE, Lisbon, Portugal (2017)
17. Sunner, D., Bajaj, H.: Classification of Functional and Non-functional Requirements in Agile by Cluster Neuro-Genetic Approach. International Journal of Software Engineering and Its Applications. 10, 129–138 (2016). https://doi.org/10.14257/ijseia.2016.10.10.13
18. Casamayor, A., Godoy, D., Campo, M.: Identification of non-functional requirements in textual specifications: A semi-supervised learning approach. Information and Software Technology. 52, 436–445 (2010). https://doi.org/10.1016/j.infsof.2009.10.010
19. Hussain, I., Kosseim, L., Ormandjieva, O.: Using Linguistic Knowledge to Classify Non-functional Requirements in SRS documents. In: Kapetanios, E., Sugumaran, V., and Spiliopoulou, M. (eds.) Natural Language and Information Systems. pp. 287–298. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
20. Mahmoud, M.: Software Requirements Classification using Natural Language Processing and SVD. International Journal of Computer Applications. 164, 7–12 (2017). https://doi.org/10.5120/ijca2017913555
21. Cleland-Huang, J., Settimi, R., Zou, X., Solc, P.: Automated classification of non-functional requirements. Requirements Engineering. 12, 103–120 (2007). https://doi.org/10.1007/s00766-007-0045-1
22. Tsai, C.-F.: Bag-of-Words Representation in Image Annotation: A Review. ISRN Artificial Intelligence. 2012, 1–19 (2012). https://doi.org/10.5402/2012/376804
23. Zhang, W., Yang, Y., Wang, Q., Shu, F.: An empirical study on classification of non-functional requirements. … of the 23rd International Conference on …. (2011)
24. Chong, T.Y., Banchs, R.E., Chng, E.S., Li, H.: Modeling of term-distance and term-occurrence information for improving n-gram language model performance. 5

25. Lu, M., Liang, P.: Automatic Classification of Non-Functional Requirements from Augmented App User Reviews. In: Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering - EASE'17. pp. 344–353. ACM Press, Karlskrona, Sweden (2017)
26. Kurtanovic, Z., Maalej, W.: Automatically Classifying Functional and Non-functional Requirements Using Supervised Machine Learning. In: 2017 IEEE 25th International Requirements Engineering Conference (RE). pp. 490–495. IEEE, Lisbon, Portugal (2017)
27. Baker, C., Deng, L., Chakraborty, S., Dehlinger, J.: Automatic Multi-class Non-Functional Software Requirements Classification Using Neural Networks. In: 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC). pp. 610–615. IEEE, Milwaukee, WI, USA (2019)
28. Roh, Y., Heo, G., Whang, S.E.: A Survey on Data Collection for Machine Learning: a Big Data - AI Integration Perspective. arXiv:1811.03402 [cs, stat]. (2018)
29. Tsoumakas, G., Katakis, I.: Multi-Label Classification: An Overview. International Journal of Data Warehousing and Mining. 3, 1–13 (2007). https://doi.org/10.4018/jdwm.2007070101
30. Cleland-Huang, J., Settimi, R., Xuchang Zou, Solc, P.: The Detection and Classification of Non-Functional Requirements with Application to Early Aspects. In: 14th IEEE International Requirements Engineering Conference (RE'06). pp. 39–48. IEEE, Minneapolis/St. Paul, MN (2006)
31. Ormandjieva, O.: Ontology-Based Classification of Non-Functional Requirements in Software Specifications : A new Corpus and SVM-Based Classifier. (2013). https://doi.org/10.1109/COMPSAC.2013.64
32. Singh, P., Singh, D., Sharma, A.: Rule-based system for automated classification of non-functional requirements from requirement specifications. In: 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI). pp. 620–626. IEEE, Jaipur, India (2016)
33. Mahmoud, A., Williams, G.: Detecting, classifying, and tracing non-functional software requirements. Requirements Engineering. 21, 357–381 (2016). https://doi.org/10.1007/s00766-016-0252-8
34. P. Miguel, J., Mauricio, D., Rodríguez, G.: A Review of Software Quality Models for the Evaluation of Software Products. International Journal of Software Engineering & Applications. 5, 31–53 (2014). https://doi.org/10.5121/ijsea.2014.5603
35. Pustejovsky, J., Stubbs, A.: Natural language annotation for machine learning. O'Reilly Media, Sebastopol, CA (2013)
36. Hinze, A., Heese, R., Luczak-Rösch, M., Paschke, A.: Semantic Enrichment by Non-experts: Usability of Manual Annotation Tools. In: Cudré-Mauroux, P., Heflin, J., Sirin, E., Tudorache, T., Euzenat, J., Hauswirth, M., Parreira, J.X., Hendler, J., Schreiber, G., Bernstein, A., and Blomqvist, E. (eds.) The Semantic Web – ISWC 2012. pp. 165–181. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)

37. Nematzadeh, A., Meylan, S.C., Griffiths, T.L.: Evaluating Vector-Space Models of Word Representation, or, The Unreasonable Effectiveness of Counting Words Near Other Words. 6 (2017)
38. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space. arXiv:1301.3781 [cs]. (2013)