Semi-automated Software Requirements Categorisation using Machine Learning Algorithms

Original Scientific Paper

Pratvina Talele
Department of Computer Engineering and Technology,
Dr. Vishwanath Karad MIT World Peace University,
Pune, India
pratvina.talele@mitwpu.edu.in

Siddharth Apte
Department of Computer Engineering and Technology,
Dr. Vishwanath Karad MIT World Peace University,
Pune, India
sidapte.01@gmail.com

Rashmi Phalnikar
Department of Computer Engineering and Technology,
Dr. Vishwanath Karad MIT World Peace University,
Pune, India
rashmi.phalnikar@mitwpu.edu.in

Harsha Talele
Department of Computer Engineering,
Pimpri Chinchwad College of Engineering,
Pune, India
harshatalele19@gmail.com

Abstract *– Requirement engineering is a mandatory phase of the Software development life cycle (SDLC) that includes defining and documenting system requirements in the Software Requirements Specification (SRS). As the complexity increases, it becomes difficult to categorise the requirements into functional and non-functional requirements. Presently, the dearth of automated techniques necessitates reliance on labour-intensive and time-consuming manual methods for this purpose. This research endeavours to address this gap by investigating and contrasting two prominent feature extraction techniques and their efficacy in automating the classification of requirements. Natural language processing methods are used in the text pre-processing phase, followed by the Term Frequency – Inverse Document Frequency (TF-IDF) and Word2Vec for feature extraction for further understanding. These features are used as input to the Machine Learning algorithms. This study compares existing machine learning algorithms and discusses their correctness in categorising the software requirements. In our study, we have assessed the algorithms Decision Tree (DT), Random Forest (RF), Logistic Regression (LR), Neural Network (NN), K-Nearest Neighbour (KNN) and Support Vector Machine (SVM) on the precision and accuracy parameters. The results obtained in this study showed that the TF-IDF feature selection algorithm performed better in categorising requirements than the Word2Vec algorithm, with an accuracy of 91.20% for the Support Vector Machine (SVM) and Random Forest algorithm as compared to 87.36% for the SVM algorithm. A 3.84% difference is seen between the two when applied to the publicly available PURE dataset. We believe these results will aid developers in building products that aid in requirement engineering.*

Keywords*: Natural Language Processing, Machine Learning, Software Engineering, Supervised Machine Learning*

1. INTRODUCTION

In system engineering, the software development life cycle (SDLC) or application development life cycle is a process utilised to plan, test, and deploy software projects. The SDLC is a standardised procedure that tries to guarantee the accuracy of the shipped software in accordance with the client's requirements. A crucial stage in SDLC is requirement engineering, which involves establishing, documenting, and managing system requirements [1]. Requirement elicitation, which is a component of requirement engineering, is the process of collecting software requirements by communicating with stakeholders. The methods used for requirement elicitation include task analysis, interviews, and brainstorming.

There are two main types of requirements: Functional and Non-Functional. Functional requirements are mandatory requirements that must be implemented in the software system. It defines what a system must do and its features and functions. Non-functional requirements are not mandatory but desirable attributes that specify how a system should function. They can be considered as quality attributes of the software systems.

Another aspect of requirement engineering is requirement prioritisation. After identifying functional and non-functional requirements, priorities are to be assigned to streamline and order the development of software. Requirement prioritisation is a continuous process encompassing multifaceted and critical decision-making events that simplify high-quality software development. This process ensures that the ordering of both functional and non-functional requirements specified by software stakeholders is correct for implementation [2].

These requirements are ordered as per their significance, such as conflict, penalty, and price [3]. Given the rising demand for software functionality, many software solutions have a wide range of requirements. Implementing various requirements in a limited amount of time with limited availability of resources and budget is challenging. Hence, software products are delivered in stages. With each new release, features can be added. However, this prioritisation of shipping the critical requirements earliest and proceeding with the less crucial aspects makes requirement engineering a challenging process. Accurate requirement classification is critical to a software project's success. There are significant risks associated with the incorrect classification of requirements [4]. These risks can entail the project being over budget, going overtime, or even failing. According to research, 71% of errors result from an unclear understanding and classification of requirements [5]. The CHAOS report by the Standish Group outlined the causes of IT project failure in the United States. It revealed that just 16.2% of IT projects were successful; the rest failed [6]. The current manual methods for classifying software requirements are labour and time-intensive and require subject-matter specialists. Additionally, manual techniques can also lead to inaccuracy and misrepresentation. Our purpose in undertaking this research is to provide a modern automated solution that will aid the process of requirement engineering by improving accuracy while reducing the time taken.

Significant research is being conducted in the field of applying machine learning models and techniques in classifying text [7] from documents and other natural language processing techniques. Based on several studies [3, 8, 9] and comparative analysis, in this paper, TF-IDF and Word2Vec techniques have been employed for feature selection and several machine learning algorithms, which are Decision tree, Random Forest, Linear regression, Neural network, KNN (K – nearest neighbours), SVM (Support vector machines) for text categorisation.

This paper is organised as follows: The literature review is covered in section two. The research methodology and techniques employed are thoroughly described in section three. Experimental results are discussed in section four. In section five, the conclusion is presented, and future scope is discussed in section six.

2. LITERATURE REVIEW

Using machine learning, researchers have employed many techniques to classify software requirements (functional and nonfunctional). The j48 decision tree is implemented with pruned and unpruned being the two types of trees to create a total of four models; model 1 handles 'authentication authorization' type of security requirements, model 2 deals with 'access control' requirements, model 3 defines 'cryptography-encryption' requirements and model 4 is concerned with 'data integrity' requirements. Out of which, model 4 gives the maximum accuracy [10]. The use of algorithms such as MNB, Gaussian Naïve Bayes (GNB), KNN, Decision tree, Support Vector Machine (SVM), and Stochastic Gradient Descent SVM (SVM SGD) was done. Of these, SVM and SGD performed better than all the algorithms with GNB performing the worst [11]. The author has used KNN, SVM, Logistic regression, and MNB. SVM and Linear regression both have shown high precision values in classifications of functional and non-functional requirements [12]. In this paper, the authors have used Multinomial Naïve Bayes, KNN, Sequential Minimal Optimization (SMO). SMO performed the best with an accuracy of 0.729 [13]. A series of support vector machine (SVM) is used along with lexical features to achieve recall and precision on 0.92 [10]. It is observed that the modified version of decision tree, Multiple Correlation based decision tree, used to classify software requirements performs better when using TF-IDF as a feature extraction method [14]. MNB and Logistic regression are employed to obtain an accuracy of 95.55% and 91.23% respectively [15]. Naïve Bayes and decision tree were employed on a dataset comprising crowdsourced software requirements with accuracy results for Naïve Bayes being 92.7% and Decision tree being 84.2% [16]. The study investigated two types of models CNN (convolutional neural network) and ANN (Artificial neural network) by varying the hyperparameters i.e., Training epoch, Batch size, Number of filters, Embedding Dimension for CNN and Training Epochs, Alpha, Hidden Neurons for ANN having achieved an accuracy of 82% to 90% for the ANN model and 82% to 94% for the CNN models [17]. A semi-supervised learning approach is described for NFR classification. This approach with EM strategy (expectation maximisation) shows an improvement in classification accuracy as compared to supervised naïve bayes, KNN and TF-IDF in terms of its accuracy. In accordance with the percentage increase of labelled requirements, the accuracy achieved ranges from 80% to higher 90's [18]. In this approach, the BoW (bag of words) is used as a text vectorization technique in combination with two machine learning algorithms, SVM and KNN. The results are presented in three types. Firstly, the classification of FR/NFR having a precision of 90% and 82%, secondly, the classification of subcategories of NFR with an accuracy of 68% and 56% and lastly, the classification of subcategories of NFR and FR with a precision of 73% and 67% respectively, for SVM and KNN algorithms [19]. Researchers applied the TF-IDF

preprocessing technique to a dataset containing SRS document phrases with functional and non-functional requirements and were able to achieve an accuracy of 78.57% for the SVM algorithm whereas decision tree performed the worst with an accuracy of 61.42% [20].

The studies mentioned had to preprocess and extract important features to manually achieve this goal. With the rise of powerful deep learning algorithms, cheaper computational power, and other advancements facilitating the use of such algorithms, it was observed that end-to-end methods could be built to do this work. The authors in [21] employed the use of Bidirectional Gated Recurrent Neural Networks (BiGRU) to classify requirements using raw text, using word and character sequence as tokens. The accuracy range for the Word sequence ranged from 76% to 87% whereas for the character sequence ranged from 56% to 70% for binary classification and multiclass multilabel classification. The results were 69% for word vector and 64% for character sequence. The authors have examined the possibility of combining standard feature selection with 9 different kinds of deep learning predictors. They have used different types of Max–Min ratio (MMR) approaches, which have shown an improvement in performance by 1 to 5% when tested on the PROMISE dataset [22].

The research gap that we identified in the reviewed publications is that they do not consider all types of software requirements (functional/non-functional) [10, 11, 13] as well as a wide array of existing machine learning algorithms have not been evaluated [10, 15, 16, 19]. To overcome this drawback, machine learning algorithms such as DT, RF, LR, NN, KNN, and SVM are considered, and feature extraction techniques, TF-IDF and Word2Vec are evaluated in this study. This will assist researchers in developing the software framework.

3. METHODOLOGY

To classify software requirements, many researchers have suggested various requirement classification strategies. Since each stakeholder has a unique viewpoint on the software, this process takes a long time to complete. This calls for automating the requirement identification process. Different machine learning (ML) techniques are utilised to automate the requirements classification. As these requirements are stated in natural language, the text pre-processing phase is necessary to turn the text into vectors. These vectors can be used as input for ML algorithms. The steps in the conduct of this research are mentioned in Fig.1.

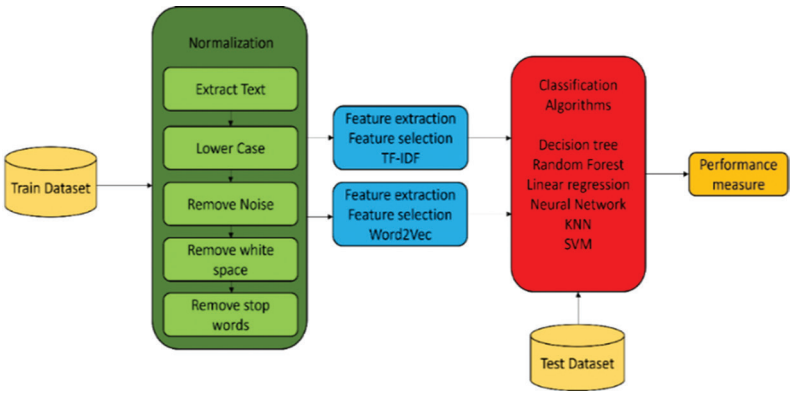

Fig. 1. Proposed Architecture for Automation of Classifying Requirements

3.1. TEXT PREPROCESSING

Natural language processing (NLP) is a branch of Machine learning that deals with teaching computers to process data that is specified in human-readable language. The raw text is cleaned by feature extraction and text preprocessing techniques, which then supply the features to machine learning algorithms. Training on supervised learning algorithms becomes easier when raw text is processed. The text preprocessing methods are as follows:

- Extract text: This method uses Python libraries to extract the text from documents in PDF or CSV formats to give the outputs in paragraphs.
- Lowercase: All uppercase characters are converted to lowercase, as having some uppercase characters might lead to incorrect feature extraction caused by duplication.
- Removal of noise: meaningless symbols that do not carry any specific meaning, such as ! $@$%&*(^$#@ {}[]:"><,.}]| are removed.
- Removal of white space: extra white space occurring between two words is removed for preprocessing.
- Removal of stop words: certain words in grammar such as "the", "a", "on", "is", "all" do not carry any information of significance and hence are removed.

3.2. EXTRACTION TECHNIQUE

Based on the literature survey, two techniques, Term Frequency – Inverse Document Frequency and Word-2Vec are used to extract features.

Term Frequency – Inverse Document Frequency (TF-IDF) – is a method to compute words in a set of documents. It quantifies the frequency of a term in a

document while considering its rarity across the entire collection, aiding in information retrieval and text analysis. It is calculated by:

$$tfidf_{w,d} = tf_{w,d} \times idf_w \quad (1)$$

where Term Frequency (TF) is simply a frequency counter for a word (w) in document(d).

$$tf_{w,d} = \frac{n_{w,d}}{Number\ of\ words\ in\ the\ document} \quad (2)$$

Inverse Document Frequency measures the informativeness of word (w). It is calculated by

$$idf_w = log\left(\frac{number\ of\ documents}{number\ of\ documents\ with\ word\ w}\right) \quad (3)$$

Word2Vec – Word2Vec's objective is to align the vectors of words with similar meanings in the vector space. It searches for mathematical similarities. Word2Vec automatically distributes numerically and vectorizes word characteristics, such as the context of specific words.

3.3. CLASSIFICATION ALGORITHMS

- Decision Tree (DT): A Decision Tree is a non-parametric algorithm for classification and regression. It organises data into a tree-like flowchart, using if-else tests on features to recursively partition the dataset. The leaves of the tree provide final predictions. Measures like Entropy, Gini Index, and Chi-Square Test guide the selection of important features, aiding in decision-making and prediction accuracy.
- Random Forest (RF): It is an ensemble method, which means that a random forest model is composed of numerous decision trees, known as estimators, each of which generates a separate set of predictions. The estimators' predictions are combined by the random forest model to yield a more precise prediction.
- Logistic Regression (LR): It is an ML Algorithm used for binary classification by predicting whether something happens or not. It uses a Sigmoid function (Logistic function) to give the probability output which is then compared to a pre-defined threshold and further labelled into one of the two categories accordingly.
- Neural Network (NN): It is an ML Algorithm used to diagnose the relationships in a set of features through a process inspired by the behaviour of the human brain. Several nodes comprise a neural network layer that can have multiple layers.
- K-nearest neighbours (KNN): It is a type of supervised learning classifier that groups individual data points together which are in close proximity (k closest relatives). It assumes that points lying near one another are similar. The class label is decided based on the class labels assigned to the surrounding points; this is referred to as "majority voting". Its performance is lower than SVM and Multinomial Naïve Bayes.
- Support Vector Machines (SVM): are a set of supervised machine learning algorithms employed for classification and regression. In an n-dimensional space, where n represents the number of features, data items are plotted, and a hyperplane is identified. This hyperplane effectively separates the coordinates into positive and negative classes, classifying the data items. SVM is less suitable for large datasets due to extended training times.

4. EXPERIMENTAL RESULTS

To the best of our knowledge, a comparison between Word2Vec and TF-IDF feature extraction techniques has yet to be performed on the PURE dataset.

4.1. EXPERIMENTAL SETTINGS

These experiments were carried out on a local computer running 64-bit Windows 11 equipped with 8GB RAM. These experiments were carried out on a local Jupyter Notebook environment with the Python3 programming language.

4.2. DESCRIPTION OF THE DATASET

Table 1. Categories of Requirements

PURE dataset [23]			
1.	Functional Requirement (FR)	11.	Quality attribute requirements
2.	Qualitative attributes	12.	Availability requirement
3.	Help module	13.	Maintainability requirement
4.	Support module	14.	Other requirements
5.	Audit module	15.	Safety requirement / Security requirement
6.	Access module	16.	Usability requirements
7.	Ease of use	17.	Portability requirements
8.	Usability	18.	Testability requirements
9.	System Availability	19.	Other nonfunctional requirements
10.	Performance and scalability / Performance requirement	20.	Software quality attributes

The publicly available dataset PURE (PUblicREquirement) used is a compendium of 79 different documents in PDF and HTML formats. These are SRS (Software Requirement Specification) documents that represent real-world requirements taken from academia, industry, etc., and contain 20 different functional and nonfunctional requirements written in natural language. These documents would be representative of the client's requirements, typically presented to the software developers. Table 1 provides an overview of the number of requirements present in the dataset [23].

4.2. RESULTS

This study uses the PURE dataset, which contains 79 SRS documents. Two text preprocessing techniques,

Word2Vec and TF-IDF, are employed. A total of 2153 feature instances were extracted, representing 20 distinct feature categories within the dataset. The results are as follows. A comparison of the results using the TF-IDF technique and the Word2Vec technique is discussed below.

Using the TF-IDF preprocessing technique, which calculates how relevant a word in a series or corpus is to a text/document [24], in our study, all of the algorithms were able to identify Functional, Access module, Maintainability requirements, other requirements, Performance requirements, software quality attributes, Usability, and software qualitative attributes, as illustrated in Fig. 2.

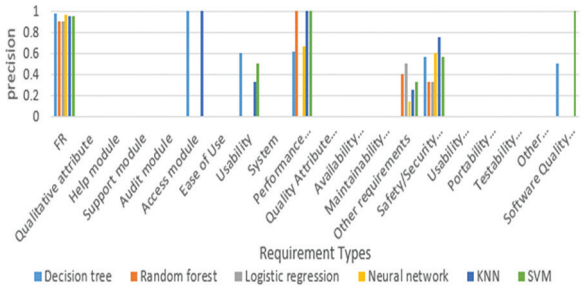

Fig. 2. Comparison of ML Algorithms Using TF-IDF (Precision)

Only Decision Tree was able to identify the Audit Module, and only Neural Network was able to identify Portability Requirements. The Help module, support module, Ease of use, Availability, Usability requirements, Testability, and Other nonfunctional requirements were all not identified by any of the algorithms. However, some algorithms, while able to correctly identify the requirements, are not in all cases able to correctly map those requirements to the corresponding test dataset, as shown in Table 2. '0' indicates that it is identified but not mapped, whereas '-' indicates it is not identified at all.

Fig. 3 shows the accuracy technique for the various algorithms. SVM (support vector machine) performs the best with an accuracy of 91.20%, followed by 90.10% for the KNN algorithm. Decision Tree, Logistic Regression, Random Forest, and Neural Network all have accuracies of 88.46%, 87.91%, 87.36%, and 84.06% respectively.

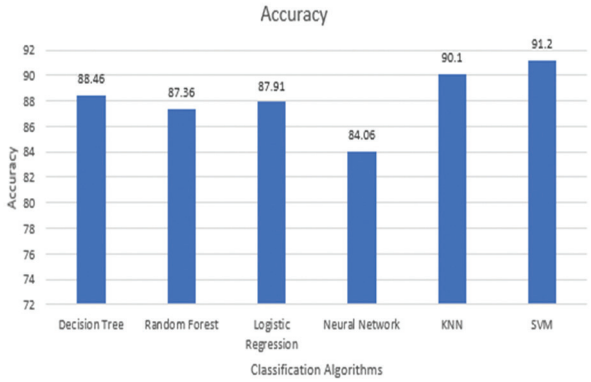

Fig. 3. Comparison of ML Algorithms Using TF-IDF (Accuracy)

Table 2. Comparison of Precision Value for ML Algorithms Using TF-IDF

	Precision									
ML algorithm	FR	Qualitative attribute	Help module	Support module	Audit module	Access module	Ease of Use	Usability	System Availability	Performance Requirements
Decision tree	0.98	0	-	-	0	1	-	0.60	0	0.62
Random forest	0.91	0	-	-	-	0	-	0	-	1
Logistic regression	0.90	0	-	-	-	0	-	0	-	0
Neural network	0.97	0	-	-	-	0	-	0	-	0.67
KNN	0.95	0	-	-	-	1	-	0.33	-	1
SVM	0.95	0	-	-	-	0	-	0.50	-	1

ML algorithm	Quality Attribute Requirements	Availability Requirements	Maintainability Requirements	Other Requirements	Safety/Security Requirements	Usability Requirements	Portability Requirements	Testability Requirements	Other Nonfunctional Requirements	Software Quality Attributes
Decision tree	-	-	0	0	0.57	-	-	-	-	0.5
Random forest	-	-	0	0.4	0.33	-	-	-	-	0
Logistic regression	-	-	0	0.5	0.33	-	-	-	-	0
Neural network	0	-	0	0.14	0.60	-	0	-	-	0
KNN	-	-	0	0.25	0.75	-	-	-	-	0
SVM	-	-	0	0.33	0.57	-	-	-	-	1

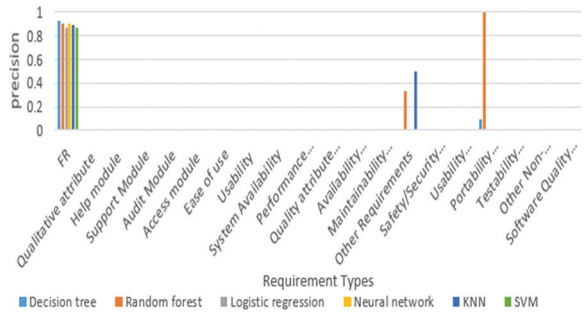

Fig. 4. Comparison of ML Algorithms Using Word2Vec (Precision)

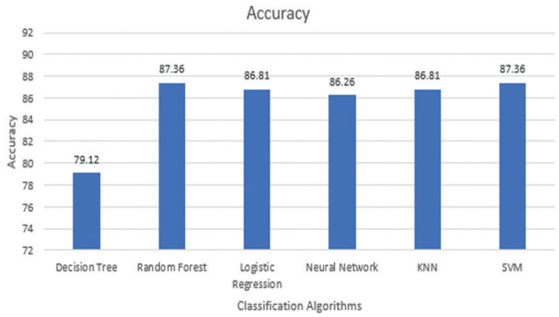

Fig. 5. Comparison of ML Algorithms Using Word2Vec (Accuracy)

	Precision									
ML algorithm	**FR**	**Qualitative attribute**	**Help module**	**Support module**	**Audit module**	**Access module**	**Ease of Use**	**Usability**	**System Availability**	**Performance Requirements**
Decision tree	0.93	0	0	-	-	0	-	0	-	-
Random forest	0.90	0	0	-	-	0	-	0	-	-
Logistic regression	0.87	0	-	-	-	0	-	0	-	-
Neural network	0.90	0	-	-	-	0	-	0	-	-
KNN	0.89	0	-	-	-	0	-	0	-	-
SVM	0.87	0	-	-	-	0	-	0	-	-

ML algorithm	**Quality Attribute Requirements**	**Availability Requirements**	**Maintainability Requirements**	**Other Requirements**	**Safety/Security Requirements**	**Usability Requirements**	**Portability Requirements**	**Testability Requirements**	**Other Nonfunctional Requirements**	**Software Quality Attributes**
Decision tree	-	0	0	0	0	0	0.1	-	0	0
Random forest	-	0	0	0.33	0	0	1	-	0	0
Logistic regression	-	-	0	0	0	0	0	-	0	0
Neural network	-	-	0	0	0	0	0	-	0	0
KNN	-	-	0	0.5	0	0	0	-	0	0
SVM	-	-	0	0	0	0	0	-	0	0

The Word2Vec learns representations by predicting word contexts in a given dataset. This enables efficient encoding of semantic similarities and differences, facilitating tasks like natural language processing and machine learning [25]. In our research, all the algorithms were able to identify Functional requirements, Qualitative attributes, Access module, Usability, Maintainability, other requirement, Safety/Security Requirements, Usability requirements, Portability, other nonfunctional requirements, and software quality attributes. Only random forest and decision tree can classify the Help module and Availability requirements; all other requirements were unidentified by any of the algorithms. However, some algorithms, while able to correctly identify the requirements, are not in all cases able to correctly map those requirements to the corresponding test dataset, as illustrated in Table 3. '0' indicates that it is identified but not mapped whereas '-' indicates it is not identified at all.

5. CONCLUSION

Through this work, we have provided a systematic comparison of several ML algorithms (DT, Random Forest, LR, NN, KNN, and SVM) used to identify and categorise a number of functional and non-functional requirements. The dataset undergoes preprocessing which includes extracting the text, conversion to lowercase, removal of noise, white space, and stop words after which the feature selection technique of TF-IDF and Word2Vec is employed. The performance of the ML algorithms is measured by the test dataset.

79 separate SRS documents in PDF and HTML formats that depict actual client requests were employed in the second phase of our research. In this, the feature selection techniques are compared while the pre-processing methods stay the same. The study was carried out using the Word2Vec and TF-IDF approaches. When

compared to the Word2Vec technique, it has been found that the TF-IDF feature selection technique produces better outcomes in subsequent algorithmic evaluations. 91.20% (for the SVM and Random Forest algorithm) as compared to 87.36% (for the SVM algorithm), a 3.84% difference is seen between the two. We are sure that these techniques will assist developers in automating the task of software requirement categorisation, thus saving time, money, and other vital resources in the SDLC while being able to ship the most suitable product tailored to the stakeholder's requirements. Additionally, it can assist industry experts in choosing the appropriate algorithm that offers the most significant degree of accuracy in the categorisation process.

6. FUTURE SCOPE

To facilitate the widespread adoption of ML algorithms in requirement engineering, the future scope entails extending the research to ensure practical applicability and scalability. The focus should be on optimising the proposed methods for large-scale projects. Additionally, there is a need to explore integration with popular development tools, thereby facilitating adoption by development teams and streamlining the software engineering process.

7. REFERENCES:

[1] P. Talele, R. Phalnikar, "Software requirements classification and prioritization using machine learning", Machine Learning for Predictive Analysis, Lecture Notes in Networks and Systems, Springer, Vol. 141, 2021, pp. 257-267.

[2] P. Talele, R. Phalnikar, "Automated Requirement Prioritisation Technique Using an Updated Adam Optimisation Algorithm", International Journal of Intelligent Systems and Applications in Engineering, Vol. 11, No. 3, 2023, pp. 1211-1221.

[3] R. Phalnikar, D. Jinwala, "Analysis of Conflicting User Requirements in Web Applications Using Graph Transformation", ACM SIGSOFT Software Engineering Notes, Vol. 40, No. 2, 2015, pp. 1-7.

[4] H. Alrumaih, A. Mirza, H. Alsalamah, "Toward Automated Software Requirements Classification", Proceedings of the 21st Saudi Computer Society National Computer Conference, Riyadh, Saudi Arabia, 25-26 April 2018, pp. 1-6.

[5] "Fixing the Software Requirements Mess", https://www.cio.com/article/255253/developer-fixing-the-software-requirements-mess.html (accessed: 2023)

[6] The Standish Group Report, "Chaos report", https://simpleisbetterthancomplex.com/media/2016/10/chaos-report.pdf (accessed: 2023)

[7] R. Pawar, S. Ghumbre, R. Deshmukh, "Developing an Improvised E-Menu Recommendation System for Customer", Recent Findings in Intelligent Computing Techniques, Advances in Intelligent Systems and Computing, Vol. 708, Springer, Singapore.

[8] M. Binkhonain, L. Zhao, "A review of machine learning algorithms for identification and classification of non-functional requirements", Expert Systems with Applications: X, Vol. 1, 2019.

[9] A. Khan, B. Baharudin, L. H. Lee, K. Khan, "A Review of Machine Learning Algorithms for Text-Documents Classification", Journal of Advances in Information Technology, Vol. 1, No. 1, 2010, pp. 4-20.

[10] R. Jindal, R. Malhotra, A. Jain, "Automated classification of security requirements", Proceedings of the International Conference on Advances in Computing, Communications and Informatics, Jaipur, India, 21-24 September 2016, pp. 2027-2033.

[11] M. A. Haque, M. Abdur Rahman, M. S. Siddik, "Non-Functional Requirements Classification with Feature Extraction and Machine Learning: An Empirical Study", Proceedings of the 1st International Conference on Advances in Science, Engineering and Robotics Technology, Dhaka, Bangladesh, 3-5 May 2019, pp. 1-5.

[12] E. D. Canedo, B. C. Mendes, "Software Requirements Classification Using Machine Learning Algorithms", Entropy, Vol. 22, No. 9, 2020 p. 1057.

[13] P. Talele, R. Phalnikar, "Classification and Prioritisation of Software Requirements using Machine Learning – A Systematic Review", Proceedings of the 11th International Conference on Cloud Computing, Data Science & Engineering, Noida, India, 28-29 January 2021, pp. 912-918.

[14] P. Talele, R. Phalnikar, "Multiple correlation based decision tree model for classification of software requirements", International Journal of Computational Science and Engineering, Vol. 26, No. 3, 2023, pp. 305-315.

[15] A. A. A. Althanoon, Y. S. Younis, "Supporting Classification of Software Requirements system Using Intelligent Technologies Algorithms", Technium, Vol. 3, No. 11, 2021, pp. 32-39.

[16] S. Taj, Q. Arain, I. Memon, A Zubedi. "To apply Data Mining for Classification of Crowd sourced Software Requirements", Proceedings of the 8th International Conference on Software and Information Engineering, New York, NY, USA, 2019, pp. 42-46.

[17] C. Baker, L. Deng, S. Chakraborty, J. Dehlinger, "Automatic Multi-class Non-Functional Software Requirements Classification Using Neural Networks", Proceedings of the IEEE 43rd Annual Computer Software and Applications Conference, Milwaukee, WI, USA, 15-19 July 2019, pp. 610-615.

[18] A. Casamayor, D. Godoy, M. Campo, "Identification of non-functional requirements in textual specifications: A semi-supervised learning approach", Information and Software Technology, Vol 52, No. 4, 2010, pp. 436-445.

[19] G. Y. Quba, H. Al Qaisi, A. Althunibat, S. AlZu'bi, "Software Requirements Classification using Machine Learning algorithm's", Proceedings of the International Conference on Information Technology, Amman, Jordan, 14-15 July 2021, pp. 685-690.

[20] S. Apte, Y. Honrao, R. Shinde, P. Talele, R. Phalnikar, "Automatic Extraction of Software Requirements Using Machine Learning", ICT with Intelligent Applications, Lecture Notes in Networks and Systems, Vol. 719, Springer, Singapore.

[21] O. Al Dhafer, I. Ahmad, S. Mahmood, "An end-to-end deep learning system for requirements classification using recurrent neural networks", Information and Software Technology, Vol. 147, 2022, p. 106877.

[22] S. Saleem, M. N. Asim, L. Van Elst, A. Dengel, "FNReq-Net: A hybrid computational framework for functional and non-functional requirements classification", Journal of King Saud University - Computer and Information Sciences, Vol. 35, No. 8, 2023.

[23] "Natural Language Requirements Dataset", http://nlreqdataset.isti.cnr.it/ (accessed: 2023)

[24] Y. Xu, C. Zhang, W. Song, "Prioritizing Customer Requirements for Science and Technology Service Platform Based on Improved TF-IDF and Sentiment Analysis", Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management, Kuala Lumpur, Malaysia, 7-10 December 2022, pp. 210-214.

[25] S. J. Putra, M. N. Gunawan, A. A. Hidayat, "Feature Engineering with Word2vec on Text Classification Using The K-Nearest Neighbor Algorithm", Proceedings of the 10th International Conference on Cyber and IT Service Management, Yogyakarta, Indonesia, 20-21 September 2022, pp. 1-6.