An Automated Approach for Identification of Non-Functional Requirements using Word2Vec Model

Muhammad Younas*[1]
Department of Computer Science
Government College University Faisalabad
Allama Iqbal Road Faisalabad, Pakistan 38000

Karzan Wakil[2]
Research Center
Sulaimani Polytechnic University
Sulaimani, Kurdistan Region, 46001, Iraq

Dayang N. A. Jawawi[3], Muhammad Arif Shah[4], Ahmad Mustafa[5]
School of Computing
Faculty of Engineering, Universiti Teknologi Malaysia
Johor Bahru, 81310 Malaysia

***Abstract*—Non-Functional Requirements (NFR) are embedded in functional requirements in requirements specification document. Identification of NFR from the requirement document is a challenging task. Ignorance of NFR identification in early stages of development increase cost and ultimately cause the failure of the system. The aim of this approach is to help the analyst and designers in architect and design of the system by identifying NFR from the requirements document. Several supervised learning-based solutions were reported in the literature. However, for accurate identification of NFR, a significant number of pre-categorized requirements are needed to train supervised text classifiers and system analysts perform the categorization process manually. This study proposed an automated semantic similarity based approach which does not needs pre-categorized requirements for identification of NFR from requirements documents. The approach uses an application of Word2Vec model and popular keywords for identification of NFR. Performance of approach is measured in term of precision-recall and F-measure by applying the approach to PROMISE-NFR dataset. The empirical evidence shows that the automated semi-supervised approach reduces manual human effort in the identification of NFR.**

***Keywords*—Identification; non-functional requirements; semantic similarity; Word2Vec model**

I. Introduction

The success of a software system is based on Functional Requirements (FR) and Non-Functional Requirements (NFR). During the requirement elicitation phase, the primary emphasis is on gathering functional requirements, however, NFRs are overlooked. The nature of agile software methodology is also a reason to ignore NFRs [1]. This ignorance of NFRs becomes a cause to produce significant cost issues in software system [2]. U.S. army's computing system for intelligence sharing with troops fighting in Afghanistan and Iraq with a budget of $2.7 billion was rejected due to the issues such as performance, and usability [3]. Electronic Health Record (EHR) was not adopted by the medical community due to lack of usability [4]. A survey revealed that more than 60% of the projects failed due to ignorance of NFR [5].

The agile software development process is based on human-centric requirement engineering which depends on knowledge of stakeholders' regarding application. in SRS documents, software requirements are represented in natural language [6]. The natural languages produce ambiguity and inconsistency in requirement statements. Therefore, it is hard to model and automate the semantic knowledge into requirement engineering activities such as elicitation [7], analysis [6], traceability [8] and reuse [9]. Furthermore, early identification of NFR is critical in the design and architectural concerns of software [10]. Agile software methodologies support a rapid change in software at any stage of development. Due to this rapid change, the importance of NFR identification approach is increased.

The theme of research is about the identification of NFR by manipulating the textual semantic of FR. domain knowledge, also called vocabulary of the domain and NFR knowledge base helps to identify NFR [6]. Furthermore, the automated approach reduces the human or manual effort in the identification of NFR from the requirements document. The scope of research is to identify the non-functional requirements from natural language based SRS documents.

This automatic approach helps requirement engineers and analysts in identification of NFR from the requirement statements in documents, interview notes, memos and reports. The contribution of paper is as follows:
This study

1) extracts the requirement from the document and identify NFR categories,
2) automates and enhance the NFR identification process through semantic similarity measure and pre-processing methods,
3) uses the Stanford Natural Language Parser (NLP) for automaton of identification process,
4) utilizes Wikipedia dump of data for measuring semantic similarity, and
5) enhances the extraction of NFR in terms of increased

precision, recall and F-measure compared to existing work.

Rest of the paper is organized as follows: Section II addressed the background of the study. Section III presents the automated identification approach. Section IV evaluates the approach. Finally, Section V, VI, VII and VIII describe related work, conclusion, limitations and future work, respectively.

II. Background

A. Non-Functional Requirements

Non Functional Requirements are usually known as "ilities"; or as the quality features or attributes of the system [11]. A study considers NFR as systematic requirements [9]. IEEE requirements practices named NFR as constraints. A study surveyed and identified 156 types of NFR and eight (8) categories [9]. The subcategories of NFR are: (i) access control, (ii) audit, (iii) availability, (iv) capacity and performance, (v) legal, (vi) look and feel, (vii) maintainability, (viii) operational, (ix) privacy, (x) recoverability, (xi) reliability, (xii) security, (xiii) usability, and other.

B. Text Embedding Techniques and Semantic Similarity

Word embedding is a process of mapping words and sentences into vector of real numbers. There are different types of methods for mappings such as neural network, co-occurrence matrix, dimensionality reduction, and probability models.

In text classification, the term frequency-inverse document frequency (TF-IDF) is being used for a long time [32]. TF-IDF is a popular term weighting scheme, and more than 80% of text-based recommender systems in digital libraries use TF-IDF [52]. Term frequency means the occurrence of a term in a document. The TF–IDF gets large value both in the given document and in the all documents because it depend on weights, it filters out common terms. ratio is greater and equal to 1 and the value of IDF and (TF–IDF) is greater than or equal to 0. If a term appears in a greater number of times in a document then the IDF and TF–IDF is near to 0.

In order to calculate semantic similarity between words, corpus-based approach named as Latent Semantic Analysis (LSA) is used. LSA uses the distributional hypothesis in which the words having similar meaning are placed in a similar place in corpus [12]. A paragraph is converted into a (m $\times$ n) matrix, where m (number of rows) represents the unique words and n represents paragraphs. A mathematical technique named as Singular Value Decomposition (SVD) is applied to reduce the number of rows in the matrix. SVD is used in such a way that the similarity structure in columns does not change.

The Text is converted into vector, the value of cosine of the angle between the two vectors is called the similarity between the two words. The cosine value closer to 1 represents the words are similar and a value closer to 0 means not similar [13]. LSA is very popular and has many strong points, however, there are some limitations. The LSA cannot incorporate the polysemy (multiple meaning of the word). Another limitation is that the LSA does not support Bag of Word (BOW). In BOW, the text is treated as an unordered collection of words [14].

Word2Vec a novel word embedding procedure is designed by Mikolov [15]. The Word2Vec model learns Word2Vector representation using the multi-level neural-network model. The proximity or semantic similarity distance between the words is calculated with the help of Word2Vec model which transforms text into vector [16]. Word2Vect is an example of a group of related neural networks to construct the linguistic context of the word. Each word has a unique vector value. In vector space, the vectors are ranked by considering the context of the words. Each vector is trained in such a way to maximize the probability of the neighbouring word in the corpus as expressed in Equation (1).

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{j\in neighb(t)} logp(W_j|W_t) \tag{1}$$

where the list of words is $W = (W_1, W_2, W_3, \ldots, W_t)$, neigb(t) is a set of neighboring word of word W_t and $p(W_j|W_t)$ is the associated word vectors V_{wj} and V_{wt} in hierarchial softmax [15].

Word2Vec is a pre-trained model [17] which is used to find similarity distance between words. Word2Vec model has many advantages on latent semantic analysis [15].

For measuring the performance of similarity measure methods, the similarity distance between word email and words password, color, font, and logon are calculated as shown in Table I. In term of similarity measure, the corpus-based methods (LSA, NGDwiki, and Word2Vecwiki) perform better as compared to the thesaurus-based similarity measure methods (i.e. Wu, Lesk and Resnik) and co-occurrence methods (PMI). In Table I, the similarity word email is calculated with the rest of the words.

A human with little knowledge about web application would rank the words given in Table I. For example, the words password and logon have more similarity than the word color and font with email. Table I shows that thesaurus-based methods perform almost the same. The method Resnik and Lesk have its similarity value in numbers with minimum zero and maximum infinity. The methods Resnik, Wu, and Lesk have a similarity of email with logon is zero, which shows that these methods do not find the word logon in the WordNet dictionary. In thesaurus-based methods, the common problem is the lack of named entities [18]. This problem arises due to the growth of natural languages data and Expert generated linguistic databases such as WordNet cannot keep up the pace of this growth. Therefore, the selection of such type of similarity measuring methods with WordNet like database might lead to outdated and misleading results.

In the case of co-occurrence-based methods such as PMI, there is a limitation of small corpora. Usually, these approaches rely on available software requirements document. Due to this, their performance is random. The small document is not sufficient for PMI to perform properly. For example, the word email and logon do not occur in the same FR statement in PROMISE dataset. Another study [19] described that the performance of PMI would be enhanced by additional training data. In contrast, the LSA achieve sensible results. However,

TABLE I. SIMILARITY MEASURE OF WORD "EMAIL" WITH DIFFERENT SIMILARITY METHODS

Words	Rensik sim *	Wu sim	Lesk sim*	LSA sim	PMI sim	NGDWiki sim	Word2Vecwiki sim
password	0.78	0.33	16	0.52	0.083	0.49	0.52
color	0.78	0.38	35	0.02	0.083	0.2	0.052
font	0.77	0.3	41	0.01	0.083	0.23	0.1
logon	0.0	0.0	0.0	0.14	0.083	0.355	0.37

* shows that the similarity values are in the range of $(0 - \infty)$ and rest methods in range (0-1)

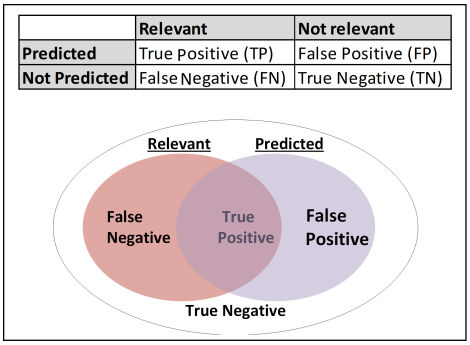

Fig. 1. Block diagram for precision-recall terms

TABLE II. DESCRIPTION OF PROMISE NFR DATASET

NFR Symbol	NFR	No. of NFR
A	Availability	21
FT	Fault tolerance	10
L	Legal	13
LF	Look and feel	38
MN	Maintainability	17
O	Operational	62
PE	Performance	54
PO	Portability	1
SC	Scalability	21
SE	Security	66
US	Usability	67
F	Functional	255

in their proposed approach the solution is brute force based and LSA has itself a computation intensive solution [18].

Word2Vec embedding has preference over LSA due to linear regularities among words [15]. The Wikipedia used in Word2Vec embedding has better results as shown in Table I. Although the NGD is using Wikipedia, their study shows that there is overhead of training Wikipedia data in converting Wikipedia XML dump into the local database to use it NGD and database size is 26.7 GB. This training is essential because NGD is designed for Google API hit count based. The API provides the service to developers to access and integrate the Google API in their program. Furthermore, there is a limit of quota enforced by Google to hit. Therefore, Word2Vec is used in this method, it has no issue of portability and compatibility with Wikipedia and it produces sensible results. A study [20] used TF-IDF and Word2Vect for classification of text. The study compares the results of both techniques with and without stop words, the results show that without stop word classification performance is better.

C. Evaluation Metrics

To measure the performance of the approach, the study used the precision-recall, and F-measure. The formulas for these measures are as: P = TP / (TP + FP). Here P is precision, a number of corrected or relevant predicted items over a total number of predicted items. The evaluation terms TP, FP, TN and FN are explained through Fig. 1.

In Fig. 1, the term TP means a number of correct predictions, FP means predicted but not relevant item. FN means an item which was relevant but not predicted by the model. The recall measure is defined as R = TP / (TP + FN). The recall is defined as the number of corrected predictions over the total relevant types of requirement sentences. The relationship between the entities can be understood with the help of truth-table and Venn-diagram in Fig. 1.

Furthermore, the relevant mean actual class and predicted mean which is identified by the model. F1 measure means harmonic mean of the precision and recall measures. The formula is F1=2 (P $\times$ R) / (P + R). In perspective of NFR extraction, precision and recall is equally important. Recall measure is necessary because requirement engineer tends to identify all NFR. Furthermore, precision of the approach cannot be neglected because large number of false positives results to produce frustration for the requirement engineers.

D. Dataset used for Evaluation

For validation, the NFR identification approach is applied to a PROMISE dataset [21]. The dataset utilized in this study is taken from the Open Science Tera-PROMISE repository. The dataset comprises of 15 requirement specifications of MS student projects of DePaul University. The dataset includes total of 625 requirements written in natural language. The distribution of requirement statements is 255 FR and 370 NFR. Each requirement is classified as FR or NFR. In this dataset the functional requirement is labelled with "F". In the dataset, the NFR has eleven sub-categories along with a count of potential NFR in requirements in the dataset is listed in Table II. The NFR types given in Table II is selected for evaluation of NFR identification approach because several studies [22]–[25] used the PROMISE dataset for their evaluation.

III. RELATED WORK

Software Requirement Specification (SRS) is a document which contains requirement statements usually written in natural languages. In SRS documents, the non-functional requirements are embedded in functional requirements. There are different manual, semi automated and automated approaches to identify NFR from requirement documents [9]. Our approach is close to the automated extraction of NFR from the text documents.

Cleland-Huang, et al. [42] classify NFR in the requirement documents. The study applies TF-IDF with additional variable

to specify the frequency of indicator keywords. The study used the collection of weighted indicators keywords to identify NFR. The study observed that the identification of NFR type, "look and feel", should be improved. The strong point of the approach is that the study provides a base for the automatic extraction of NFR through indicator keywords. The limitation is that the study has very low precision. The recall has a high value of 81%, this is seeming intentional to show performance. A possible reason is that at that time, the high recall was a challenge in the approaches, which the study solved. Due to large false positive, the user gets frustrated in checking the NFR and then discards it.

Zhang, et al. [49] proposed text mining-based technique to identify NFR from requirements documents. The Support Vector Machine (SVM) with linear kernel is used for extraction process and measure the performance of the technique by executing it to the PROMISE dataset. Their study compares the performance of n-gram, single word and Multi Words Expressions (MWE). The comparison is depicted in Table III.

Slankas and Williams [32] present a tool-based approach. The study extracts 14 distinct NFR types from different documents such as installation manuals, requirement specifications, and user manuals. Study selects indicator keywords using probability analysis. For identification of NFR, they use support vector machine, multinomial Naïve Bays and K-nearest neighbors algorithms and compares the results. Furthermore, the approach is tested on different datasets such as CCHIT, PROMISE, iTrust and openEMR.

Mahmoud and Williams [18] present an information theoretic approach for identification of NFR from SRS document. The indicator potential keywords are selected through clustering algorithms. For classification of NFR, the study calculates the semantic similarity between the words using Normalized Google Distance (NGD). For evaluation of approach, The dataset such as SafeDrink, SmartTrip and BlueWallet are used. The average precision of the approach is 53% and recall of 83%.

Another study [50] extracts NFR from the user review of WhatsApp and iBooks. The approach is keyword augmentation based. The keywords are selected using the Word2Vec model. For classification of NFR, the study used supervised learning approaches such as Naive Bayes, J48, and Bagging. In these approaches, the Naïve revealed the best results. Three words to vectorization traditional techniques are used in their study, such that BOW, CHI2, TF-IDF. For enhancing the performance of word to vectorization, they used AUR-BOW, which makes use of Word2Vec to exploit textual semantics to augment user reviews. The strong point of the approach is used keyword augmentation and semantic similarity. The limitation of the study is that the study only considers four NFR (reliability, usability, portability, and performance). Furthermore, they evaluate their approach to self-designed dataset.

Majority of the studies used supervised learning method for detection of NFR. It is observed that supervised learning methods are labor-intensive and have the disbursal of training the model. In case of unavailability of training data, the manual work is required to prepare training data. If the size of the dataset is large, then large training data is required to achieve acceptable results. On the change of the domain of the dataset, the same process is repeated to train the model.

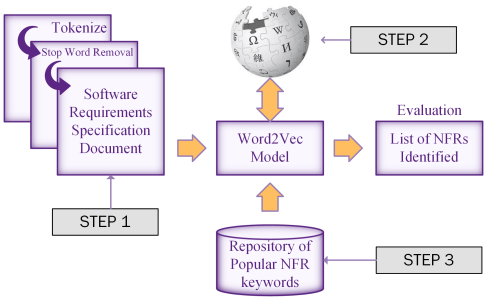

Fig. 2. NFR Identification Procedure

Furthermore, the pre-processing is effective in text mining and machine learning methods. In most of the existing studies, the performance is low in term of precision-recall. There is a need to adopt a semantic similarity-based method to identify the NFR. Internet-based data such as Google data repository and Wikipedia data help in similarity measure.

In Table IV, the existing studies are analyzed with respect to features. The study [42] utilized basic pre-processing and indicator keywords for identification, the Cleland study compares different algorithms. Another study [32] used the synonyms of words and hypernym of potential keywords. The study [18] by Mahmoud and Williams uses three different datasets for evaluation. However, the dataset was not shared due to a confidentiality agreement.

The proposed NFR Identification (NFRId) approach focused on three artifacts to improve the performance. Preprocessing is applied on all requirement sentences before identification process. The literature shows that pre-processing has a key role in improving the performance of machine learning based approaches [51]. Potential indicator keywords are selected by applying probabilistic analysis. The third reason for the improvement is the use of semantic similarity distance for identification. The NFRId has strong point against the existing studies in terms of using semantic similarity measure instead of string matching. Furthermore, NFRId approach has less dependency of trained data as compared to supervised learning based approaches.

IV. Automated Approach for NFR Identification

The study used the three steps procedure to explain the NFR identification (NFRId) approach. The approach utilizes repository of keywords contains the indicator NFR keywords, Word2Vector model is trained with Wikipedia and preprocessing methods for identification of NFR in the requirements document. The steps of identification approach are shown in Fig. 2.

1) Pre-Processing: Non-functional requirements are enclosed in Functional requirements and are usually express in some natural language. Usually, a manual process is adopted for identification of NFR from FR by the requirement engineers. There are some studies that proposed manual, semi automated and automated solutions for identification of NFR in a document [26]–[29]. The pre-processing techniques have a significant role in improving the performance of machine

TABLE III. OVERVIEW OF RECENT STUDIES AND EVALUATION MEASURES

Study	Method / Technique	Dataset	Feature	Advantages	Drawbacks
Cleland-Huang [42]	TF-IDF	PROMISE	The frequency of indicator	Average classification recall is 81%, TF-IDF with extra parameter frequency of indicator term	Large no. of False Positive, precision is 0.12%
Zhang, Yang [49]	SVM-linear kernel	PROMISE dataset	n-gram, individual word, (MWE)	Less effort is required by the analyst in term of preparing training dataset	Consider look and feel, security, legal, usability
Slankas and Williams [32]	KNN, SMO, MNBs, NFRLocator Tool	PROMISE, CCHIT, openEMR, iTrust	Sentence representation (SR), vertex distance logic	Tradition pre-processing, probability analysis in indicator keyword selections	Low value of accuracy 0.38
Mahmood and Williams [18]	NGD, K-mean Clustering	SmartTrip, SafeDrink, BlueWallet	Normalized Google Distance	Use an unsupervised learning approach, semantic similarity based	Evaluated by private dataset not standardized
Lu and Liang [50]	Naive Bayes, J48, and Bagging BOW, CHI^2, TF-IDF	User reviews from iBooks and WhatsApp	Word2Vec	Semantic similarity, keyword Augmentation	Selected NFR, (reliability, usability, portability, and performance), dataset self-prepared

TABLE IV. FEATURE WISE ANALYSIS OF THE EXISTING STUDIES

Study	Word2Vec / NGD	TF-IDF	Semantic similarity	Indicator keywords	Traditional pre-processing	POS tagging / Language Parser	Supervised learning	Public dataset
Cleland-Huang, Settimi [42]	X	✓	✓	✓	✓	X	✓	✓
Zhang, Yang [49]	X	✓	X	✓	✓	X	✓	✓
Slankas and Williams [32]	X	✓	X	✓	✓	✓	✓	✓
Mahmoud and Williams [18]	✓	X	✓	✓	✓	X	X	X
Lu and Liang [50]	✓	✓	✓	X	X	X	X	X
Proposed NFRId approach	✓	X	✓	✓	✓	✓	X	✓

TABLE V. TOKENIZATION, STOP WORDS REMOVAL AND LEGITIMIZATION EXAMPLE

Original statement	The system shall display the Events in a graph by time.
Stop word removal	system shall display events graph time.
Punctuation removal	system shall display events graph time
Lemmatization	system shall display event graph time

learning based approaches such as feature selection, extraction and classification [18], [30]–[32]. The identification approaches usually convert text into vector form, so, there is no meaning of arrangement of words or tokens. Tokenization is a process of breaking text into pieces (words or sentences) called as tokens.

The process of removing unnecessary words is called filtering of data. The unnecessary words are stop-words and punctuation ("the", "in", "by", "of"). The punctuation decreases the ambiguity of meaning. As the words are transformed into vector. So, punctuations has a least significant effect on words embedding' s order. Therefore, removal of punctuation increases the computation performance of the extraction process [33], [34]. The effect of each process on requirement statement with the example is listed in Table V.

2) Training of Word2Vector model: In the second step, Wikipedia dump of data is used to configure Word2Vec model [35]. The size of a dump of Wikipedia article is about 8 GB in compressed form. After converting into plain text, the dump file is of size (down to 13GB) by using the genism script. The plain text is pre-processed and converted into Word2Vec model format, also called word to vector embedding through Genism toolkit. Each topic in Wikipedia is converted into one sentence. This process takes a long time with good CPU speed (take 7+ hours on mac PRO (CPU is 4 core and RAM is 16G) [36]. Pre-trained embedding has an important role to achieve better generalization [37], [38]. Therefore, Word2Vec model is configured on Wikipedia dump of data [39]. For This study, a Genism Word2vec model built on the Engl5ish Wikipedia (Feb 2015), with 1000 dimensions, 10cbow, and no stemming is used [40].

3) Indicator keywords: The source of indicator terms is from different studies [32], [41], [42] and quality standards. These popular keywords are normally used to express certain non-functional requirement. The indicator keywords have an important role in the classification of NFR from a text document [41], [43]. The other sources for indicator keywords are quality standards ISO/IEC 25010 [44] standards document and IEEE Standard 610.12 [45].

In experimentation, the study used PROMISE dataset with a total of 625 sentences, 370 of them is annotated as "NFR", while 255 of them as "FR", for further use, will be referred to as CorpusNFR and CorpusFR, respectively. Like Chung, et al. [46] approach, the paper measures the probability of potential indicator keywords in the dataset. The frequencies of the words are used for calculating the probability of indicator keywords. Each group is ranked according to the feature probability measures [47]. The probability of the keywords is measured with the following formula.

TABLE VI. LIST OF INDICATOR KEYWORDS

NFR type	popular keywords
Availability	Availability achieve addition available schedule year period
Legal	Legal law regulations audit license standard custodian definition scope jurisdiction lawyer regulation insurance standard comply ramification liability
Look and feel	access color font graphic green magnify picture red simple blue look feel schema thump appealing
Maintainability	able change configurable integrate maintain new support update
Operational	Operational format mysql infrastructure interoperability machine platform extraction model operate interchange
Performance	date memory processor refresh response startup second speed hour trans transmit time signal live
Scalability	Scalability scalable multiple capable concurrent handle maximum simultaneous
Security	password authenticate authorize protect allow decrepit deny attack malicious protect login email log register role encrypt biometric sensitive restrict prevent
Usability	Usability wrong learn drop realtor voice collision easily successfully estimator intuitive easy enterer word community help symbol training conference let map
Fault tolerance	Fault avoidance tolerance failure unavailable remain restored offline operate remain
Portability	Portability portable

$$P(Word) = \frac{\#ofWordsinCorpusNFR}{(\#ofWordsinCorpusNFR) + (\#ofWordsinCorpusFR)} + \frac{\#ofWordsinCorpusNFR}{(\#ofWordsinCorpusFR + 1) \times \alpha} \quad (2)$$

where α is a constant for scaling factor. The smaller value of α, higher the scalability. In this study, the $\alpha = 10$ is used. The keywords selected by Equation (2) are listed in Table VI.

4) Execution of NFR Extraction Approach: The requirement statement has N number of words $W = W_1, W_2, ...W_N$ and each NFR type have M number of variant or morphological words $R = R_1, R_2, ...R_M$ to represent NFR type taken from literature as listed in Table VI. Word2Vec is a model trained on Wikipedia data to calculate the similarity between two words. The procedure for calculating the sentence similarity with a particular NFR type is expressed in the form of the mathematical formula given in Equation (3).

$$SentSim = \frac{1}{N} \sum_{i=1}^{N} \max_{0 \le j \le M} Word2Vec_{wikipedia}(W_i | R_i) \quad (3)$$

Equation (3) is implemented in the in the pseudocode given below:

1) Extract the first requirement sentence from the requirements document and get tokens.
2) Find the maximum similarity value between the first token of requirement sentence and all variants of the first NFR type given in Table VI. Repeat the same process with the second token and so on. Find the average value of all the token having the highest similarity value.
3) Repeat step 2 for all NFR types listed in Table VI and find the average similarity value.
4) The NFR type having the highest average similarity value calculated in step (2) and (3) is our required NFR type.
5) Repeat step (1 to 4) for all sentences in requirements documents one by one.

TABLE VII. SOME SAMPLE NFR TYPES AND A SIMILARITY VALUE

Requirement	Availability	Legal	Look and feel	Maintainability	Operational	Performance	Scalability	Security	Usability	Fault tolerance	Portability	Result
application color schema forth department homeland security	0.384065	0.248786	0.938915	0.53095	0.49276	0.322887	0.238473	0.346868	0.307832	0.305455	0.314048	LF
system form table system form table	0.235096	0.233892	0.298771	0.33138	0.335732	0.267577	0.238804	0.223263	0.275927	0.262082	0.286431	F

The pseudo code finds the similarity value between a requirement statement and NFR type. Take the first word of requirement statement and find similarity value of the NFR variant having a maximum value. Take the second word of requirement statement with the highest similarity value and so on. The Algorithm in pseudo code returns the average similarity value of all words in the requirement statement.

In Table VII, the similarity value against each NFR type is calculated by using Equation 3 and the procedure described in the pseudo code. The requirement given in Table VII is in preprocessed form. In the evaluation, only the NFR type with the highest value but greater than threshold similarity value (λ) is considered as the extracted NFR type, otherwise considered as FR. The Threshold value taken in Table VII is λ =0.5. In Table VII, the highlighted value meets the criteria and identified as Look and Feel (LF). Other NFR types having similarity distance values greater than threshold similarity value (λ) but not highest are discarded.

For results, the study executes the approach and get results by applying pre-processing method given in Section III-1. The results are calibrated with the threshold similarity value (λ) as shown in Table IX.

V. EVALUATION

Performance is measured in term of precision, recall, and F-measure. The precision-recall is a method explained in Section II earlier. We used the PROMISE NFR [48] data set for the evaluation. The classification breakdown of different NFRs in SRS document is given in Table VIII.

Precision and recall are equally important in extraction process. The precision increases the satisfaction of analyst. In the NFR extraction perspective, the proposed NFRId solution helps the analyst in the extraction of NFR and finally, an analyst re-confirms the NFR types. So, maximum recall also helps the analyst in a greater number of NFRs identifications.

TABLE VIII. PERFORMANCE OF EACH NFR TYPES AT THRESHOLD λ=0.5

Requirements	TP	FP	FN	Precision	Recall	F-measure
A	12	18	9	0.4	0.5714	0.4706
L	6	5	7	0.5455	0.4615	0.5
LF	12	6	26	0.6667	0.3158	0.4286
MN	4	23	13	0.1481	0.2353	0.1818
O	27	14	35	0.6585	0.4355	0.5243
PE	18	5	36	0.7826	0.3333	0.4675
SC	7	2	14	0.7778	0.3333	0.4667
SE	39	20	27	0.661	0.5909	0.624
US	39	10	28	0.7959	0.5821	0.6724
FT	4	53	6	0.0702	0.4	0.1194
PO	0	0	1	0.0000	0.0000	0.0000
F	172	129	83	0.5714	0.6745	0.6187
Average				0.5065	0.4111	0.4228

TABLE IX. PERFORMANCE OF NFR IDENTIFICATION APPROACH AT DIFFERENT VALUE OF (λ)

Threshold (λ)	Precision	Recall	F-measure
0	0.3372	0.4327	0.3484
0.1	0.3372	0.4327	0.3484
0.2	0.3372	0.4327	0.3484
0.3	0.3375	0.4327	0.3486
0.4	0.4531	0.4449	0.4151
0.5	0.5065	0.4111	0.4228
0.55	0.503	0.3786	0.3988
0.6	0.5039	0.3448	0.3694
0.61	0.5073	0.3442	0.3695
0.65	0.5064	0.321	0.3522
0.7	0.5379	0.2932	0.3356
0.8	0.552	0.2194	0.2563
0.9	0.6225	0.1413	0.1546
0.99	0.034	0.0833	0.0612

Highlighted values show the best value of the measure

Based on the literature, pre-processing has importance in the field of information retrieval methods. So, the study first applies traditional pre-processing on NFR extraction approach. The study applies tokenization, stop word removal and lemmatization before identification process. The highest value of precision, recall, and f-measure at different threshold values of λ (0-0.99) are given in Table VII. In Table VII, the highest value of recall is 45% at λ=0.4, the highest value of F-measure 42% at λ=0.5 and precision 62% at λ=0.9. It is noted that precision value (62%) is at the cost of recall only 14%. The average performance of extraction with respect to maximum F-measure value is as precision of 50%, recall of 41%, and f-measure of 42% at λ=0.5. The performance of identification approach varies on different values of λ. The effect of change is given in Table IX.

Table X shows that the overall performance of the proposed NFRId approach outperforms the Cleland study and Slankas study. All the studies given below in table are used tera-PROMISE NFR dataset. The Slankas study did not describe the performance of their approach in term of precision-recall while evaluating on PROMISE dataset. The automated approach reduced the manual human effort in the identification of NFR from the requirements document.

VI. CONCLUSION

In supervised learning, a significant number of pre-categorized requirements are needed to train text classifiers. These pre-categorized requirements are generated by manual classification then the trained model produces better results in terms of identification of NFR. The other limitation of the supervised approach is that if the model is trained for one domain and work well, however, it may not work on some other domain. Experts from different domain use different terminologies. So, you must re-train or re-tune the model. Furthermore, supervised l earning approaches are effective for a small system but face challenges on a large scale or when the systems are not well structured. In this paper, an NFRId approach is proposed and designed for identification of NFR from the requirements document. Our approach does not need pre-categorized requirements for training of Word2Vec model. Therefore, the human's manual effort is reduced in NFR identification process. In the approach, we find the similarity distance between popular NFR keywords and requirements statements. The similarity distance is measured by using Word2Vec model which is pre-trained on Wikipedia dump of text data. The approach is applied on NFR dataset taken from PROMISE dataset repository and performance is measured in term of precision-recall and F-measure. Our result in term of F-measure is 0.47.

TABLE X. COMPARISON OF THE PROPOSED APPROACH WITH EXISTING STUDIES

Study	Precision	Recall	F-measure
Cleland-Huang, et al. [42]	0.147	0.626	0.239
Slankas and Williams [32]	-	-	0.38
Proposed NFRId Approach	0.5065	0.4111	0.4228

VII. LIMITATIONS

1) Indicator keywords are focused for PROMISE dataset. so, there is chance to affect the performance for another dataset, for example, medical domain has a different set of word to represent the NFR type.
2) Number of NFR types are bound by the Creator of dataset who labeled it.
3) The dataset has some misconception in labeling of NFR. For example, R2.18 "The product shall allow the user to view previously downloaded search results, CMA reports and appointments" is labeled as NFR in the tera-PROMISE dataset by its creators, however, this is a functional requirement. Our approach also detects it as a functional requirement. The selection of different NFR in the tera-PROMISE dataset seems to some bias.
4) Word2Vec model is bound to Wikipedia vocabulary bank. If the word present in the requirements are not present in Wikipedia, the model cannot find similarity value. It is the limitation of Word2Vec model.

VIII. FUTURE WORK

NFRId approach uses unsupervised learning-based model, however, it uses indicator keywords which is a manual process so, as a whole, the approach is semi supervised learning based. In future work, there should be an approach that will use some way to extract indicator terms from clustering or some unsupervised way then our approach becomes a fully unsupervised approach.

ACKNOWLEDGMENT

The authors would like to thank to Universiti Teknologi Malaysia (UTM) for providing resources for this research.

REFERENCES

[1] W. M. Farid, "The NORMAP Methodology: Lightweight Engineering of Non-functional Requirements for Agile Processes," in 2012 19th Asia-Pacific Software Engineering Conference, 2012, pp. 322-325.

[2] B. Yin and Z. Jin, "Extending the problem frames approach for capturing non-functional requirements," in Computer and Information Science (ICIS), 2012 IEEE/ACIS 11th International Conference on, 2012, pp. 432-437.

[3] C. Hoskinson, "Army's Faulty Computer System Hurts Operations," Politico, 2011.

[4] J. Bertman, N. Skolnik, and J. Anderson, "EHrs get a failing grade on usability," Internal Medicine News, vol. 43, p. 50, 2010.

[5] V. Bajpai and R. P. Gorthi, "On non-functional requirements: A survey," in Electrical, Electronics and Computer Science (SCEECS), 2012 IEEE Students' Conference on, 2012, pp. 1-4.

[6] M. Sadighi, "Accounting System on Cloud: A Case Study," in Information Technology: New Generations (ITNG), 2014 11th International Conference on, 2014, pp. 629-632.

[7] L. Xu, G. Tan, X. Zhang, J. Zhou, and Ieee, "Aclome: Agile Cloud Environment Management Platform," in 2013 Fourth International Conference on Digital Manufacturing and Automation, ed, 2013, pp. 101-105.

[8] v. one. (2017). 11th annual stat of art agile sruvey Available: http://stateofagile.versionone.com/

[9] N. Kannan. (2012, 2-1-2016). 6 Ways the Cloud Enhances Agile Software Development. Available: ttp://www.cio.com/article/2393022/enterprise-architecture/6-ways-the-cloud-enhances-agile-software-development.html

[10] X. Li, T. Guozhen, Z. Xia, and Z. Jingang, "Aclome: Agile Cloud Environment Management Platform," in Digital Manufacturing and Automation (ICDMA), 2013 Fourth International Conference on, 2013, pp. 101-105.

[11] versionone. (2015, 12/01/2015). 7th ANNUAL STATE of AGILE VERSIONONE® Agile Made Easier DEVELOPMENT SURVEY. Available: http://www.versionone.com/pdf/7th-Annual-State-of-Agile-Development-Survey.pdf

[12] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," Journal of the American society for information science, vol. 41, p. 391, 1990.

[13] S. T. Dumais, "Latent semantic analysis," Annual review of information science and technology, vol. 38, pp. 188-230, 2004.

[14] V. Abedi, M. Yeasin, and R. Zand, "Empirical study using network of semantically related associations in bridging the knowledge gap," Journal of translational medicine, vol. 12, p. 324, 2014.

[15] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.

[16] Y. Goldberg and O. Levy, "word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method," arXiv preprint arXiv:1402.3722, 2014.

[17] word2vec. (2013). WordeVec. Available: https://code.google.com/archive/p/word2vec/

[18] A. Mahmoud and G. Williams, "Detecting, classifying, and tracing non-functional software requirements," Requirements Engineering, pp. 1-25, 2016.

[19] R. Budiu, C. Royer, and P. Pirolli, "Modeling information scent: A comparison of LSA, PMI and GLSA similarity measures on common tests and corpora," in Large scale semantic access to content (text, image, video, and sound), 2007, pp. 314-332.

[20] J. Lilleberg, Y. Zhu, and Y. Zhang, "Support vector machines and word2vec for text classification with semantic features," in Cognitive Informatics & Cognitive Computing (ICCI* CC), 2015 IEEE 14th International Conference on, 2015, pp. 136-140.

[21] PROMISE, "PROMISE Software Engineering Repository data set," 2010, [online]. avaiable: https://terapromise.csc.ncsu.edu/!/#repo/view/head/requirements/nfr.

[22] J. Slankas and L. Williams, "Automated extraction of non-functional requirements in available documentation," in Natural Language Analysis in Software Engineering (NaturaLiSE), 2013 1st International Workshop on, 2013, pp. 9-16.

[23] J. Cleland-Huang, R. Settimi, X. Zou, and P. Solc, "Automated classification of non-functional requirements," Requirements Engineering, vol. 12, pp. 103-120, 2007. [

[24] D. Sunner and H. Bajaj, "Classification of Functional and Non-functional Requirements in Agile by Cluster Neuro-Genetic Approach," International Journal of Software Engineering and Its Applications, vol. 10, pp. 129-138, 2016.

[25] A. Rashwan, O. Ormandjieva, and R. Witte, "Ontology-based classification of non-functional requirements in software specifications: a new corpus and svm-based classifier," in Computer Software and Applications Conference (COMPSAC), 2013 IEEE 37th Annual, 2013, pp. 381-386.

[26] N. A. Ernst and J. Mylopoulos, "On the perception of software quality requirements during the project lifecycle," in International Working Conference on Requirements Engineering: Foundation for Software Quality, 2010, pp. 143-157.

[27] A. Casamayor, D. Godoy, and M. Campo, "Identification of non-functional requirements in textual specifications: A semi-supervised learning approach," Information and Software Technology, vol. 52, pp. 436-445, 2010.

[28] A. Mahmoud, "An information theoretic approach for extracting and tracing non-functional requirements," in 2015 IEEE 23rd International Requirements Engineering Conference (RE), 2015, pp. 36-45.

[29] W. M. Farid, "The Normap methodology: Lightweight engineering of non-functional requirements for agile processes," in Software Engineering Conference (APSEC), 2012 19th Asia-Pacific, 2012, pp. 322-325.

[30] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," Information Processing & Management, vol. 50, pp. 104-112, 2014.

[31] G. Feng, J. Guo, B.-Y. Jing, and L. Hao, "A Bayesian feature selection paradigm for text classification," Information Processing & Management, vol. 48, pp. 283-302, 2012.

[32] J. Slankas and L. Williams, "Automated extraction of non-functional requirements in available documentation," in Natural Language Analysis in Software Engineering (NaturaLiSE), 2013 1st International Workshop on, 2013, pp. 9-16.

[33] H. Saif, M. Fernández, Y. He, and H. Alani, "On stopwords, filtering and data sparsity for sentiment analysis of twitter," 2014.

[34] C. Silva and B. Ribeiro, "The importance of stop word removal on recall values in text categorization," in Neural Networks, 2003. Proceedings of the International Joint Conference on, 2003, pp. 1661-1666.

[35] wikipedia. (2018). Wikipedia:Database download. Available: https://en.wikipedia.org/wiki/Wikipedia:Database_download

[36] TextMiner. (2015). Training Word2Vec Model on English Wikipedia by Gensim.

[37] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," Journal of Machine Learning Research, vol. 12, pp. 2493-2537, 2011.

[38] D. Chen and C. Manning, "A fast and accurate dependency parser using neural networks," in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 740-750.

[39] Gensim. (2017). training word2vec on full Wikipedia. Available: https://groups.google.com/forum/#!topic/gensim/MJWrDw_IvXw

[40] wiki2vec. (2018). Generating Vectors for DBpedia Entities via Word2Vec and Wikipedia Dumps. Available: https://github.com/idio/wiki2vec

[41] J. Cleland-Huang, R. Settimi, X. Zou, and P. Solc, "The detection and classification of non-functional requirements with application to early aspects," in Requirements Engineering, 14th IEEE International Conference, 2006, pp. 39-48.

[42] J. Cleland-Huang, R. Settimi, X. Zou, and P. Solc, "Automated classification of non-functional requirements," Requirements Engineering, vol. 12, pp. 103-120, 2007.

[43] L. Rosenhainer, "Identifying crosscutting concerns in requirements specifications," in Proceedings of OOPSLA Early Aspects, 2004.

[44] ISO/IEC. (2008, Oct 2016). Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE). Available: https://www.iso.org/obp/ui/fr/#iso:std:iso-iec:25010:ed-1:en

[45] I. Std. (1990). IEEE standard glossary of software engineering terminology. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=159342

[46] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, Non-functional requirements in software engineering vol. 5: Springer Science & Business Media, 2012.

[47] I. Hussain, L. Kosseim, and O. Ormandjieva, "Using linguistic knowledge to classify non-functional requirements in SRS documents," in International Conference on Application of Natural Language to Information Systems, 2008, pp. 287-298.

[48] C. Beutenm, #252, ller, S. Bordag, and R. Assadollahi, "A private living lab for requirements based evaluation," presented at the Proceedings of the 2013 workshop on Living labs for information retrieval evaluation, San Francisco, California, USA, 2013.

[49] W. Zhang, Y. Yang, Q. Wang, and F. Shu, "An empirical study on classification of non-functional requirements," in The Twenty-Third International Conference on Software Engineering and Knowledge Engineering (SEKE 2011), 2011, pp. 190-195.

[50] M. Lu and P. Liang, "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, 2017, pp. 344-353.

[51] Hussain, I., Kosseim, L., and Ormandjieva, O. "sing Linguistic Knowledge to Classify Non-Functional Requirements in Srs Documents", in Proceeding of the International Conference on Application of Natural Language to Information Systems, 2008, pp. 287-298.

[52] Beel J, Gipp B, Langer S, Breitinger C. paper recommender systems: a literature survey. International Journal on Digital Libraries. 2016 Nov 1;17(4):305-38.