I.J. Information Engineering and Electronic Business, **2023, 2, 38-53**
Published Online on April 8, 2023 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijieeb.2023.02.05

Assessing Similarity between Software Requirements: A Semantic Approach

Farooq Ahmad
Department of Computer Application, Integral University, Lucknow, India
Email: farooqa@iul.ac.in
ORCID iD: https://orcid.org/0000-0002-3944-7710

Mohammad Faisal
Department of Computer Application, Integral University, Lucknow, India
Email: mdfaisal@iul.ac.in
ORCID iD: https://orcid.org/0000-0002-6120-5259

Received: 17 August, 2022; Revised: 18 September, 2022; Accepted: 15 November, 2022; Published: 08 April, 2023

Abstract: The majority of projects fail to achieve their intended objectives, according to research. This could arise for a number of reasons, such as ensuring requirements are managed, excessive documentation of the code, or the difficulty in delivering software that includes all the requested features on time. An effort could be made to overcome such failure rates by establishing a proper management of requirements and concept of reusability. The correct requirements can be identified by checking similarity between the requirements received from the various stakeholders. A reusable software component can result in substantial savings in both time and money. It can be challenging to make a choice regarding the reuse of certain software components. A comparison of the requirements of a new project with those of previous projects prior to starting a new project or even at a later stage during development is useful for identifying reusable components. This paper proposes a framework (ReSim) for identifying software requirements' similarities, in an attempt to improve reusability and identify the correct requirements. A crucial component of ReSim is to measure similarity between software requirements. Different well-known similarity measurement techniques used by the researchers to evaluate the similarity between the software requirements. Some of the methods used to measure this include dice, jaccard, and cosine coefficients, but in this paper, we have used recently developed hybrid method which considers not only semantic information including lexical databases, word embeddings, and corpus statistics, but also implied word order information and produced significant improvements in the results related to the measurement of semantic similarity between words and sentences. As part of the experiments, the study used PURE dataset - in order to demonstrate the efficacy of the proposed framework. As a result, recently developed hybrid method of measuring the requirements similarity is more accurate than Dice, Jaccard, and Cosine, while Cosine is a better choice than Dice, and Jaccard is more accurate than Dice. Thus, ReSim outperforms existing approaches when tested on the PURE dataset, providing the most accurate results for both functional and non-functional requirements.

Index Terms: Measurement; Requirements; Similarity; Semantic; Reusability; Framework

1. Introduction

Natural language processing can be performed in a variety of ways that rely on the notion of semantic similarity, including terms sense disambiguation, automatic synonym extraction, language modeling, and document clustering. Semantic similarity measurements use a variety of methods to measure similarity. A federated database, data integration system, message passing system, web service, or peer-to-peer data management system [1] may utilize text similarity as a method to match schemas to minimize semantic heterogeneity, a major problem in any data sharing system. A growing number of applications provide data retrieval through similarity measurement operations, including multimedia [2, 3], data mining [4], CAD database system [5], time series databases [6], information retrieval system (IR) [7], geographic information systems (GIS) and tourism information system.

Requirements Engineering (RE) on the other hand, is gaining importance and is acknowledging the importance of a successful software project in the early stages [7]. A system is concerned with identifying the goals to be achieved by the stakeholders [8], achieving these goals through systematic implementation, and assigning responsibilities to the appropriate agents of a system such as humans, devices, and software [9].

Recent applications of natural language processing present a need for an effective method to compute the similarity between very short texts or sentences. An example of this is a requirements management system expected for identifying software requirements' similarities, in an attempt to improve reusability and identify the correct requirements. Researches show that a reasonable number of software projects failed to achieve their desired goals. This could arise for a number of reasons, such as ensuring requirements are managed, excessive documentation of the code, or the difficulty in delivering software that includes all the requested features on time. An effort could be made to overcome such failure rates by establishing a proper management of requirements and concept of reusability. The correct requirements can be identified by checking similarity between the requirements received from the various stakeholders. A reusable software component can result in substantial savings in both time and money. It can be challenging to make a choice regarding the reuse of certain software components. A comparison of the requirements of a new project with those of previous projects prior to starting a new project or even at a later stage during development is useful for identifying reusable components. A framework is expected for identifying software requirements' similarities, in an attempt to improve reusability and identify the correct requirements.

The problem of measuring sentence similarity is an essential issue in the natural language processing area. Computing semantic similarity between software requirements in natural language is a related issue. In requirement engineering, the task of measuring requirements similarity is to find semantic symmetry in two sentences of software requirements, regardless of word order and context of the words. It is necessary to measure the similarity between the sentences of software requirements accurately. A framework (ReSim) is presented in this paper for identifying software requirements' similarities, in an attempt to improve reusability and identify the correct requirements.

The main purpose of the proposed framework (ReSim) is to identify the similarities between the requirements of a current software project and those available in the software repository [10]. It would be more beneficial, in a more efficient way, to use a similarity measurement technique to form more effective measurements [11], which would ultimately allow reusable software components to be used based on these similar requirements. The recently developed hybrid method [12] has been used in ReSim to measure similarity of software requirements. Further, it represents a path to achieving any software project's ultimate goal, such as saving time, effort, costs etc.

Reusability refers to the ability to use existing components to solve different problems [13]. Through reuse of existing libraries, components, modules, and sources, software can be made reusable. This can also apply to software designs [14]. By reusing existing libraries, components, modules, and sources, software can be made more valuable and productive. A reusability strategy can save time, effort, and money [15-17].

After identifying similar components, reusing the design and code of a completed project can be accomplished smoothly. Measuring the similarity between requirements can improve and facilitate reusability [18]. In addition to construction for reuse, construction with reuse refers to the use of reusable components in the construction of a new system. The former refers to the development of components for further reuse while the latter refers to the construction of components for reuse [15,19]. We propose an approach that encourages construction with reuse in this paper.

A crucial component of ReSim is to measure similarity between software requirements. Different well-known similarity measurement techniques used by the researchers to evaluate the similarity between the software requirements. Some of the methods used to measure similarity between the software requirements include dice, jaccard, and cosine coefficients, but in the proposed framework (ReSim) we have used recently developed hybrid method [12] which considers not only semantic information including lexical databases, word embeddings, and corpus statistics, but also implied word order information and produced significant improvements in the results related to the measrurement of semantic similarity between words and sentences. The main objective of the proposed framework (ReSim) is to measure the similarity of requirements documents for the purpose of facilitating the reuse of some components.

Organizing this work is as follows. Section two contains a brief introduction and background information. Section three is devoted to a discussion of related works. The problem of identifying similarities between requirements is described in section four. We provide a step-by-step explanation of the proposed similarity framework in section five. The experimental work and results are discussed in section six. Finally, the conclusions are drawn after summarizing the findings.

2. Background

In this section, we cover preliminary concepts needed in our work. These preliminary concepts are the requirements engineering, similarity of requirements and similarity measurement.

2.1 Requirements Engineering

Since it has become increasingly important, requirements engineering has gained more attention in recent years. Requirements engineering involves several steps, including requirements elicitation, analysis, specification, validation, and management. It is the important objective of requirements engineering to develop a set of requirements that meets some criteria such as completeness, consistency, and correctness. In general, software requirements fall into functional and non-functional categories. The requirement engineering process should not require further iteration if it produces requirements that meet the customer's needs [20-22].

In requirements engineering, a document of system requirements is created and maintained. The process involves three stages of activities that are arranged iteratively.

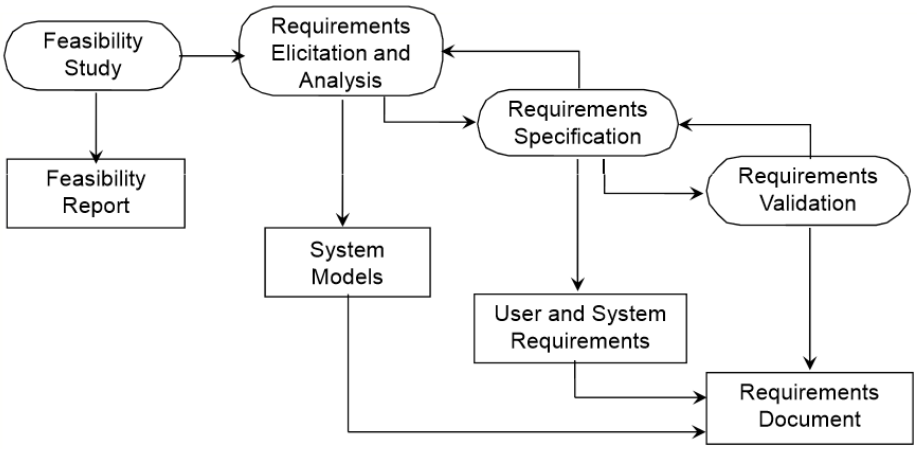

Fig. 1. requirements engineering process [20-22]

2.2 Requirements Similarity

A requirement, by definition, is an exact description of what the system will do without stating how it will accomplish it.

In order for requirements engineer to select and configure appropriate components, specific information must be targeted, which could be explicit or implicit. Implicit information are perceptions, rules, standards, and domain knowledge, which requirements engineers employ on behalf of implied information. Specific information comprises written text, flow charts, drawings, and other artifacts.

It has been under consideration for some time to work on similarity analysis and modeling, but this paper presents a different approach. Using information retrieval techniques for determining statistical measurements of requirements, Natt och Dag et al. analyzed textual requirements for automatic similarity analysis to prevent duplicate requirements from being identified [23].

2.3 Similarity Measurement

In order to measure semantic similarity, individuals, texts, features, concepts, or ontologies can be considered. In this paper, we examine the idea of inter-concept (or inter-text) similarity. Various characterizations of similarity are based on how representations are constructed, such as how concepts are represented as unstructured features, how dimensions can be specified in multidimensional spaces [24, 25], or how descriptive logic can specify different restrictions [26, 27]. Computer concepts are the same as concepts in the human mind; the similarity is determined by what is said about these concepts [28].

2.3.1 Lexical-Based Similarity

The concept of lexical similarity only applies to syntax. In other words, it consists of reading or reading sequences of characters, or how similar the words are despite their differences in meaning. Character-based similarity and term-based similarity are the two primary groups of lexical similarity measures. Many algorithms have been used to determine lexical similarity, including Longest Common Substring (LCS), Damerau-Levenshtein, Jaro, Jaro-Winkler, and others [29,30].

The algorithm identifies the degree of similarity between documents by displaying all documents as a matrix. Each row represents tokens or words extracted from each document, and every column represents a document. Each row in the matrix corresponds to a score, and every column corresponds to the amount of similarity between documents [31,32]. In the case of larger numbers of documents, such as the requirements documents for several years of a multinational company, the computations will be extremely tedious. In light of this, it may be appropriate to aggregate certain documents, and to represent their relationships logically. In a relation between two documents, R(d1,d2,v) is a measure of the number of common tokens between the documents, where d1 represents the first document, d2 represents the second document [32]. The following equation 1 can be used to calculate the similarity between d1 and d2 documents:

$$\mathrm{Sim}(\mathrm{d1},\mathrm{d2}) = \frac{|\mathrm{d1} \cap \mathrm{d2}|}{|\mathrm{d1} \cup \mathrm{d2}|} \quad (1)$$

2.3.2 Semantic-Based Similarity

A phenomenon of semantic similarity was discovered in the 1990s in psychological experiments [33]. In psychology, semantic similarity is a basic concept that helps understand and organize objects. By comparing the meaning behind the words, semantic similarity determines how similar the words are. By way of example, lexically, 'gift' and 'present' are completely unlike each other, but semantically they are also similar. Comparing large corpora of words through corpus-based similarity techniques. An extensive set of text used for research purposes is referred to as a corpus. The use of semantic similarity in query answer systems helps users to find what they are looking for regardless of how the characters are written [29-31]. Similarity is also measured using ontologies. By connecting conceptualization with semantic pointers, ontologies offer organized structures and clear representations of knowledge [32, 34].

In this study, uses a hybrid methodology presented by Farooq [12] for calculating semantic similarity between the requirements received from different stakeholders or between the requirements for current projects and those for previous projects.

We can measure the similarity between requirements written by the same person (or group) by using Farooq's [12] hybrid methodology. As requirements written in natural language vary from person to person, so do the style of writing and words selected, so the similarity measurements could be impacted as well. Ontologies and synonyms could be another important direction for research. The same meaning can be applied to words in natural languages regardless of where they are placed. The similarity matching between two requirements should be high if they are written with different synonyms, but they have the same meaning and scope.

2.3.3 Hybrid Methodology for Computing Semantic Similarity

Farooq [12] has proposed a hybrid technique (HydMethod) that considers not only the semantics of the texts but also the implicit word order. Incorporating corpus statistics will enable us to adapt lexical databases to be used in different domains by modeling human common sense knowledge. Accordingly, the methodology can be applied to various domains. In order to demonstrate how effective, the methodology is, Farooq [12] used two standard datasets - Pilot Short Text Semantic Similarity Benchmark [29] and MS paraphrase [35, 36]. When tested on both of these datasets, the proposed method is found to perform better than the existing approaches; this correlation is highest both for word similarity and sentence similarity. This method has an overall improvement of 32% over the use of word vectors and WorldNet methods alone. Farooq [12] found a high Pearson correlation coefficient of 0.8953 for Rubenstein and Goodenough [37] word & sentence pairs.

An analysis of the methods used to measure sentence similarity is presented in a survey paper by Farooq [38]. This study focused on narrowing down the issues and identifying more research gaps associated with the calculation of similarity.

The hybrid methodology presented by Farooq [12] is based on WordNet and vector similarity which is integrated at the word level. As shown in the equation 2 below, the measure of similarity involves a weighted average of both similarities.

$$sim(w_1,w_2) = \text{Vector sim}(w_1,w_2) * \mu + \text{WordNet sim}(w_1,w_2) * (1-\mu) \quad (2)$$

where μ is a constant used to determine the weighted average of the similarities. The value μ is set to 0.33 in our experiment.

In order to determine the semantic similarity between the sentences s1 and s2, we can use the following equation. We also calculate the similarity between s1 and s2 and between s2 and s1, in order to generate an accurate similarity analysis. To compute the final sentence similarity, the following equation 3 calculates the average of these two similarities.

$$sim(S1,S2) = \text{Average } ((\sum_{i=0}^{n} Max(Word\ sim(wi,S2)))/n + (\sum_{j=0}^{m} Max(Word\ sim(wj,S1)))/m) \quad (3)$$

With this approach, S1 consists of n words, and sim(wi,S2) represents the similarity between a word i in S1 and the statement S2. Similarly, word j in S2 is similar to the word wi in S1, and sim(wi,S1) shows their similarity.

After constructing the vectors, the word order similarity is calculated using the following equation 4.

$$Sw = 1 - \frac{||V1-V2||}{||V1+V2||} \quad (4)$$

By combining word order similarity and semantic similarity, final similarity can be achieved using the following equation 5.

$$Sim(S_1, S_2) = \sqrt{} (\gamma^2 S_s + (1 - \gamma^2) S_w) \quad (5)$$

Similarity in word order is represented by Sw, while semantic similarity is represented by Ss. The value of γ varies from 0 to 1.

According to the results of experiments using standard datasets, hybrid methods produce good results. Incorporating word order similarity improves the measure of sentence similarity further.

3. Related Work

Several approaches in software engineering use similarity measurements to analyze the relationships between different software artifacts. Typical examples include identifying features [39], finding features [13], recovering architecture [40], identifying reused services [14], and detecting clones [41].

Natural language is defined by the Cambridge Dictionary as "language that has developed in the usual way as a method of communicating between people, rather than language that has been created, for example for computers." It is the language learn to interpret the spoken and written word, and to speak and write ourselves. For a human, this usually comes natural as they learn to speak and write in the youth. However, for a computer, that is a different issue.

When computing similarity in the RE field, Natural Language Processing (NLP) techniques are typically used to represent the requirements [42], as they are written in Natural Language (NL) [43-45]. Similarity computation is crucial for many typical requirements management tasks, including tracking [46-49], identifying equivalent requirements [50], analyzing change impacts [51, 52], extracting glossary terms and grouping them [9], and retrieving artifacts [35]. In the following, we present a comparison of our work with representative studies from the field of RE, focusing specifically on the topics of traceability, change impact analysis, and recommender systems, all of which are closely related to our work, as they involve both requirements and software similarities.

It is crucial to compare similar words, sentences, paragraphs, and document for research in many fields including information extraction, document clustering, and disambiguation of word senses, automatic essay scoring, small answer scoring, machine translation, and text summarization. Several techniques exist to measure document similarity [16,30,53], including document clustering.

The SimReq framework has been proposed by Illyas [54] to measure the similarity between software requirements of a running project and previously completed projects. The software compares each requirement component of the running project with the textual requirements of the software.

Illyas [55] has presented a comparative study of different similarity measurement techniques used in the SimReq similarity measurement framework. Through SimReq frame work, Illyas [55] emphasized the use of reusable components of software like design, coding, and test cases since SimReq frame work focused on finding similarities between new and old requirements of software projects.

Gomaa [56] presented a survey of existing works on text similarity and categorized them into three approaches; string-based similarities, corpus-based similarities, and knowledge-based similarities. Additionally, examples of these similarities are presented in combination.

A new research challenge is proposed to the community of requirements engineering by Tarawneh [57], who has discussed the relationship between requirements engineering and natural language processing.

Fatma [58] has proposed a framework for assessing requirements similarity using four-phases frameworks (FPS). The objective of FPS is to compare the textual requirements of different projects and measure their similarity. By measuring similarity Fatma [58] aim to facilitate reusability of certain components such as designs, codes, and test cases.

In an extension of a previous work, Fatma [59] presented an automated system that measures the similarity between software requirements to increase reusability of software. In this work Fatma [59] employed lexical text similarity to measure the similarity between a new project and a set of previously completed projects, which suggests some components that may be reused.

According to Samosir [10], a mapping can be generated between a set of requirements statements and the classes of a project that meet those requirements. Based on the class-requirement mapping and the associations between classes, the method also generates associations among requirements. A class diagram of the respected project is used to store relational information.

Azzam [53] explains how to resolve discrepancy between developers and clients about elicitation results for software requirements by combining elicitation along with a requirement statement. Azzam [53] accomplished this goal by following specific procedure steps, namely: determining the similarity between the software requirements specifications and the elicitation results, and analyzing the elicitation results by using text mining.

Abbas [60] has compared different state-of-the-art NLP approaches and found correlations between requirements and source code. A real-world evaluation is conducted using two industrial projects in the railway sector as inputs.

Abbas [61] has conducted an empirical investigation of the relationship between requirements and code similarity in the context of a large railway company. The purpose of this work was to explore to what extent similar requirements can be used to locate similar code. Abbas [61] analyzed two related projects in the company and used different seminal NLP-based language models to represent the requirements and calculate similarity across the two projects.

The main objective of this paper is to propose the framework “ReSim”. ReSim measures similarity between requirements by simply comparing textual requirements received from stakeholders and the requirements of the running project (or software) with those from previously completed projects. In this study, a recently developed hybrid methodology [12] to assess similarity between requirements.

Using PURE (public requirements dataset) - a collection of 79 publicly available documents [55, 62] describing natural language requirements obtained from the Internet, Table 1 presents a comparison of the semantic-based hybrid approach with other approaches used in ReSim to measure similarity of requirements. According to the results stated in his paper, Dice similarity measurement achieves 0.837 Pearson correlation. Moreover, Jaccard similarity measurement obtains a correlation coefficient of 0.842 and Cosine similarity measurement achieves 0.862. The hybrid methodology presented by Farooq [12] achieves 0.887. In Figure 7, hybrid approach in ReSim outperforms existing approaches when tested on the PURE dataset.

Table 1. Comparison of the discussed approaches to the semantic-based hybrid approach with Pearson's correlation coefficient

Similarity Measurement Method	Pearson Correlation
Semantic-based hybrid method [12]	0.887
Cosine coefficient [54, 63]	0.862
Jaccard coefficient [54, 64, 65]	0.842
Dice coefficient [54, 55]	0.837

4. Problem Description

According to studies, there are a reasonable number of projects that do not reach their goals. There could be a variety of reasons for this situation, including an inability to manage requirements and excessive documentation of code, along with the difficulty of delivering the software on time with all necessary features. Such a failure rate could be improved by managing requirements properly and utilizing the concept of reusability.

For proper management of requirements and adoption of the concept of reusability, requirement similarity measures are necessary at the following places in the requirement engineering process:

- Examining the similarity between requirements received from different stakeholders.
- Comparing the requirements of current projects with those of previously completed software projects.

The following reasons should be taken into consideration when attempting to measure the similarity of requirements received from different stakeholders:

- Stakeholders lack an understanding of their needs
- Stakeholders express their preferences differently

This could result in conflicting requirements from different stakeholders.

5. ReSim: A Framework for Similarity Measurement

When creating a new computer-based system, there are two ways to get started: either build all the components from scratch, or reuse some of the existing ones. A reusable component is an existing component that can be reused in other situations. Using reusable components improves productivity, accuracy, and quality. Additionally, reusability may reduce costs, effort, and time [21,12,17]. Reusing an existing system or starting over from scratch is not an option that can be taken lightly. In some cases, a decision-maker may require assistance. Also, it can be challenging to identify reusable components. Similarity measurements can be used to make reusing decisions. As shown in figure 2, reusability can be achieved if a project is similar to previous ones, as opposed to starting completely from scratch.

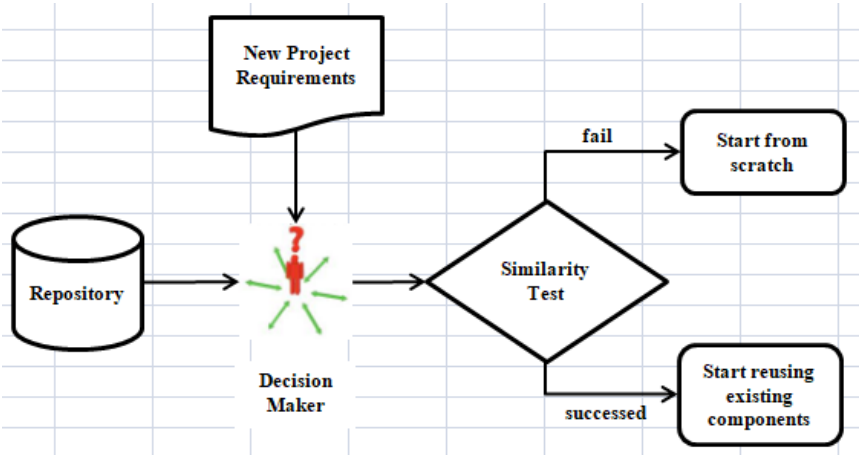

Fig. 2. Measuring the similarities between the requirements of current projects and those of previous projects [34]

Customer and end-user requirements are elicited through the use of interaction with the system as well as services that the solution should provide, as well as other constraints. The figure 3 shows the main processes involved in requirements elicitation and analysis.

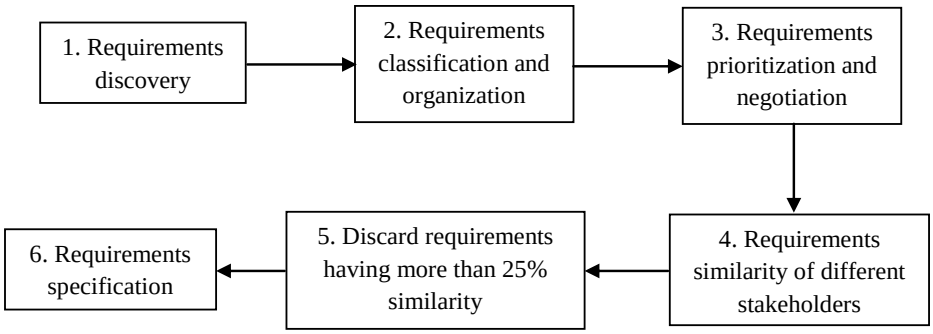

Fig. 3. requirements elicitation and analysis [19-21]

As part of the requirement elicitation process, stakeholders of a new project are asked to present their requirements without any ambiguity. Identification of the precise requirements and the expectations of various stakeholders is essential to determine what is expected of the project. In the process of resolving ambiguity between the requirements of different stakeholders, it is essential that you clarify the requirements first. The figure 4 illustrates the reasons why it is important to measure similarity in requirements across stakeholders.

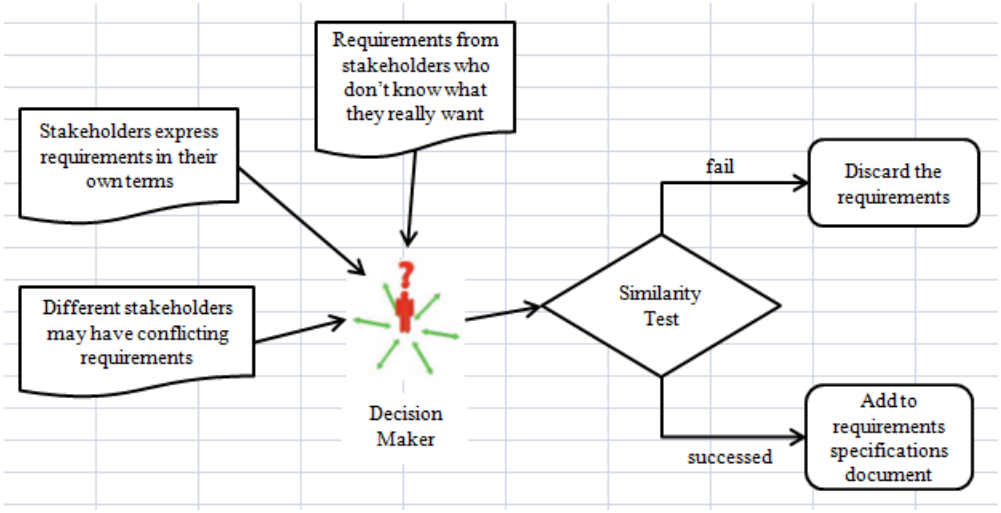

Fig. 4. Measuring the similarities between the requirements received from different stakeholders [59]

Using the framework, software requirements can be compared with requirements received from different stakeholders and with requirements of currently running projects (or software). Firstly, it examines the textual requirements of a stakeholder and compares them against the requirements of each other stakeholder. On success, requirements will be added to the requirements specifications document; on failure, requirements will be rejected. In a second step, a comparison is made between the requirements of the software stored in the repository and those of the active project. Once the matching requirements have been met, the matching code and design will be considered for reuse. If no match is found, a new design is generated and code for this requirement is also generated. Post-processing activities in ReSim include configuration, testing, and implementation.

5.1 Data preparation

Before we can measure the similarity of requirements, we must first translate their sentences into words, in which case we will perform a hybrid method [12] analysis of the requirements. As an input, it converts the requirement sentence into words or tokens. Data should be simplified using simple English and less hyphens and punctuation in order to reduce the amount of data overhead. As shown in Figure 5, the first part of the ReSim defines this procedure as a subset of the software requirements [54].

5.2 Requirements Similarity measurement

In addition to calculating requirement similarity from requirements received from different stakeholders (see second part of ReSim Figure 5) and requirements of current projects, ReSim also measures requirement similarity from requirements of previously completed software projects [54].

Different well-known similarity measurement techniques used by the researchers to evaluate the similarity between the software requirements. Some of the methods used to measure this include dice, jaccard, and cosine coefficients, but recently researchers have proposed a few advanced algorithms based on WordNet, Word Embedding, and corpus statistics [54]. A semantic-based hybrid method proposed by Farooq [12] has been used in ReSim to measure similarity between the requirements (described in the section 2.3.3).

5.3 Design and Code Reusability

All those requirements which are similar will be incorporated into reusable components taken from a software repository using the design and code that they have created. According to some researchers, designing new software can be reduced by 30% if pre-existing product features can be reused and even 80% if existing product features are heavily repurposed [18, 54].

To a certain extent, this reduces project costs, time, effort, and resources. New modules of design and coding will be developed separately according to those requirements that do not have a similarity. It is intended that the configuration of design and code will be tested and evaluated in accordance with plan until implementation [54].

5.4 ReSim Architecture

In Figure 6, ReSim Framework's component architecture is shown. Sections 2 and 3 of the ReSim framework serve as the basis for this architecture, as shown in Figure 5. In the application of requirements requisitions, requirements are transformed into a shape that is ready for similarity matching. It matches requirements of new projects with requirements of already completed projects, based on similarity of the requirements extracted from the software repository database. In the software repository database, the result of the similarity matching will be saved [54].

Users will be suggested to use the software repository if there is a high degree of similarity in the matching, such as use cases, design modules, and coding schemes. However, even when there is an average or low level of similarity, the repository components can still be repurposed for developing different use cases, design modules, and coding schemes [54].

Algorithm: Algorithm for Calculating Requirements Similarity

Input: sRP // a repository of an existing projects
Input: sNP // a new project requirements
Output: msim // maximum similarity value
Variables: OPs // a single older project an exists in sRP
rr_new //the list of requirements for sNP
rr_old //the list of requirements for one OPs
asim[] //an existing projects requirement similarity
g1Max[] //maximum similarity of the sentences
Avgsim[] //an average similarity of a project requirements
1: OPs = 0, rr_old = ' ', rr_new = ' ';
2: rr_new=getAllrequirements (sNP)
3: for every OPs in sRP
4: rr_old=getAllrequirements (sRP [OPs])
5: for every requirement in rr_new
6: rr_new_description=getDescription (sNP)
7: rr_old_description=getDescription (rr_old)
8: rr_new_description =convertToWords (rr_new)
9: rr_old_description =convertToWords (rr_old)
10: for every sentence in rr_new_description
11: rr_new_sentence=getSentence (rr_new_description)
12: for every sentence in rr_old_description
13: rr_old_sentence=getSentence (rr_old_description)
14: asim[] = Sim **(**rr_new_sentence**,** rr_old_sentence**)**
15: End for
16: g1Max[] = getMax (asim[])
17: End for
18: Avgsim[] = Average (gMax[])
19: End for
20: End for
21: msim **=** getMax (Avgsim[])
22: End for
23: Return msim

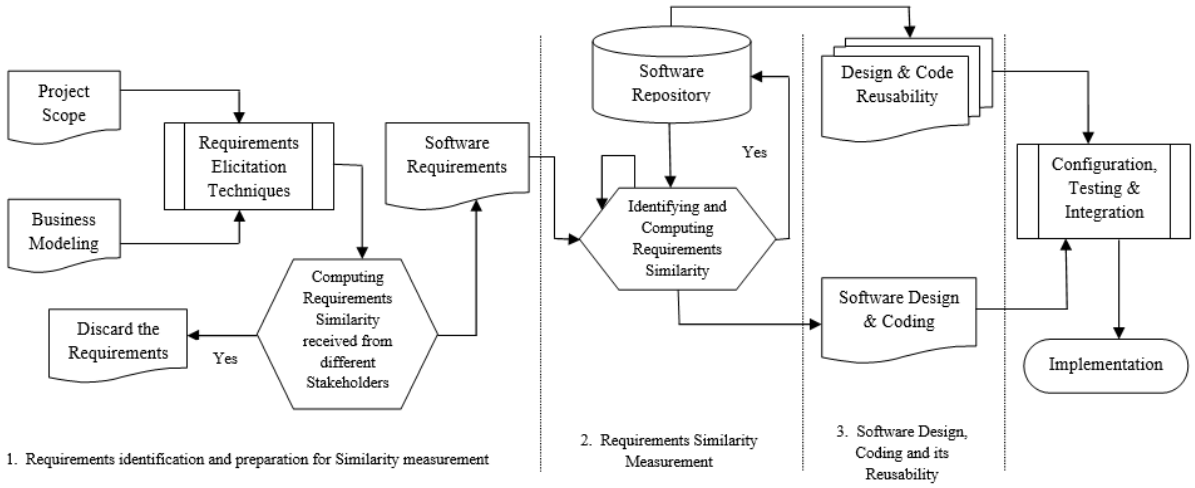

Fig. 5. ReSim Framework [54]

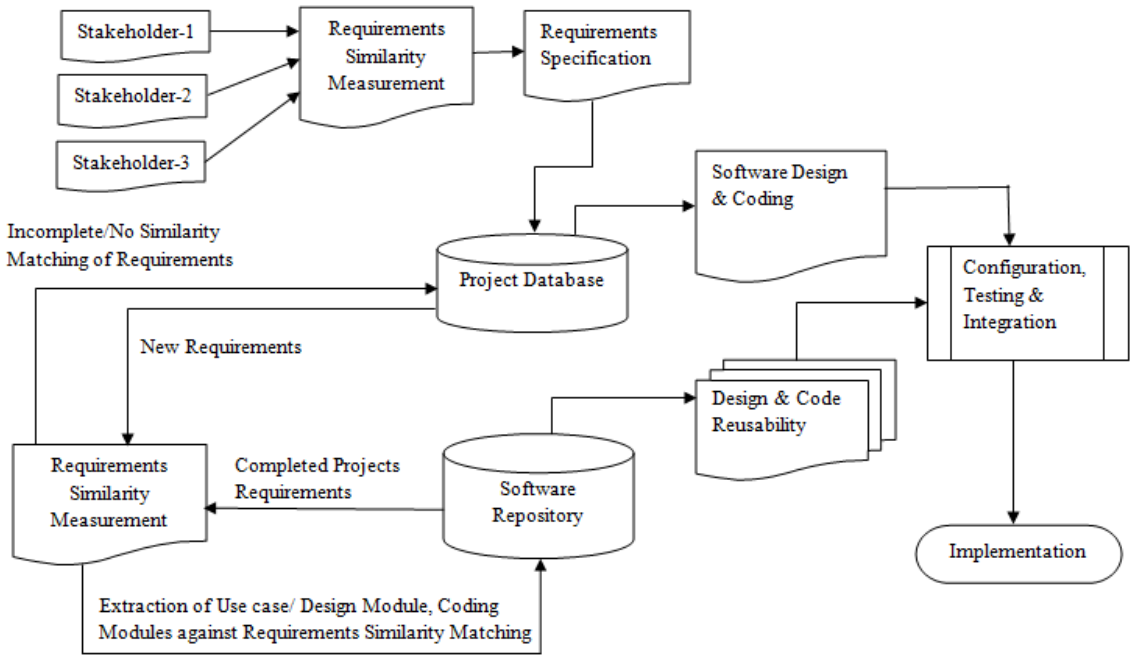

Fig. 6. ReSim Architecture [54]

6. Experimental Work and Results

With ReSim, our aim to provide a platform that would help users to find similar requirements between (or among) a new project and a previously completed project. Requirement similarity measurement will be used in choosing reusable components like design modules, coding modules, test cases, etc. when finding similarity between new and old requirements. In the end, it contributes significantly to project success by saving project costs, time, and other resources.

To measure the similarity between requirements in ReSim, we have used recently developed hybrid methodology [28] to measure similarity of software requirements. Our similarity measurement process consists of four different techniques such as Dice, Jaccard, cosine coefficient, and semantic-based hybrid approaches. With the help of these four different similarity measurement techniques, we have performed a comparative study to determine how efficient it is to measure requirements similarity [55].

Two samples of different requirements were taken in order to perform this experiment. There were 50 requirements in each sample. Both samples were taken from PURE (public requirements dataset) - a collection of 79 publicly available documents [55,62] describing natural language requirements obtained from the Internet. In this dataset, 34,268 sentences are included, and the dataset can be used to perform natural language processing tasks typical in requirements engineering, such as model synthesis, abstraction detection, and document structure analysis. Further annotations can make it work as a benchmark for many other tasks, including ambiguity detection, requirements

categorization, and identification of equivalent requirements. A random sample of 50 requirements was chosen from the PURE dataset [55,62]. A matching exercise was done between every requirement of one sample and every requirement of the other sample, so 2500 measurements were conducted in total [55].

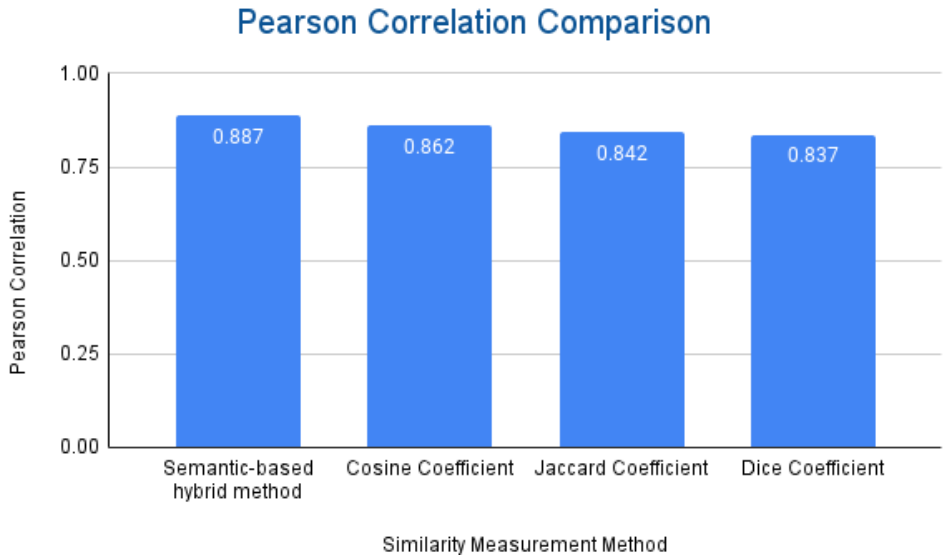

Fig. 7. Comparison of the semantic-based hybrid approach with other approaches with Pearson's correlation coefficient

A similarity measurement was performed on all words in each requirement. Table 2 shows a sample of data from these measures. In Figure 8 and 12, you can see a graphic representation of all four types of similarity measurements calculated for all requisites. Only results with a similarity matching greater than zero are considered in the graph. Out of 2500 similarities, there are 2227 when the value of matching exceeds zero. The resultant values were set in ascending order based on the similarity value before drawing the graph.

As a result, the Dice similarity measurement coefficient is greater than its Jaccard counterpart by 95.58 %. In comparison to Cosine and Jaccard, Cosine measures similarity better than Jaccard by over 96.66%. The value of Cosine is always greater than value of Dice measurement. In comparison to recently developed hybrid method and Cosine, hybrid method is 98.42% greater than Cosine. Only a few abnormal results (less than 1%) show that Jaccard's values are very high when measured against dice and cosine. Almost all other records seem to be normal. From these results, we can conclude that hybrid method like similarity measurements are more accurate than Dice, Jaccard, and Cosine, while Cosine is a better choice than Dice, and Jaccard is more accurate than Dice [55]. Linear regression model for Dice, Jaccard and Cosine similarity measurement against recently developed hybrid method are shown in the figure 9, 10 and 11.

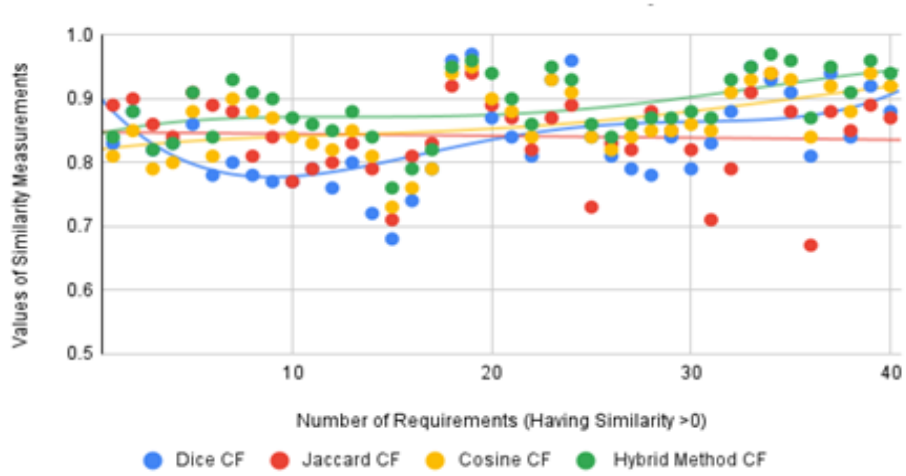

Fig. 8. Comparison of the number of requirements versus their similarity measurements using different similarity measurement approaches.

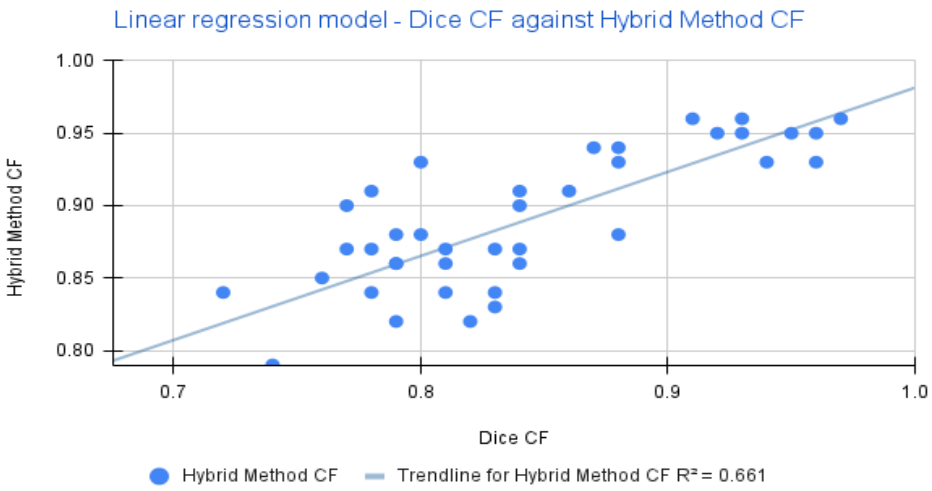

Fig. 9. Linear regression model - Dice CF against Hybrid Method CF

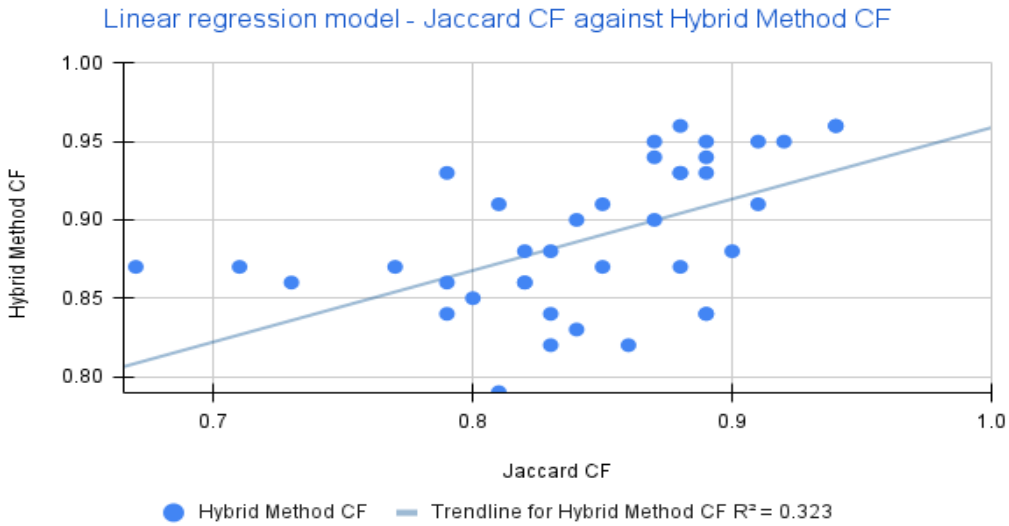

Fig. 10. Linear regression model - Jaccard CF against Hybrid Method CF

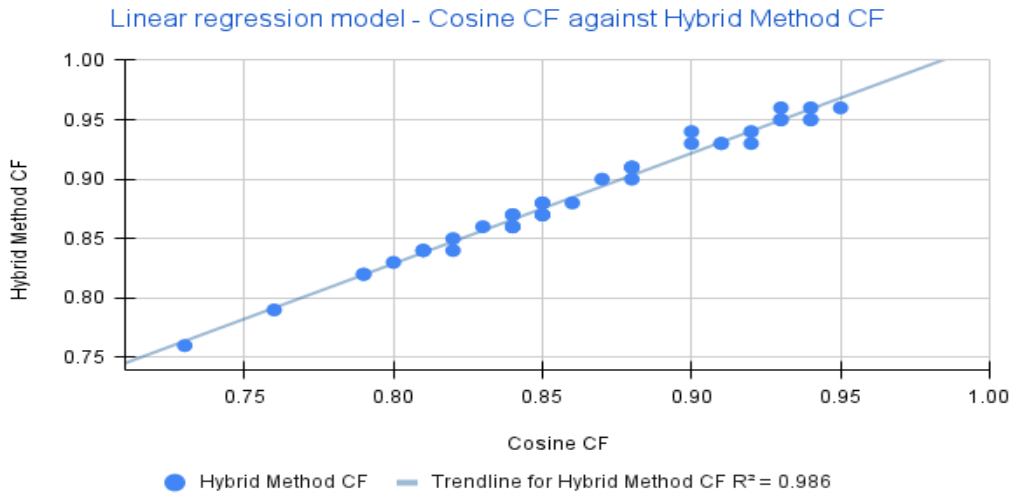

Fig. 11. Linear regression model - Cosine CF against Hybrid Method CF

Table 2. Similarity Measurement Data [55, 62]

#	Req1 ID	Req2 ID	Req1 WD	Req2 WD	SIMIL WD	Total WD	Dice CF [54,55]	Jaccard CF [54, 64, 65]	Cosine CF [54, 63]	Semantic-based Hybrid Method CF
1	42	9	39	46	61	85	0.83	0.89	0.81	0.84
2	34	13	29	43	60	72	0.88	0.90	0.85	0.88
3	42	30	39	42	59	81	0.82	0.86	0.79	0.82
4	42	13	39	43	58	82	0.83	0.84	0.80	0.83
5	34	30	29	42	58	71	0.86	0.91	0.88	0.91
6	34	9	29	46	57	75	0.78	0.89	0.81	0.84
7	34	12	29	37	51	66	0.80	0.88	0.90	0.93
8	42	12	39	37	49	76	0.78	0.81	0.88	0.91
9	34	19	29	33	42	62	0.77	0.84	0.87	0.90
10	44	9	13	46	42	59	0.77	0.77	0.84	0.87
11	44	13	13	43	41	56	0.79	0.79	0.83	0.86
12	34	18	29	37	40	66	0.76	0.80	0.82	0.85
13	44	30	13	42	39	55	0.80	0.83	0.85	0.88
14	34	25	29	32	37	61	0.72	0.79	0.81	0.84
15	42	19	39	33	36	72	0.68	0.71	0.73	0.76
16	34	33	29	28	35	57	0.74	0.81	0.76	0.79
17	44	12	13	37	35	50	0.79	0.83	0.79	0.82
18	42	25	39	32	34	71	0.96	0.92	0.94	0.95
19	42	40	39	31	34	70	0.97	0.94	0.95	0.96
20	42	33	39	28	34	67	0.87	0.89	0.90	0.94
21	23	13	11	43	34	54	0.84	0.87	0.88	0.90
22	28	13	11	43	34	54	0.81	0.82	0.84	0.86
23	42	20	39	32	33	71	0.93	0.87	0.93	0.95
24	31	13	26	43	33	69	0.96	0.89	0.91	0.93
25	42	18	39	37	32	76	0.84	0.73	0.84	0.86
26	34	20	29	32	32	61	0.81	0.83	0.82	0.84
27	2	13	16	43	32	59	0.79	0.82	0.84	0.86
28	38	30	17	42	32	59	0.78	0.88	0.85	0.87
29	34	40	29	31	31	60	0.84	0.85	0.85	0.87
30	38	13	17	43	31	60	0.79	0.82	0.86	0.88
31	34	45	29	43	30	72	0.83	0.71	0.85	0.87
32	31	30	26	42	30	68	0.88	0.79	0.91	0.93
33	38	9	17	46	30	63	0.95	0.91	0.93	0.95
34	23	30	11	42	30	53	0.93	0.94	0.94	0.96
35	28	30	11	42	30	53	0.91	0.88	0.93	0.96
36	31	9	26	46	29	72	0.81	0.67	0.84	0.87
37	2	9	16	46	29	62	0.94	0.88	0.92	0.93
38	2	30	16	42	29	58	0.84	0.85	0.88	0.91
39	23	9	11	46	29	57	0.92	0.89	0.94	0.95
40	28	9	11	46	29	57	0.88	0.87	0.92	0.94

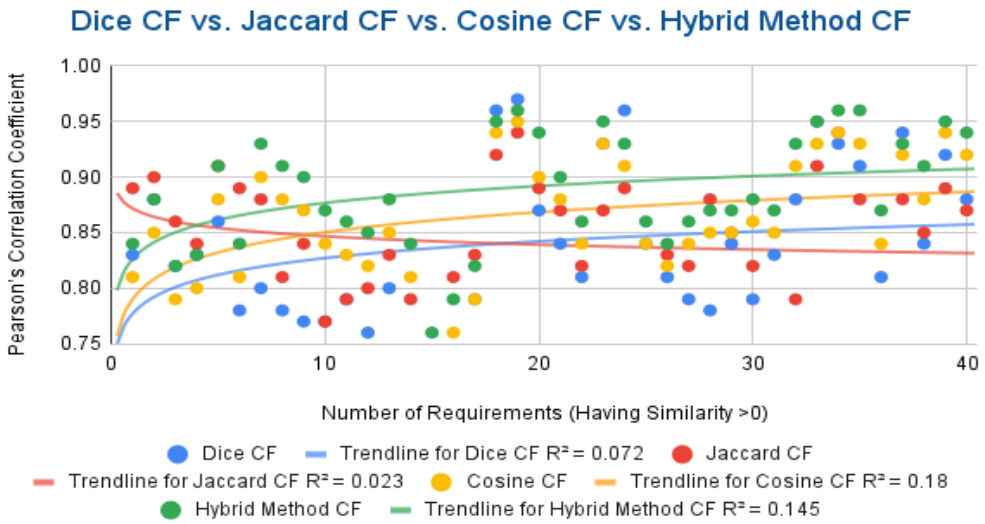

Fig. 12. Comparison of Hybrid Method CF vs. Dice and Jaccard and Cosine CF using number of requirements

7. Conclusion and Future Work

The main objective of this paper is to propose a framework "ReSim". ReSim measures similarity between requirements by simply comparing textual requirements received from stakeholders and the requirements of the running project (or software) with those from previously completed projects. In this study, a recently developed hybrid methodology [12] to assess similarity between requirements.

A hybrid methodology recently developed by Farooq [12] used to analyze the similarity of requirements written by the same person (or group). A requirement written in natural language could also vary in style and word selection from person to person, so this may also affect similarity measurements. Ontology's and synonyms are another important direction for research. Various words in natural languages have the same meaning and might be used in conjunction with one another.

Thus, if two requirements are written with different synonyms but have the same meaning and scope, and then there should be a high probability of similarity matching between them.

When tested on the PURE dataset [55, 62], ReSim was found to outperform the existing approaches, giving the best results both for functional and non-functional requirements.

In the future, we will extend this work by presenting in detail all phases of the framework's implementation, in addition to continuing to study similarity measurements in user queries and results. ReSim allows us to build an interactive system for measuring the semantic similarity between a new incoming project and the completed projects in the repository, so that we can identify the components that can be reused. The reusability phase will also be addressed and traceability techniques will be implemented to identify the components to be used again. Instead of building each reusable component from scratch, we can customize them to save effort and time. This proposed system will be integrated with the Reusability Module to provide a comprehensive system for measuring the similarity of requirements components and for facilitating reusability of software. The framework will test on a bigger number of projects; also, it will be tested with some real life projects.

References

[1] Madhavan, J., Bernstein, P. A., Doan, A., and Halevy, A. 2005. Corpus-Based Schema Matching. In Proceedings of the 21st international Conference on Data Engineering (April 05 - 08, 2005). ICDE. IEEE Computer Society, Washington, DC, 57-68.

[2] Seidl, T. and Kriegel, H. 1997. Efficient User-Adaptable Similarity Search in Large Multimedia Databases. In Proceedings of the 23rd international Conference on Very Large Data Bases (August 25 - 29, 1997), San Francisco, CA, 506-515.

[3] Ortega, M., Rui, Y., Chakrabarti, K., Mehrotra, S., and Huang, T. S. 1997. Supporting similarity queries in MARS. InProceedings of the Fifth ACM international Conference on Multimedia (Seattle, Washington, United States, November 09 - 13, 1997). Multimedia '97. ACM, New York, NY, 403- 413.

[4] Braunmueller, B., Ester, M., Kriegel, H. P. and Sander, J. 2000. Efficiently Supporting Multiple Similarity Queries for Mining in Metric Databases. In Proceedings of the 16th International Conference on Data Engineering, February 28 - March 03, 2000, Washington, DC, 256.

[5] Berchtold, S. and Kriegel, H. 1997. S3: similarity search in CAD database systems. SIGMOD Rec. 26, 2 (Jun. 1997), 564-567.

[6] Faloutsos, C., Ranganathan, M., and Manolopoulos, Y. 1994. Fast subsequence matching in time series databases. SIGMOD Rec. 23, 2 (Jun. 1994), 419-429.

[7] Badue, C. S., Baeza-Yates, R., Ribeiro-Neto, B., Ziviani, A., and Ziviani, N. 2007. Analyzing imbalance among homogeneous index servers in a web search system. Inf. Process. Manage. 43, 3 (May. 2007), 592-608.

[8] White M, Tufano M, Vendome C, Poshyvanyk D (2016) Deep learning code fragments for code clone detection. In: International Conference on Automated Software Engineering (ASE), pp. 87–98. IEEE
[9] Arora C, Sabetzadeh M, Briand L, Zimmer F (2016) Automated extraction and clustering of requirements glossary terms. Trans Softw Eng 43(10):918–945
[10] H. Samosir, D. Siahaan. 2019. Identifying Requirements Association Based on Class Diagram Using Semantic Similarity. Institute for Research and Community Services, Udayana University, Bali-Indonesia, VOL. 10, NO. 1 APRIL 2019
[11] Wael H. Gomaa, Aly A. Fahmy. 2013. A survey of text similarity approaches. *International Journal of Computer Applications, 68(13)*.
[12] Farooq Ahmad, Dr. Mohammad Faisal. 2022. A novel hybrid methodology for computing semantic similarity between sentences through various word senses, International Journal of Cognitive Computing in Engineering 3 (2022) 58–77, Volume 3, June 2022, Pages 58-77
[13] Eyal-Salman H, Seriai AD, Dony C (2013) Feature-to-code traceability in a collection of software variants: Combining formal concept analysis and information retrieval. In: 2013 IEEE 14th International Conference on Information Reuse & Integration (IRI), pp. 209–216
[14] Shatnawi A, Seriai A, Sahraoui H, Ziadi T, Seriai A (2020) Reside: Reusable service identification from software families. JSS 170:110748
[15] Bourque, P., & Fairley, R. E. 2014. Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0. *IEEE Computer Society Press*.
[16] Steinberger, R., Pouliquen, B., & Hagman, J. 2002. Cross-lingual document similarity calculation using the multilingual thesaurus eurovoc. *In Computational Ling*uistics and Intelligent Text Processing (pp. 415-424). Springer Berlin Heidelberg.
[17] Michael Keating. 2012. Reuse Methodology Manual for System-On-A-Chip Designs. *Springer Science & Business Media*.
[18] The Design Reuse Benchmark Report Seizing the Opportunity to Shorten Product Development February 2007. Aberdeen Group © 2007.www.304.ibm.com/jct03004c/tools/cpeportal/fileserve/download0/103335/ AberdeenPLM.pdf?contentid=10333.
[19] Dante Carrizo a, , Oscar Dieste b , Natalia Juristo. 2014. Systematizing requirements elicitation technique selection. *Information and Software Technology, 56(6), 644-669.*
[20] Apoorva Mishra and Deepty Dubey. 2013. A comparative study of different software development life cycle models in different scenarios. *International Journal of Advance research in computer science and management studies.*
[21] Swarnalatha k s, GN Srinivasan, Meghana Dravid, Raunak kasera, Kopal Sharma. 2014. A Survey on Software Requirement Engineering for Real Time Projects based on Customer Requirement. *International Journal of Advanced Research in Computer and Communication Engineering.*
[22] Pierre Bourque and Richard E. Fairley. 2014. Guide to the software engineering body of knowledge (SWEBOK (R)):Version 3.0. *IEEE Computer Society Press.*
[23] J. Natt och Dag, B. Regnell, P. Carlshamre, M. Andersson and J. Karlsson. Evaluating Automated Support for Requirements Similarity Analysis in Market-Driven Development. 7th Int. Workshop on Requirements Engineering: Foundation for Software Quality, June 4-5 2001, Interlaken, Switzerland.
[24] A. Rodr ́guez and M. Egenhofer. Comparing geospatial entity classes: an asymmetricand context-dependent similarity measure. International Journal of Geographical Information Science, 18(3):229–256, 2004.
[25] M. Raubal. Formalizing conceptual spaces. In A. Varzi and L. Vieu, editors, Formal Ontology in Information Systems, Proceedings of the Third International Conference, FOIS 2004.
[26] C. d'Amato, N. Fanizzi and F. Esposito. A semantic similarity measure for expressive description logics. In CILC 2005, Convegno Italiano di Logica Computazionale, Rome, Italy, 2005.
[27] R. Ara ́ujo and H. S. Pinto. Semilarity: Towards a modeldriven approach to similarity. International Workshop on Description Logics, volume 20, pages 155–162. Bolzano University Press, June 2007.
[28] K. Janowicz, P. Mau ́e, M. Wilkes, S. Schade, F. Scherer, M. Braun, S. Dupke and W. Kuhn. Similarity as a Quality Indicator in Ontology Engineering. Formal Ontology in Information Systems: Proceedings of the Fifth International Conference (Fois 2008).
[29] C. Fellbaum, WordNet. Wiley Online Library, 1998.
[30] Rada Mihalcea, Courtney Corley and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. *Association for the Advancement of Artificial Intelligence.*
[31] Papias Niyigena, Zhang Zuping, Weiqi Li and Jun Long. 2015. Efficient Pairwise Document Similarity. Computation in Big Datasets. *International Journal of Database Theory and Application.*
[32] David S ́anchez, Montserrat Batet, David Isern and Aida Valls. 2012. Ontology-based semantic similarity: A new feature-based approach. *Expert Systems with Applications.*
[33] Goldstone, R. L. (1994). Similarity, interactive activation, and mapping. *Journal of Experimental Psychology: Learning, Memory, and Cognition.*
[34] Monika Lanzenberger, Jennifer Sampson. 2008. Making Ontologies Talk: Knowledge Interoperability in the Semantic Web. *IEEE Intelligent Systems.*
[35] Y. Matsuo and M. Ishizuka, "Keyword extraction from a single document using word co-occurrence statistical information," International Journal on Artificial Intelligence Tools, vol. 13, no. 01, pp. 157–169, 2004.
[36] D. Bollegala, Y. Matsuo, and M. Ishizuka, "Measuring semantic similarity between words using web search engines." www, vol. 7, pp. 757–766, 2007.
[37] H. Rubenstein and J. B. Goodenough, "Contextual correlates of synonymy," Communications of the ACM, vol. 8, no. 10, pp. 627–633, 1965.
[38] Farooq Ahmad, Mohd. Faisal, 2021. Comparative Study of Techniques used for Word and Sentence Similarity, 8th International Conference on Computing for Sustainable Global Development (INDIACom), IEEE Xplore, New Delhi, India, 17-19 March 2021

[39] Ziadi T, Frias L, da Silva MAA, Ziane M (2012) Feature identification from the source code of product variants. In: 2012 16th European Conference on Software Maintenance and Reengineering, pp. 417–422. IEEE
[40] Shatnawi A, Seriai AD, Sahraoui H (2017) Recovering software product line architecture of a family of object-oriented product variants. J Syst Softw 131:325–346
[41] Salton, G. 1989. Automatic Text Processing: the Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley Longman Publishing Co., Inc.
[42] Zhao L, Alhoshan W, Ferrari A, Letsholo KJ, Ajagbe MA, Chioasca EV, Batista-Navarro RT (2021) Natural language processing for requirements engineering: a systematic mapping study. ACM Comput Surv. https:// doi. org/ 10. 1145/ 34446 89
[43] Fernández DM, Wagner S, Kalinowski M, Felderer M, Mafra P, Vetrò A, Conte T, Christiansson MT, Greer D, Lassenius C et al (2017) Naming the pain in requirements engineering. Empir Softw Eng 22(5):2298–2338
[44] Ferrari A, Dell'Orletta F, Esuli A, Gervasi V, Gnesi S (2017) Natural language requirements processing: a 4d vision. IEEE Ann History Comput 34(06):28–35
[45] Kassab M, Neill C, Laplante P (2014) State of practice in requirements engineering: contemporary data. Innov Syst Softw Eng 10(4):235–241
[46] Borg M, Runeson P, Ardö A (2014) Recovering from a decade: a systematic mapping of information retrieval approaches to software traceability. Emp Softw Eng 19(6):1565–1616. https:// doi. org/ 10. 1007/ s10664- 013- 9255-y
[47] Cleland-Huang J, Gotel OC, Huffman Hayes J, Mäder P, Zisman A (2014) Software traceability: trends and future directions. In: Future of Software Engineering Proceedings, pp. 55–69
[48] Gervasi V, Zowghi D (2014) Supporting traceability through affinity mining. In: International Requirements Engineering Conference (RE), pp. 143–152. IEEE
[49] Guo J, Cheng J, Cleland-Huang J (2017) Semantically enhanced software traceability using deep learning techniques. In: International Conference on Software Engineering (ICSE), pp. 3–14. IEEE
[50] Falessi D, Cantone G, Canfora G (2011) Empirical principles and an industrial case study in retrieving equivalent requirements via natural language processing techniques. Trans Softw Eng 39(1):18–44
[51] Arora C, Sabetzadeh M, Goknil A, Briand LC, Zimmer F (2015) Change impact analysis for natural language requirements: An nlp approach. In: International Requirements Engineering Conference (RE), pp. 6–15. IEEE
[52] Borg M, Wnuk K, Regnell B, Runeson P (2016) Supporting change impact analysis using a recommendation system: an industrial case study in a safety-critical context. IEEE Trans Softw Eng 43(7):675–700
[53] Abdullah Azzam, Yudi Priyadi, Jati Hiliamsyah Husen. 2021. Similarity Software Requirement Specification (SRS) Elicitation Based on the Requirement Statement Using Text Mining on the MNC Play Inventory Management Application. 2021. 4th International Conference of Computer and Informatics Engineering (IC2IE), IEEE Xplore, Depok, Indonesia, 14-15 Sept. 2021
[54] Muhammad Ilyas, Josef Küng. 2009. A Similarity Measurement Framework for Requirements Engineering. *Fourth International Multi-Conference on Computing in the Global Information Technology (2009).*
[55] Muhammad Ilyas, Josef Küng. 2009. A comparative analysis of Similarity Measurement Techniques through SimReq Framework. *FIT* '09: Proceedings of the 7th International Conference on Frontiers of Information Technology (2009). Article No.: 47, Pages 1–6
[56] Wael H. Gomaa, Aly A. Fahmy. 2013. A Survey of Text Similarity Approaches. International Journal of Computer Applications (0975 – 8887), Volume 68– No.13, April 2013
[57] Mohammad Mahmoud Tarawneh. 2017. Software Requirements Classification using Natural Language Processing and SVD. International Journal of Computer Applications (0975 – 8887), Volume 164 – No 1, April 2017
[58] Fatma A. Mihany, Hanan Moussa, Amr Kamel, Ehab Ezat. 2016. A Framework for Measuring Similarity between Requirements Documents. INFOS '16: Proceedings of the 10th International Conference on Informatics and Systems (2016), Pages 334–335.
[59] Fatma A. Mihany, Hanan Moussa, Amr Kamel, Ehab Ezzat, Muhammad Ilyas. 2016. An Automated System for Measuring Similarity between Software Requirements. AMECSE '16: Proceedings of the 2nd Africa and Middle East Conference on Software Engineering (2016), Pages 46–51
[60] Abbas M, Ferrari A, Shatnawi A, Enoiu EP, Saadatmand M (2021), Is requirements similarity a good proxy for software similarity? an empirical investigation in industry. In: The 27th international working conference on requirements engineering: foundation for Software Quality, pp. 3–18. Springer International Publishing
[61] Muhammad Abbas, Alessio Ferrari, Anas Shatnawi, Eduard Enoiu, Mehrdad Saadatmand & Daniel Sundmark. 2022. On the relationship between similar requirements and similar software. A case study in the railway domain, Springer Nature, 18 January 2022
[62] Alessio Ferrari, Giorgio O. Spagnolo, Stefania Gnesi. 2017. PURE: A Dataset of Public Requirements Documents. IEEE 25th International Requirements Engineering Conference (RE). September 2017
[63] Li, B. and Han, L. (2013) "Distance weighted cosine similarity measure for text classification," *Intelligent Data Engineering and Automated Learning – IDEAL 2013*, pp. 611–618. Available at: https://doi.org/10.1007/978-3-642-41278-3_74.
[64] Hancock, J.M. (2004) "Jaccard distance (Jaccard Index, Jaccard similarity coefficient)," *Dictionary of Bioinformatics and Computational Biology* [Preprint]. Available at: https://doi.org/10.1002/9780471650126.dob0956.
[65] Achananuparp, P., Hu, X. and Shen, X. (no date) "The evaluation of sentence similarity measures," *Data Warehousing and Knowledge Discovery*, pp. 305–316. Available at: https://doi.org/10.1007/978-3-540-85836-2_29.

Authors' Profiles

Farooq Ahmad is currently working as Assistant Professor and Pursuing PhD in Computer Application from Department of Computer Application at Integral University, Lucknow India. He has received his master degree from Harcourt Butler Technological Institute, Kanpur in 1999. He has diverse background and around 22 years of total experience in IT software development industry and education with 15 years in industry & 7 years in academic. He is a Oracle Database 12c SQL Certified Associate. He has written a chapter "Virtualization in the Cloud" in the book published by Cambridge Scholars Publishing, UK, 2020, ISBN: 978-1-5275-3556-5 and a chapter "Challenges Generated by the Integration of Cloud and IoT" in the book published by CRC Press, New York, 2022**,** ISBN: 9781003155577**.** He has published quality research papers in Journals, National and International Conferences of repute. He is contributing his knowledge and experience as member of technical committee in various international Journals/Conferences of repute. His research interest includes Computer Programming, Software Engineering, RDBMS, NoSQL, Web Technologies, Artificial Intelligence, Natural Language Processing, Information Retrieval, Big Data Analytics, Cloud Computing and Machine Learning.

Dr. Mohammad Faisal is currently working as an Associate Professor and Head of the Department in the Department of Computer Application, Integral University, Lucknow, India. He has more than 17 years of teaching & research experience. His areas of interest are Software Engineering, Requirement Volatility, Distributed Operating System, Cyber Security, and Mobile Computing. He has published a book "Requirement Risk Management: A Practioner's Approach" published by Lambert Academic Publication, Germany, ISBN: 978-3-659-15494-2. He has published quality research papers in Journals, National and International Conferences of repute. He is contributing his knowledge and experience as member of Editorial Board/Advisory committee and TPC in various international Journals/Conferences of repute. Dr. Mohammad Faisal is an active member of various professional bodies IAENG, CSTA, ISOC-USA, EASST, HPC, ISTE, IAENG, UACEE and many more.

How to cite this paper: Farooq Ahmad, Mohammad Faisal, "Assessing Similarity between Software Requirements: A Semantic Approach", International Journal of Information Engineering and Electronic Business(IJIEEB), Vol.15, No.2, pp. 38-53, 2023. DOI:10.5815/ijieeb.2023.02.05