International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 11, No.1 (Jan-2022)

http://dx.doi.org/10.12785/ijcds/110150

Interdependency Aware Qubit and Brownboost Rank Requirement Learning for Large Scale Software Requirement Prioritization

Raghavendra Devadas[1] **and Nagaraj G Cholli**[2]

[1]*Research Scholar, ISE research center, RV College of Engineering, affiliated to Visveswaraya Technological University, Assistant Professor, CSE Dept, Presidency University, Bengaluru, India.*
[2]*Associate Professor, Dept of ISE, RV College of Engineering, Bengaluru, India.*

Received 23 Jun. 2021, Revised 11 Sep. 2021, Accepted 4 Jan. 2022, Published 20 Jan. 2022

Abstract: Requirements Prioritization (RP) is very indispensable and laborious phase in the course of requirement management of software engineering. Numerous research works have been conducted in the prioritization of small size requirements. However, problems are said to occur while considering large set software project requirements. In order to address the issue, in this paper we present the novel method called the Interdependency-aware Qubit and BrownRoost Rank (IQ-BR) method to prioritize the huge number of requirements. Optimization is a model that identifies the optimal requirements from a set of probable functions with respect to their attributes or requirements. Quantum Optimization is the familiar optimization algorithms is used in the IQ-BR. The novelty of the work lies in the use of the Interdependency-aware Qubit Requirement Selection algorithm and BrownBoost Rank Requirement Prioritization Learning model. An Interdependency-aware Qubit Requirement Selection algorithm is used to address the requirements prioritization issues to handle volatile and interdependencies among requirements during RP. With the optimal requirement selection results, BrownBoost Rank Requirement Prioritization Learning is finally applied to rank the requirements based on the BrownBoost Rank function. The proposed IQ-BR and existing methods are discussed with different factors such as requirement prioritization accuracy, requirement prioritization time, true positive rate and false-positive rate with respect to different functional and non-functional requirements. The observed results show superior performance of our proposed IQ-BR method when compared to state-of-the-art methods.

Keywords: Requirement Prioritization, Software engineering, Interdependency-aware, Qubit, BrownBoost Rank, Quantum Optimization, Requirement Selection, Attributes

1. INTRODUCTION

The requirement engineering is the most important problem in software engineering to select the prerequisites that are advantageous and examined initially for fulfillment. The decision-making process is a more complicated process for large-scale software project requirements. It is important to obtain, examine and prioritize requirements, economic value, and benefit of any software system that is needed to be enhanced which heavily depends upon affirmation of user's requirements. One of the solutions to create the correct solution is to prioritize between numerous options. One of the tricky prioritizing software requirements are to have thousands of requirements. It is feasible that users need to integrate the entire software project. But the entire features or requirements are not important.

An Intuitionistic Fuzzy Approach (IFS) was proposed in [1] with the purpose of bestowing an interactive support system for bringing out users' and developers' initial ranking decisions in a collective manner. The IFS was utilized to assist the users whereas the interrelationship analysis, requirement slicing, and backtracking was utilized in supporting developers' views with the aid of a weighted page rank algorithm. Finally, with these two mechanisms, a ranking was said to be generated collaboratively to assist requirement prioritization, therefore producing accurate results with minimum errors.

Despite improvements observed with minimum errors, due to large scale requirement prioritization to be done with minimum resources, the requirement prioritization accuracy was not improved with less time. Therefore, the IQ-BR method is designed by segregating the requirements into two types as functional and non-functional requirements, both interdependency and volatile requirements are handled.

The achievement of the requirement prioritization task heavily hangs on distinct restrictions and predominant char-

E-mail address: raghudevdas@gmail.com, nagaraj.cholli@rvce.edu.in

acteristics are maneuver through the users during prioritization. A method called, Dependency Based Collaborative Requirement (CDBR) was proposed in [2] for reducing the divergence of belief among users and developers to the efficient association and the healthier estimation of prioritization outcomes with minimum processing time.

Despite improvements observed both in terms of processing time and prioritization results, the errors (false positive rate) involved during the prioritization process were not focused on. To address this issue, BrownBoost Rank Requirement Prioritization Learning model is applied with the aid of the BrownBoost Rank function which not only reduces the false positive rate involved in RP but also improves the true positive rate using Universe of Optimal Requirement pairs.

The novel contributions of this work are listed below.

A novel method called the IQ-BR method is introduced to prioritize a large number of requirements. It is designed with the Interdependency-aware Qubit Requirement Selection algorithm and BrownBoost Rank Requirement Prioritization Learning model. The Interdependency-aware Qubit Requirement Selection model is used to handle volatile and interdependence among requirements during large-scale requirement prioritization with aid of the Qubit function. The novelty of the BrownBoost Rank Requirement Prioritization Learning model is used in the IQ-BR method to precisely prioritize requirements and reduce the noisy requirement prioritization. Finally, a series of experiments were conducted to measure the performance analysis of the proposed IQ-BR method along with conventional methods based on various performance metrics.

The rest of the paper is divided into five sections. The literature survey is reported in Section 2. Section 3 describes our proposed method, IQ-BR. The research results with the aid of a table and graphical representations are shown in section 4. The paper is summarized in section 5.

2. LITERATURE SURVEY

The primary phases in requirement engineering are the Requirement Prioritization owing to the reason that not all requirements have identical customer satisfaction. As far as the project manager is concerned, the requirements that can be performed effortlessly and in a straightforward manner should be executed first. On the other hand, the financial manager is in help of the requirements that result in lesser cost. As a result, each requirement has distinct attentiveness based on the software to be developed. Hence, RP is a paramount factor in software products, wherein not prioritizing a software product may result in a complete failure.

The user requirement automation process was carried out in [3] Incomprehension of analytical RP provides quality minimization and disgruntled clients. A survey of certain critical issues of RP in Agile Software Development like scalability, complexity, uncertainty, dependency issues concentrated on the nonfunctional requirements was investigated in [4]

It is also analyzed that existing methods of RP like, grouping, validated learning works better in functional requirements. However, it is inadequate while handling non-functional requirements. With the purpose of handling this issue, a meta-model with value-related RP was proposed in [5] to prioritize both the requirements. The protectors of quality software systems, RP have seldom been utilized in hand-picking the most paramount requirements as discerned by customers. To date, numerous RP methods that apply various methods are investigated in earlier research. The advantages, disadvantages of these existing methods were analyzed in [6] where a detailed study of RP based on standard review guidelines was presented.

Regression analysis is pivotal in confer those changes made was not found in the unfavorable result. But analysis made using regression is found with high cost and time. Model-based Test Case Prioritization (MB-TCP) was proposed in [7] to perform prioritization. Constructing a better configurable system with user fulfillment as a constraint is a difficult task. In [8], a quantitative RP method for better configurable systems was introduced, therefore contributing to scalability.

In the past few years, Requirements Engineering (RE) include a swift increase inefficiently utilizing different Machine Learning (ML) techniques to address the issue faced by RE. The provoking problem is the detection and categorization of software requirements concerning Stack Overflow (SO). Systematic Literature Review (SLR) was used to concentrate the recognition and classification of the ML approach for software requirements recognition in [9]. Yet another method concentrating on the fault reduction during RP using the nature of bees was presented in [10]. A systematic mapping study covering aspect-oriented methodologies concerning Software Development Life Cycle (SDLC) was proposed in [11].

Research results materialized that none of the methods can be observed the best and precise than the other methods, owing to the reason that the selection purely is based on the result nature of the frequency of requirements intricate during prioritization. The study made in [12] addresses the advantages of current methods and also provides the disadvantages in detail. Formatting and summarizing the release set with requirements made in a prioritized fashion are considered to be a challenging task because the requirements contain their characteristics. Therefore Search-Based Software Engineering (SBSE) was introduced to address the above issues by utilizing meta-heuristics for identifying feasible solutions based on the objectives and constraints.

In [13] a method called, Verbal Decision Analysis (VDA) was proposed to improve the probabilities of addressing the Next Release Problem. Yet another meta-

heuristic technique using a Genetic Algorithm to address multi-objective functions was proposed in [14] to generate high-quality solutions. An exhaustive literature survey portrays that no requirements prioritizing techniques prioritizes dependent requirements.

After analyzing numerous techniques, a novel prioritization methodology was proposed with the purpose of prioritizing independent and independent requirements. Also, Analytic Network Process (ANP) was designed to prioritize the interdependent requirements in [15]. However, with large-scale requirement prioritization being the need of the hour, still challenging.

In [16], clustering techniques were employed for the large-scale prioritization of software. Moreover, a certain amount of uncertainty is said to be involved during RP. To address this issue, uncertainty-wise requirement prioritization was proposed in [17].The multiple perspective prioritization technique was applied in [18] to concentrate on the scalability aspect. Identification and requirement ranking using the Delphi technique was proposed in [19] to concentrate on the failure aspects involving RP.

On the Contrary, to address the issues in previous work, a novel method is designed to handle interdependence and volatile requirements to prioritize a large number of software requirements by employing the IQ-BR method. An elaborate description of the IQ-BR method is presented in the following sections

3. METHODOLOGY

Determining which, among a set of requirements, are to be taken into consideration first and in which sequence is a deliberate process concerning software development. This task is specifically referred to as the RP. In this section, a method called, IQ-BR method, which integrates requirement selection and requirement prioritization via ML techniques is proposed.The architecture of the proposed IQ-BR method is shown in Fig. 1

As shown in the Fig. 1, first, the Interdependency-aware Qubit Requirement Selection model is applied to the raw software requirement dataset to select computationally efficient requirements even in the case of interdependency and volatility. The second BrownBoost Rank Requirement Prioritization Learning model over the optimal requirement attributes is used, thus supporting an adaptive prioritization process.

A. *Interdependency-aware Qubit Requirement Selection model*

The requirement selection process refers to the collection of needs that has to be satisfied by the system under development. In other words, the requirement selection is measured as the number of actions carried out through software engineers regarding cost minimization and customer satisfaction maximization. Hence, requirement selection is one of these management tasks that determine

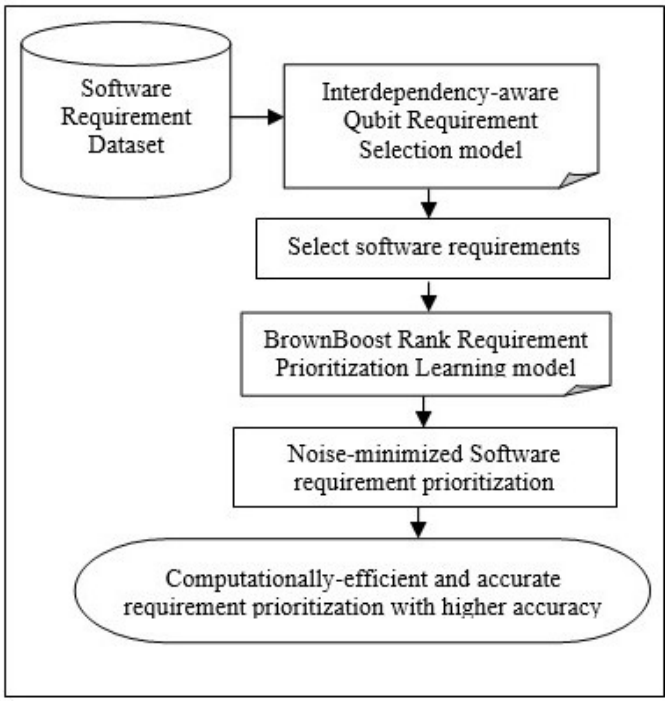

Figure 1. Architecture of proposed IQ-BR method

which requirements have to be taken into consideration with the available resources. To handle volatility and interdependence among requirements during large-scale RP, both functional 'FR' and nonfunctional requirements 'NFR' have to be considered. In this paper, we first propose an Interdependency-aware Qubit Requirement Selection model that aims to integrate the process of volatile handling and interdependence during RP on software projects Fig. 2 shows the block diagram of the Interdependency-aware Qubit Requirement Selection model.

As shown in Fig. 2, to develop a methodology for volatile large-scale requirement prioritization, initially the attributes or the requirements have to be identified. There are various attributes or requirements on which these volatile requirements are said to be prioritized. These volatile requirements are selected based on the various characteristics. In our work, volatility is handled by considering value interdependencies, that are determined to be present or not. Once all positive and negative value interdependencies are recognized, the requirement is assessed against each interdependency result.

To start with the 'FR' and 'NFR' are identified that are required to be prioritized. Let 'i' denote the candidate 'FR' and 'j' denote the candidate 'NFR'. Let us consider that there include 'm' candidate functional requirements '$FR_1..FR_m$' and 'n' nonfunctional requirements '$FR_1..FR_n$', which need to be prioritized. Also, all the functional and non-functional requirements are not necessary for all the customers, and hence according to the customers' objec-

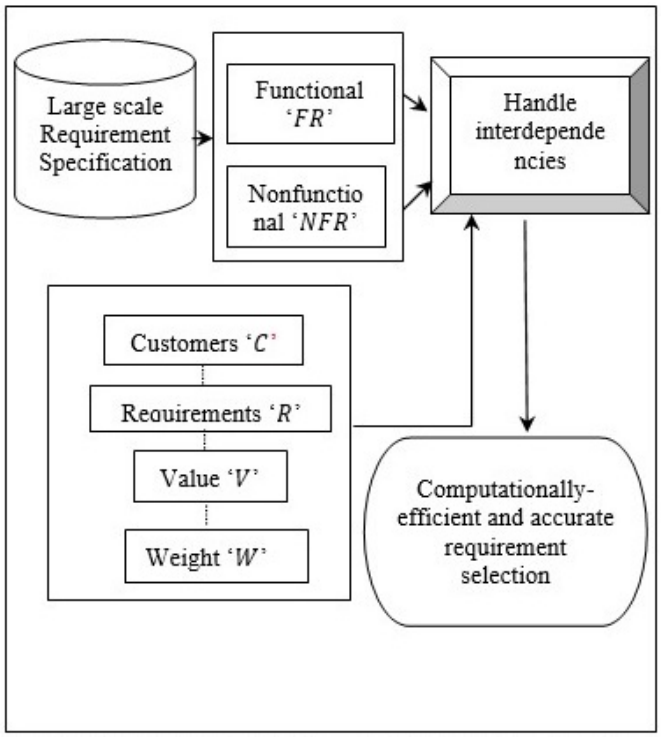

Figure 2. Block diagram of Interdependency-aware Qubit Requirement Selection model

tives, the requirements have to be analyzed.

The second step is obtaining a 'm*n' decision vector-matrix namely 'DVM', where the 'm' candidate functional requirements and 'n' candidate nonfunctional requirements are placed in the m*n matrix as given below.

$$DVM = \begin{vmatrix} NFR/FR & NFR_1 & NFR_2 & NFR_3 & \dots & NFR_n \\ FR_1 & \dots & \dots & \dots & \dots & \dots \\ FR_2 & \dots & \dots & \dots & \dots & \dots \\ FR_3 & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ FR_m & \dots & \dots & \dots & \dots & \dots \end{vmatrix} \quad (1)$$

In the next step, decision maker perceptions to ascertain significance degree for quantifying value interdependencies on the basis of user or customer preference for respective requirements involving pair 'FR' and 'NFR'.Let us consider a set of 'm' customers 'C=$C_1, C_2, \dots, C_m$', then for given set of customers let us further assume the set of requirements as 'R=$R_1, R_2, \dots, R_n$'. Each customer in turn possesses a proportion of significance on the basis of features or characteristics like, order equilibrium, terms and conditions in payment, trustworthiness and so on that can be disclosed via a weight factor. The weight then connected with each respective customer 'W=$W_1, W_2, \dots, W_m$,j∈[0,∈1]'. In order to execute each requirement of the respective customer over functional and non-functional requirements, resources like workforce, interfaces, utilities, plugins, and hardware setups are required that can be interpreted in terms of value. Then the value linked with each requirement for its execution is 'V=$V_1, V_2, \dots, V_n$'. Owing to the reason that entire requirements are not uniformly salient for the customers. Each customer 'C_j' allocates a rating for requirement 'R_j' denoted by 'rating(R_i, C_j)'. Then, the score 'S_i' for requirement 'R_i' is mathematically formulated as given below.

$$S_i = \sum W_i * rating(R_i, C_j) \quad (2)$$

In above equation 2, the score 'S_i' value is computed. 'W_i' indicates the weight. 'R_i' denotes the requirement, each customer 'C_j' assign a rating for requirement 'R_i' indicated by 'rating(R_i, C_j)'. Next, the two objectives to optimize are formulated as given below.

$$Min\,Fun_1 = \sum_{i=1}^{n} V_i * DVM_i \quad (3)$$

$$Max\,Fun_2 = \sum_{i=1}^{n} S_i * DVM_i \quad (4)$$

With the above two objective functions as given in equations 3 and 4, interdependencies between requirements are obtained using the Eels function. 'S_i' denotes the score.'V_i' Represents the value, and decision vector-matrix namely 'DVM_i'. The purpose of applying this function is due to its efficiency in obtaining positive and negative factors during RP. This interdependency measure is estimated as given below.

$$\tau_{ij} = Prob\left(R_i|R_j\right) - Prob\left(R_i|R_j'\right), \alpha_{ij} \in [-1, 1] \quad (5)$$

From the above equation 5, the sign of 'τ_{ij}, provides the strength of value interdependency from requirement 'R_i' to requirement 'R_j'. Also 'Prob($R_i|R_j'$)' and 'Prob($R_i|R_j$)' denote the positive strength and negative strength from requirement 'R_i' to requirement 'R_j' respectively. If the resultant value is greater than zero, refers to the positive relationship between 'R_i & R_j'.

On the other hand, if the resultant value is less than zero, refers to the negative relationship between 'R_i & R_j'. Finally, volatile and interdependent requirements are selected using Quantum Optimization functions. The Interdependency-aware Qubit Requirement Selection utilizes a representation based on Q-bit, for the probabilistic value interdependency representation employing the concepts of qubits, probably in '1-state' or in '0-state' as given below.

$$\left|\psi >= \alpha\left(\tau_{ij}\right)\right| 0 > +\beta\left(\tau_{ij}\right)|1 > \quad (6)$$

From the equation 6, '$\alpha_i\left(\tau_{ij}\right)$' and '$\beta_i\left(\tau_{ij}\right)$' denotes the large scale or a large number of requirements specifying the probability magnitudes of the respective states.

$$\left|\alpha_i\left(\tau_{ij}\right)\right|^2 + \left|\beta_i\left(\tau_{ij}\right)\right|^2 = 1 \quad (7)$$

Finally, '$\left|\alpha\left(\tau_{ij}\right)\right|^2$' denotes the probability that the qubit will be found in the '0-state' and '$\left|\beta\left(\tau_{ij}\right)\right|^2$' denotes the probability that the qubit will be found in the '1-state'.

$$\left|\alpha\left(\tau_{ij}\right)\right|^2 + \left|\beta\left(\tau_{ij}\right)\right|^2 = 1 \tag{8}$$

A Q-bit individual is a string of m Q-bits presented as

$$\begin{pmatrix} \alpha_1\left(\tau_{ij}\right) & \alpha_2\left(\tau_{ij}\right) & \alpha_3\left(\tau_{ij}\right) & \ldots & \alpha_m\left(\tau_{ij}\right) \\ \beta_1\left(\tau_{ij}\right) & \beta_2\left(\tau_{ij}\right) & \beta_3\left(\tau_{ij}\right) & \ldots & \beta_m\left(\tau_{ij}\right) \end{pmatrix} \tag{9}$$

From the above equation 9, the relevant and optimal requirement selection is made for functional and non-functional requirements. The pseudo-code representation for Interdependency-aware Qubit Requirement Selection is given in algorithm 1. As given in the Interdependency-aware Qubit Requirement Selection algorithm, the objective is to optimize the requirement selection with maximum accuracy and minimum time even in case of interdependency between requirements and volatility. With this objective first, functional and non-functional requirements are split into two using decision vector matrix. Followed by which, interdependency between requirements is handled by employing Q-bit function, and error occurring due to the change in requirements (volatility) is addressed via Quantum Optimization function, therefore contributing to minimum software requirement time and software prioritization accuracy during the later stage.

B. BrownBoost Rank Requirement Prioritization Learning model

With the results of the optimal software requirement selection, the next step processes the RP using the BrownBoost Learning model that aids in precise decision-making for ordering a set of optimal software requirements. The BrownBoost Learning model provides a repeated prioritization process that operates single and multiple customer decision-makers and distinct ordering benchmarks. An eccentricity of this method is the utilization of ML to minimize the elicitation endeavor that is the aggregate of information required from customer's stakeholders, for achieving adaptive requirement prioritization.

Let us consider a bounded collection of Optimal Requirements OR='OR_1, OR_2, $\ldots, OR_n$' that has to be ranked, and for it defines the Universe of Optimal Requirement pairs UORP='$\left\{\left(OR_i, OR_j\right); i < j\right\}$ referring to the sequence relationships between two requirements from a customer priority. This is mathematically expressed as given below.

$$\phi\left(\mathrm{OR}_i, \mathrm{OR}_j\right);\ \phi : UORP \rightarrow \{-1, 0, 1\} \tag{10}$$

The results for the equation 10 in terms of the function '$\phi\left(OR_i, OR_j\right)$' is formulated as given.

$$\phi\left(\mathrm{OR}_i, \mathrm{OR}_j\right) = \begin{matrix} -1, if ORj \prec ORi \\ 1, if ORi \prec ORj \\ 0, No sequence requirement between ORi and ORj \end{matrix} \tag{11}$$

Algorithm 1 Interdependency-aware Qubit Requirement Selection algorithm

Input: Customers '$C = C_1, C_2, ..., C_m$', Requirements '$R = R_1, R_2, ..., R_n$', Weight '$W = W_1, W_2, ..., W_m$', Value '$V = V_1, V_2, ..., V_n$'

Output: Computationally-efficient and accurate software requirement 'OR'

Begin:

Step 1: For eac00h Customers 'C' with Requirements 'R', Weight 'W' and Value 'V' return
Step 2: Obtain decision vector matrix for functional and non-functional requirements as in equation (1)
Step 3: Estimate the score as in equation (2)
Step 4: Formulate two objectives as in equation (3) and equation (4)
Step 5: Estimate interdependency measure as in equation (5)
Step 6: Estimate Q-bit, for the probabilistic value interdependency between requirements as in equation (6)
Step 7: **if** '$\tau_{ij} > 0$'
Step 8: Exists positive relationship between 'R_i & R_j'
Step 9: Return relevant functional requirements
Step 10: **Endif**
Step 11: **if** '$\tau_{ij} < 0$'
Step 12: Exists negative relationship between 'R_i & R_j'
Step 13: Return relevant nonfunctional requirements
Step 14: **Endif**
Step 15: **End for**
Step 16: **End**

From the equation 11, the third condition states Non-sequenced Requirement Pair and on the other hand, the first and second condition states a Sequenced Requirement Pair (SRP) that is formulated as given below.

$$\varphi_\gamma = \left\{\left(\mathrm{OR}_i, \mathrm{OR}_j\right); i < j | \phi\left(\mathrm{OR}_i, \mathrm{OR}_j\right) \neq 0\right\} \tag{12}$$

Finally, the ranking function is defined using BrownBoost (function) to be the rankings obtained by the features or attributes of each requirement, such as the significance for the user. The objective behind the use of the BrownBoost function to the SRP is that noisy OR_s will be over and over mislabeled by the weak hypotheses whereas the non-noisy OR_s will be correctly labeled. Hence, only noisy OR_s will be discarded whereas non-noisy OR_s will contribute to the final classifier. This in turn aids in the improvement of the true positive rate and minimization of the false-positive rate.

The BrownBoost Ranking function starts with the setting of weights 'W_i' for each OR_s, 'OR_i' as given below.

$$W_i(\mathrm{OR}_i) = e^{-\frac{\varphi_\gamma}{c}} \tag{13}$$

From the equation 13, 'φ_γ' refers to the margin of sample Optimal Requirement pairs obtained at the time 'c'.

$$\begin{aligned} C_i : OR \rightarrow \{-1, +1\}, \\ such\ that \sum W_i(\mathrm{OR}_i) C_i(\mathrm{OR}_i)\, \mathrm{OR}_j > 0 \end{aligned} \tag{14}$$

According to the classification results, as stated in the above equation 14, the ranking of the optimal requirements pairs is analyzed, and accordingly, requirement prioritization is achieved with minimum false positive rate. The pseudo code representation of BrownBoost Rank Requirement Prioritization Learning is given below.

As given in algorithm 2, BrownBoost Rank Requirement Prioritization Learning algorithm, the objective is on concentrating on two different metrics, true positive and false positive rate. To attain these objectives, first, Universe of Optimal Requirement Pairs is via sequence relation between two requirements. Followed by which the BrownBoost function is applied to the hypothesized results, therefore reducing noisy requirement prioritization.

Algorithm 2 BrownBoost Rank Requirement Prioritization Learning

Input: Optimal Requirements '$OR = OR_1, OR_2, ..., OR_n$'
Output: Noise-minimized requirement prioritization
Step 1: **Initialize** Universe Optimal Requirement pairs 'UORP'
Step 2: **Initialize** 'i' training Optimal Requirement pairs, time 'c'
Step 3: **Begin**
Step 4: **For** each Optimal Requirements 'OR'
Step 5: Measure Universe of Optimal Requirement pair as in equation (10) and equation (11)
Step 6: Measure Sequenced Requirement Pair as in equation (12)
Step 7: Assign weights 'W_i' for each OR_s, 'OR_i' as in equation (13)
Step 8: Formulate classifier as in equation (14)
Step 9: **Return** (minimum-noise requirement prioritization results)
Step 10: **End for**
Step 11: **End for**
Step 12: **End**

4. IMPLEMENTATION AND RESULTS DISCUSSION

The illustration of the IQ-BR method is experimented using dataset [20]. The dataset consists of a total of 625 requirements, out of which 255 are functional and 370 are nonfunctional requirements. We use the dataset to evaluate its performance which is presented in Python. Followed by which the results obtained via the utilization of numerous quality indicators, accuracy, time, true positive and false positive for different numbers of requirements are presented. Moreover, also a detailed comparison is presented with the purpose of comparing the outcome of the IQ-BR method with existing methods Intuitionistic Fuzzy Approach (IFS) [1] and Dependency Based Collaborative Requirement (CDBR) [2] respectively. Experiments of IQ-BR method is performed with Intel(R) Core (TM) i_3 CPU 2.13 GHz processor with 4 GB RAM.

The performance analysis of four different parameters, requirement prioritization accuracy, requirement prioritization time, true positive rate and false positive rate for IQ-BR, Intuitionistic Fuzzy Approach (IFS) [1] and Dependency Based Collaborative Requirement (CDBR) [2] are discussed in detail with the aid of table and graphical representations.

A. *Performance analysis of requirement prioritization accuracy*

One of the significant and foremost parameters for large scale-based requirement prioritization is the accuracy rate. In other words, requirement prioritization accuracy refers to the accurate rate with which the requirement prioritization is made. This is mathematically formulated as given below.

$$RPA = \sum_{i=1}^{n} \frac{R_{AP}}{R_i} \tag{15}$$

From the above equation 15 the requirement prioritization accuracy 'RPA' is measured based on the requirements considered for simulation 'R_i' and the requirement accurately prioritized 'R_AP'. It is computed in units of percentage (%).

Fig. 3 shows the requirement prioritization accuracy made with respect to 600 different functional and non-functional requirements obtained at different time intervals. From the figure, it is inferred that for software requirements between 60 and 360 the accuracy saw a decreasing trend whereas for software requirements between 420 and 540 the accuracy saw an increasing trend. However, simulations conducted for 60 requirements, the requirement accurately prioritized using IQ-BR was found to be 57, 55 using IFS, and 54 using CDBR. With these three results, the accuracy rate was observed to be 95%, 91.66%, and 90% respectively, therefore seeing an improvement using IQ-BR. The reason behind the accuracy improvement was owing to the application of the Interdependency-aware Qubit Requirement Selection model. By applying this model, accurate features or requirements are selected using Eels function. With this function, relevant features or requirements from the software requirements are selected in an accurate manner, therefore improving the requirement prioritization accuracy using IQ-BR by 4% compared to IFS and 2% compared to CDBR.

TABLE I. TABULATION FOR REQUIREMENT PRIORITIZATION ACCURACY

Requirements	Requirement Prioritization Accuracy (%)		
	IQ-BR	*IFS*	*CDBR*
60	95	91.66	90
120	94.85	90.25	88.95
180	94.25	89.55	88.35
240	94	89.35	87.25
300	93.15	89	87
360	93	90.15	86.35
420	94.25	90.85	88.25
480	94.55	91.35	89.15
540	95	90.15	88.05
600	93.15	89.55	87

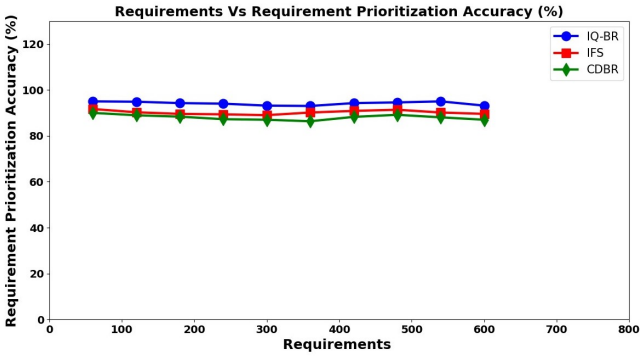

Figure 3. Graphical representation of requirement prioritization accuracy

B. *Performance analysis of requirement prioritization time*

The second metric of consideration is the requirement prioritization time. A significant amount of time is said to be consumed during the process of requirement prioritization. In other words, requirement prioritization time refers to the time consumed in prioritizing the requirements. This is mathematically formulated as given below.

$$RPT = \sum_{i=1}^{n} RP_i * Time\ \ (RP) \tag{16}$$

From the equation 16, the requirement prioritization time 'RPT' is measured based on the requirements analyzed during simulation 'RP_i' and the time consumed in prioritizing the requirements '$Time\ \ (RP)$' It is computed in units of milliseconds (ms).

Fig. 4 illustrates the requirement prioritization time with respect to 600 distinct functional and nonfunctional requirements via decision vector matrix. The requirement prioritization time is directly proportional to the requirements used for simulation. To simply say, increasing the number of requirements causes an increase in the requirement prioritization time. But simulations performed with 60 requirements consumed an overall time of 0.185ms when applied with IQ-BR and 0.215ms, 0.235ms when applied with [1] and [2]. The reason behind the minimization of requirement prioritization time using IQ-BR was due to the application of the Interdependency-aware Qubit Requirement Selection algorithm. By applying this algorithm, initially, both the functional and non-functional requirements were split into two distinct vectors by utilizing a decision vector matrix. Next, interdependency between requirements was handled using Q-bit function, therefore contributing to minimum software requirement time using IQ-BR by 36% compared to IFS and 44% compared to CDBR.

TABLE II. .TABULATION OF REQUIREMENT PRIORITIZATION TIME

Requirements	Requirement Prioritization Time (%)		
	IQ-BR	*IFS*	*CDBR*
60	11.1	12.9	14.1
120	13.35	18.15	21.55
180	15.95	25.35	30.25
240	20.55	31.45	33.55
300	25.35	50.15	55.25
360	27.25	55.35	60.15
420	30.55	58.25	65.35
480	35.35	60.55	70.25
540	41.55	63.15	75.35
600	50	65	80.25

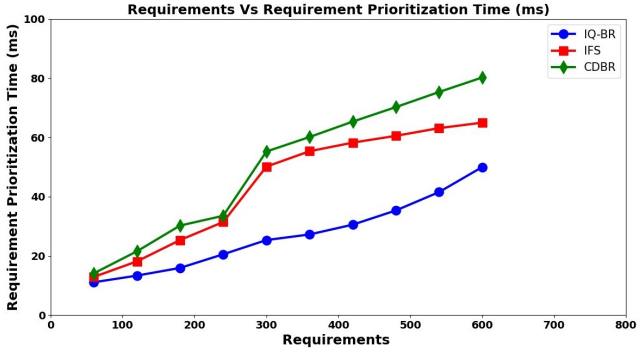

Figure 4. Graphical representation of requirement prioritization time

C. *Performance analysis of true positive rate*

TPR is defined as the potentiality of a screening test for detecting a true positive. In other words, the true positive rate in our work estimates the ratio of positive values (i.e., requirement prioritization correctly made) identified in a given raw software requirement dataset.

$$TPR = \sum_{i=1}^{n} \frac{TP}{TP + FN} \tag{17}$$

In equation 17, 'TPR', are computed depended on true positive 'TP' (i.e., requirement prioritization correctly made)

and false negative 'FN' (i.e., incorrectly requirement prioritization) respectively. The true positive rate is estimated in units of percentage (%).

Fig. 5 illustrates the true positive rate concerning 600 distinct functional and non-functional requirements. The TPR refers to the possibility where an actual prioritization of requirements has been made. Also, a steady curve was observed using three different methods, IQ-BR, IFS, and CDBR. However, simulations performed with 60 software requirements of both functional and nonfunctional saw 55 requirement prioritizations correctly made using IQ-BR, 51 and 48 made using IFS and CDBR, therefore resulting in a true positive rate of 91.66%, 85%, and 80% respectively. From this result, the true positive rate or the requirement prioritization correctly made using IQ-BR was comparatively better than IFS and CDBR. The reason behind the improvement was owing to the application of the BrownBoost Rank Requirement Prioritization Learning model. By applying this model, Universe of Optimal Requirement pairs was first evaluated and according to these results, Sequenced Requirement Pair (SRP) and Non-sequenced Requirement Pair were obtained. With this, the true positive rate using IQ-BR was found to be comparatively better than IFS by 6% and CDBR by 7% respectively.

TABLE III. .TABULATION OF TPR

Requirements	Requirement Prioritization Time (%)		
	IQ-BR	*IFS*	*CDBR*
60	91.66	85	80
120	90.35	84.85	79.55
180	90	84.25	79.35
240	89.75	84.15	79
300	89	84	78.55
360	88.55	83.55	78.3
420	88.15	83.25	78
480	88	83	77.55
540	87.35	82.15	77.25
600	87	82	77

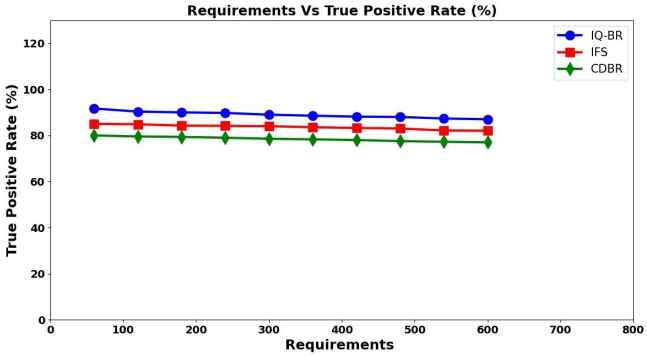

Figure 5. Graphical representation of TPR

D. Performance analysis of false positive rate

During the process of requirement prioritization, when performing large-scale requirement analysis false positive rate is said to occur.Here, FPR is defined as the probability of incorrectly eliminating the null hypothesis. In other words, FPR is measured as the percentage ratio between negative events (i.e., normal requirements wrongly prioritized) and the total actual negative events (i.e., the total number of requirements). This is mathematically expressed as given below.

$$FPR = \frac{FP}{FP + TN} * 100 \tag{18}$$

In equation 18 'FPR', is computed depending on the no. of false positives 'FP' and no. of true negatives 'TN' respectively. It is estimated in units of percentage (%).

Fig. 6 given illustrates the false positive rate for 600 different software requirements considered for analyzing the large-scale software requirement prioritization. As not all the software requirements are necessary for all the users or customers, requirements are first selected. During this process, a small number of requirements are also falsely rejected and therefore resulting in an error or false positive rate. As shown in the Fig. 6, a small portion of the increase in false positives was observed using all three methods. However, a simulation conducted with 50 requirements found 2 falsely rejected using IQ-BR, 4 and 7 requirements rejected using IFS and CDBR. From this, the false positive rate was found to be 3.33%, 6.66%, and 11.66% using the three methods. The improvement of false-positive rate using IQ-BR was due to the application of the BrownBoost Ranking function during the ranking operation. These accurate rankings were made, therefore reducing the false positive rate using IQ-BR by 32% compared to IFS and 58% compared to CDBR.

TABLE IV. .TABULATION OF FPR

Requirements	Requirement Prioritization Time (%)		
	IQ-BR	*IFS*	*CDBR*
60	3.33	6.66	11.66
120	3.85	6.85	11.85
180	4	7	12
240	4.25	7.15	12.35
300	4.85	7.3	12.55
360	5	7.45	13
420	5.75	7.65	13.45
480	6.35	7.85	13.85
540	8.15	9.25	14
600	9.25	11.35	14.35

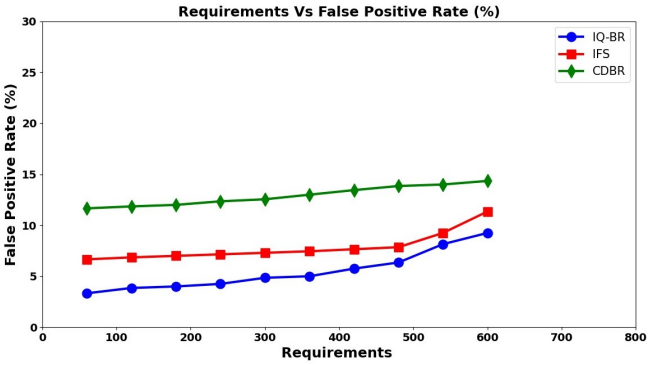

Figure 6. Graphical representation of TPR

5. Conclusions and Future Work

The elaborate descriptions of software requirement prioritization for largescale software projects. The IQ-BR method is used to permit the practitioners to concurrently prioritizing requirements. It brings out the prioritized indices of functional and non-functional requirements independently. The major innovation of the IQ-BR method in contrast to the conventional method is to handle interdependence and volatile requirements for large-scale software projects considering requirements cooperatively during the prioritization. The proposed IQ-BR method is validated by conducting experiments and comparing the results with IFS and CDBR. We concentrated specifically on four metrics, requirement prioritization accuracy, requirement prioritization time, TPR, and FPR. The experimentation is performed using 600 functional and non-functional requirements. From the result, IQ-BR method is comparatively better than the IFS and CDBR with respect to the time consumed and accuracy in prioritizing the requirements. Also, IQ-BR showed a significant improvement in terms of both true positive and false positive rates.

In future work we consider the Pugh Trapezoidal Fuzzy and Gradient Reinforce Learning (PTF-GRL) to handle uncertainty and test suite execution issue among different stakeholders for large scale software requirement prioritization.

Acknowledgment

This research was supported by Visvesvaraya Technological University, Jnana Sangama, Belagavi -590018 for Grant of financial assistance.

References

[1] A. Gupta and C. Gupta, "A novel collaborative requirement prioritization approach to handle priority vagueness and interrelationships," *Journal of King Saud University-Computer and Information Sciences*, 2019.

[2] A. Gupta and C. Gupta, "Cdbr: A semi-automated collaborative execute-before-after dependency-based requirement prioritization approach," *Journal of King Saud University-Computer and Information Sciences*, 2018.

[3] A. Sapunkov and T. Afanasieva, "Software for automation of user requirements prioritization," in *Proceedings of the 2019 2nd International Conference on Geoinformatics and Data Analysis*, 2019, pp. 1–5.

[4] N. H. Borhan, H. Zulzalil, and N. M. A. Sa'adah Hassan, "Requirements prioritization techniques focusing on agile software development: A systematic literature," *International Journal of Scientific and Technology Research*, vol. 8, no. 11, pp. 2118–2125, 2019.

[5] M. Masood, F. Azam, M. W. Anwar, and A. Amjad, "Defining metamodel for value-oriented requirement prioritization technique," in *Proceedings of the 2019 7th International Conference on Computer and Communications Management*, 2019, pp. 72–77.

[6] F. Hujainah, R. B. A. Bakar, M. A. Abdulgabber, and K. Z. Zamli, "Software requirements prioritisation: a systematic literature review on significance, stakeholders, techniques and challenges," *IEEE Access*, vol. 6, pp. 71 497–71 523, 2018.

[7] M. L. Mohd-Shafie, W. M. N. Wan-Kadir, M. Khatibsyarbini, and M. A. Isa, "Model-based test case prioritization using selective and even-spread count-based methods with scrutinized ordering criterion," *PloS one*, vol. 15, no. 2, p. e0229312, 2020.

[8] A. Ali, Y. Hafeez, S. Hussain, and S. Yang, "Role of requirement prioritization technique to improve the quality of highly-configurable systems," *IEEE Access*, vol. 8, pp. 27 549–27 573, 2020.

[9] A. Ahmad, C. Feng, M. Khan, A. Khan, A. Ullah, S. Nazir, and A. Tahir, "A systematic literature review on using machine learning algorithms for software requirements identification on stack overflow," *Security and Communication Networks*, vol. 2020, 2020.

[10] A. Ramamohanreddy, "A regression test based on the test case prioritization techniques by using the nature of bees," *International Journal of Innovative Technology and Exploring Engineering*, 2019.

[11] F. Pinciroli, J. L. B. Justo, and R. Forradellas, "Systematic mapping study: on the coverage of aspect-oriented methodologies for the early phases of the software development life cycle," *Journal of King Saud University-Computer and Information Sciences*, 2020.

[12] M. D. Hamad, A. Elsayed, M. El-Borai, and W. M. Abdelmoez, "Software product requirements prioritization techniques: Hardly easy," *International Journal of Artificial Intelligence and Mechatronics*, vol. 4, no. 6, pp. 198–209, 2016.

[13] P. A. Barbosa, P. R. Pinheiro, F. R. Silveira *et al.*, "Selection and prioritization of software requirements applying verbal decision analysis," *Complexity*, vol. 2019, 2019.

[14] M. Marghny, H. El-Hawary, and W. H. Dukhan, "An effective method of systems requirement optimization based on genetic algorithms," *Inf Sci Lett*, vol. 6, no. 1, pp. 15–28, 2017.

[15] J. Khan, I. ur Rehman, L. Ali, S. Khan, Y. H. Khan, and I. J. Khan, "Requirements prioritization using analytic network process (anp)," *International Journal of Scientific & Engineering Research*, vol. 7, no. 11, 2016.

[16] P. Achimugu, A. Selamat, and R. Ibrahim, "A clustering based technique for large scale prioritization during requirements elicitation," in *Recent Advances on Soft Computing and Data Mining*. Springer, 2014, pp. 623–632.

[17] H. Zhang, M. Zhang, T. Yue, S. Ali, and Y. Li, "Uncertainty-

wise requirements prioritization with search," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 30, no. 1, pp. 1–54, 2020.

[18] I. Ibriwesh, S.-B. Ho, I. Chai, and C.-H. Tan, "Prioritizing solution-oriented software requirements using the multiple perspective prioritization technique algorithm: an empirical investigation," *Concurrent Engineering*, vol. 27, no. 1, pp. 68–79, 2019.

[19] J. Iqbal, R. B. Ahmad, M. Khan, S. Alyahya, M. H. Nizam Nasir, A. Akhunzada, and M. Shoaib, "Requirements engineering issues causing software development outsourcing failure," *PloS one*, vol. 15, no. 4, p. e0229785, 2020.

[20] Souvik, "Software requirements dataset," https://www.kaggle.com/iamsouvik/software-requirements-dataset.

Raghavendra Devadas is a research scholar in ISE research center, RV College of Engineering, Bangalore. He is working as Assistant Professor in the department of Computer Science, Presidency University, Bangalore. He is pursuing PhD in Software Engineering specialization. His area of interest includes software engineering, data analytics, fuzzy logic, soft computing, R programming. He has presented papers in both national and international conferences and also, he is a life member of ISTE.

Nagaraj Cholli is working as Associate Professor at R V college of engineering, Bangalore, India. His area of interest includes Software Engineering, Data Mining, Cloud Computing, Image Processing, AI and ML, AR and VR. He has authored many national and international journals. He is member of prestigious professional bodies like Institute of Electrical and Electronics Engineers, Indian Society for Technical Education, science and Engineering Institute and also, he has filed 6 patents under his name. He is also involved in many research and consultancy projects