ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 4, April 2018

Efficient and Noiseless Mining of Software Engineering Data with Optimized Effort using TLBO: A Novel Approach

Gurtej Singh Ubhi[1], Jaspreet Kaur Sahiwal[2]

M.Tech Scholar, Department of Computer Science, Lovely Professional University, Phagwara, Punjab, India[1]

Assistant Professor, Department of Computer Science, Lovely Professional University, Phagwara, Punjab, India[2]

Abstract: The domain of software engineering has been emerging as a challenging field in other domains of expertise. With the rapid evolution and the presence of software based tools have proved the applicability of data mining in the field of software engineering. This field is linked with the application of data mining methods to offer valuable insights into how to advance software engineering processes and software itself, supporting decision-making. The main rationale here is to convey the role of software engineering as a method so as to grab the attention of our community that can prove as an attractive leeway for data mining applications and to show how data mining can considerably add to software engineering research. Utilizing entrenched information mining strategies, professionals and analysts can investigate the capability of this important information keeping in mind the end goal to better deal with their activities and to create higher-quality programming frameworks that are conveyed on time and within spending plan or budget. This paper offer an approach that how data mining can make a noteworthy involvement to the success of current software engineering efforts and will tend to provide some set of recommendations that is meant to increase the success of experts participation and model satisfactoriness

Keywords: Software Mining, Teaching Learning Based Optimization, MMRE, Artifacts.

I. INTRODUCTION

It is said that the software systems are complex in nature and they are tedious to conceptualize. This convolution is linked by obscure dependencies and dissimilar programming styles and it not only slows development and safeguarding activities but also causes faults and a defect that eventually increases the cost of software Due to hefty and multipart data that is generated day by day at a high rate, so as a result of this the data mining is introduced in software engineering [1]. Using well-established data mining methods, engineers and researchers have started exploring the probable usage of this valuable data to better manage their projects and to generate higher quality software systems that are delivered within budget and stipulated time period. Software engineers uses the concepts of data mining to look into deeper the previously unknown and unique data statistics within a set of collected data. Data mining tools are useful in predicting the future trends and behaviors. which are useful for engineers to take practical knowledge driven decisions in the future aspects as well. Software data have the ability to provide valuable insights into how to improve software engineering processes and software itself, supporting decision-making. Regardless of the capable results that are achieved in the field of data mining for software engineering but there is frequently a lack of argument on the part of software engineering practitioners amidst the data mining approaches.

II. SOFTWARE MINING

The tern software mining is defined as the way of application of mining methods in the field of software engineering. This task includes the wide range of extraction of knowledge from the engineering data and using it for the improvement of the process that relates to the software development. In the real terms the software mining is a combination of three major concepts. [2]. i) The Goals: This defines the set of tasks that are tangible in nature. For instance this could include design, coding and monitoring at the runtime. These tasks could be technical [3] or people oriented [4]. ii) The Input Data: The input data may be from the variety of the sources. This could be the combination of the static analyzers, code review repositories. To exemplify, it consists the natural language processors for the written documents [5]. Further it could also contain the variety of inputs like Data flow diagrams , Use case Diagrams , Project detail documentations etc. iii) The Techniques: The techniques could be machine learning based. It can also be composed of the neural networks, clustering approaches. In addition this may also includes the regression approaches which could be a combination of both adaptive and model approach as well. This mainly differs in terms of the

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
IJARCCE
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 4, April 2018

parameters that are defined at the levels for the distribution of the qualitative aspects.

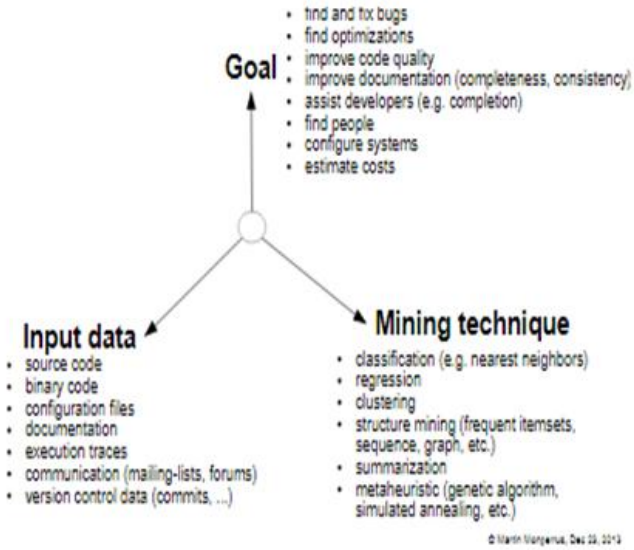

Figure 1. Fishbone diagram for Software Artifacts.

As it is clear from the Fig 1.that input data, techniques and the goals are mutually exclusive to each other. So each part of the given set of input data directly or indirectly dependents upon the mining techniques and it in turn fulfill the goals that have been set at the beginning of it.

II. THE EVALUATION PROCESS

As it is found that the software engineers or more precisely the set of professionals that use software as a part of their daily transaction has gathered them in gigantic numbers with an hope to use the same in future for the betterment of the processes and utilities. Furthermost the software mining has evolved as a major helping hand for the better exploration of the data. It is closely related to the process of the information mining for the better exploration of the engineering data that is aligned to the software domain.

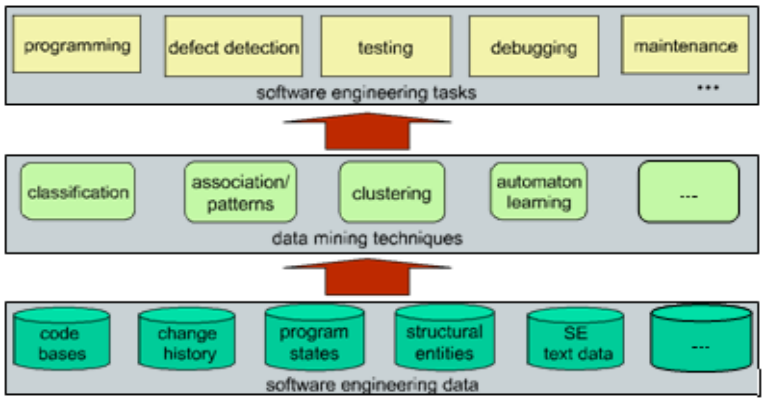

Figure 2. General Idea of Mining Software Engineering Data

From the above figure it is clear that software engineering data is directly linked with the data mining techniques that are imbibed for software engineering tasks.

III. LEVELS OF SOFTWARE ENGINEERING

This is directly allied to the theory of reverse engineering. Software mining addresses structure, behavior as well as the data processed by the software system. This might occur at a range of stages:

- Program level: This is defined at the level of the program execution
- Blueprint prototype level: This is defined at structural level of the software.
- Call graph level: This is the level which comprises the procedures along with associations that marks the various associations and links.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 4, April 2018

- Architectural level: This level is a combination of sub-systems with their interfaces and shows the various mapping among the interfaces.
- Data level: At this level data mapping and their dependencies are defined and organized.
- Application level: This is utmost level that is at the second point of hiérachies and allows for the various transactions.
- Business level: This at the topmost level of hierarchy that defines the roles and depicts organizational objectives.

IV. THE RELATION BETWEEN KNOWLEDGE DISCOVERY AND SOFTWARE ENGINEERING

Software Mining is ever growing domain that constitutes two major focal goals like Prediction and Description. Prediction in general is linked with the foundation of finding the future trends or the estimation of the future values using the target variable. Description on the other hand is linked with the formation of the clear prediction towards are more clear direction. A good description is also a matter of the fact that is consist better analysis of the data behavior in many folds. The relevance importance of the prediction and description may vary for different means for software mining. The connection of Data mining with software engineering tasks can be easily achieved by illuminating in much detail and analyzing in depth about the software artifacts and processes that relates both the data and facts. Based on data mining methods one can extort associations among software projects.

A handful of data mining methods has been planned and employed to gratify the requirements of different applications. These help to accomplish a set of data mining functionalities to spot and portray interesting patterns of knowledge extracted from a data set. They not only help in looking the useful facts but also allows for finding the set of functionalities that are of utmost usage.

Following is the brief description of the main data mining tasks and how they can be employed for the purpose of software engineering

1) Clustering – Unsupervised Learning Techniques: This is one of the most employed methods in data mining which is applied for the need of finding the hidden patterns and exploratory analysis of the data without any labeled responses. This can also be used in finding the inferences from the dataset without any guidance.

2) Classification – Supervised Learning Techniques: Classification is known as the method which classifies the data item or the dataset into a predefined classes or label. The main aim of the classification is to look for the target class in the set and it is generally achieved by a classifier.

3) Frequent Pattern Mining and Association Rules: This rule finds all the underlying "correlations" surrounded by the attributes or more precisely the features in the data set. A frequent pattern could be a subsequence, substructure or a set of items. It is quite important as it discloses the hidden and crucial properties of a data item.

4) Data Characterization and Summarization: It is the process that involves the [6] summarization of the data characteristics of definite class of data. This data is composed on the basis of user demands or needs. It is generally based upon the analysis of the attribute relevance. It can also be defined as zooming out the broader layer of the issue to drill down the solution. In general, it is the method of developing the consecutive layers of synopsis based data in an evaluation database.

5) Change and Deviation Detection: This looks for finding or looking out the changes in the data from the view of the calculated values. It is also sometimes referred to as anomaly detection or outlier detection in which the set of the patterns that do not confirm to the dataset or the patterns are detected and confined using the algorithms. This is sometimes also called as noise

V. SOFTWARE ENGINEERING DATA

In the mechanics of the data science as the software forms the crucial part of any organization. The data that is employed in the software engineering could be in terms of logs, test cases, use case diagrams, code blocks and so on. Each level of the software engineering process takes into some set of inputs and process it as a output which serves as a artifacts . Following are the various kind of the software engineering data for which the concepts of mining could be utilized.

Documentation: This involves the collection of enormous data which includes the archives of licenses, textual data and information related to contracts. The mining tool is very efficient in this case as it can help in larger landscape to

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
IJARCCE
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 4, April 2018

provide useful insights about the frequent patterns and allow the user to decide the best fitness function to be used at the time of the evaluation.

Source code: This includes the code that forms the foundation of the implementation part of any software. The code could be a primary element of the reusability and enhancing the maximum coverage in terms of cost and optimization.

Issue Tracking and Bug Database: This is the database that consist the record of all errors and bug that has been reported during the time of testing and performing the QA of the software. Machine learning methods have been employed to look for the fault tolerance methods to make the code and software more immune to the bugs and errors.

Execution Traces: This includes the important data that shows the steps of execution and to retrace if any fault occurs. Various mining methods have been employed to look any malicious task hampering the normal execution of the software.

VI. DATA MINING FOR THE PURPOSE OF SOFTWARE ENGINEERING

The data mining and software engineering are orthogonal to each other. In broad terms, with the application of data mining methods, the researchers can look into the potential usage of data used in software engineering and use the mining that can prove fruitful in better supervision of projects and to yield higher quality software systems that are delivered on time and on plan. In short, these mining methods not only help to reduce the execution time but also allow mining the efforts, cost and other important parameter.

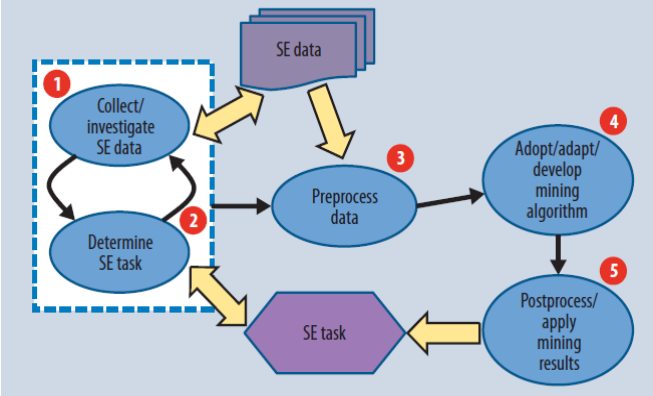

Fig3. Methodology for mining the Software Engineering Data

VII. CLASSIFIERS AND SOFTWARE MINING

The classifier is defined as the method that maps the input of the numerical data to a defined class or category. It is generally achieved by the applications of the algorithms. Below is the list of some classifiers that are utilized the software mining process. The methodology of classification is a defined as a supervised learning approach in which the computer program learns from the data input given to it in the form of some numerical facts and then uses this for the process of learning so that it can put into a category of new observation.

VIII. OPTIMIZING MINING METHODS AND SOFTWARE PROJECT ESTIMATION

Mining graphs can be used as a tool for the purpose of optimization. Example SE graph data that consists static or dynamic call graphs as well as program dependence graphs [7] where edges stand for data or control dependence and nodes symbolize statements. Program executions are meant for the evaluations of various predicates throughout a program. The further can be used to deploy the software on the basis of the efforts and predictions.
The software model that is employed for the effort estimation is composed of the following parameters.

- List significant or critical cost drivers.
- Generate a scaling model for every cost driver.
- Locate projects with similar environments.
- Contrast the project with previous familiar projects.
- Evaluate the project whether it is feasible inside the budget constraints.
- Incorporate the critical features in an iterative manner.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 4, April 2018

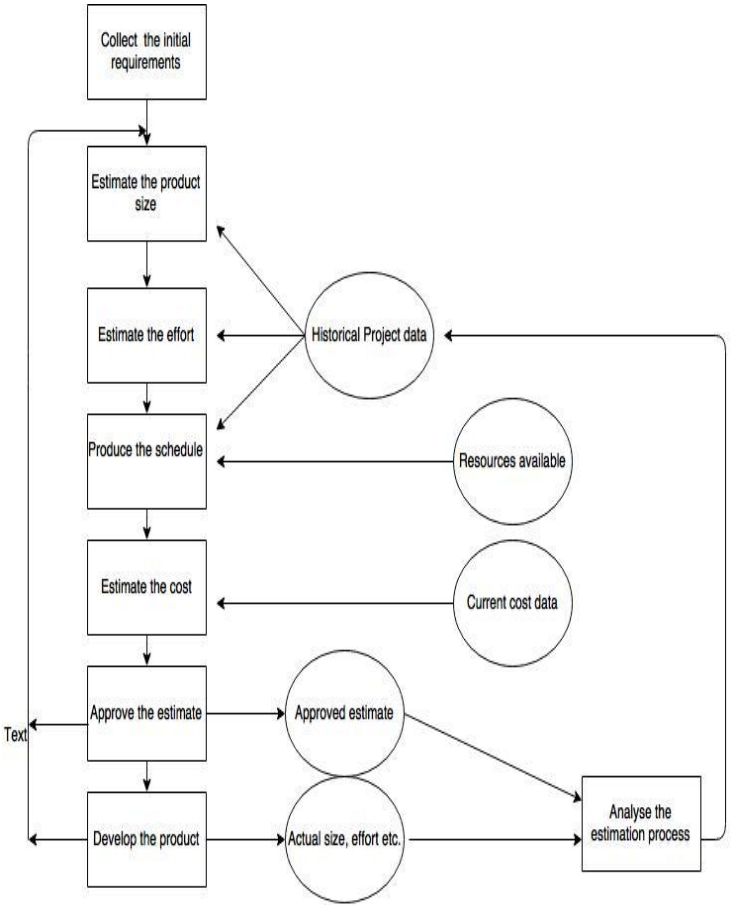

Figure .4 Relationship of Software estimation and Software Mining

IX. DESIGN METHODOLOGY OF PROPOSED SYSTEM

The proposed methodology will be implemented using the MATLAB tool. The research problem is a vital part of any research based activity. If the nature of the problem is clear then it becomes quite easy to solve the problem. Software engineering domain has been a wide domain of research work and investigation. The work is based upon mining the efforts of a software engineering data. The problem of the study is to look for the best algorithm that can effectively mine the efforts of the software based on the parameters defined. All the parameter that is employed for the purpose of evaluation is listed and they are compared against the outputs of the proposed methodology and existing methodology. In the past years, many techniques have been intended for calculation of the effort estimation which are model based techniques, neural networks based techniques, use case based techniques etc. The technique mentioned in [8] is a hybrid technique of Use case and neural networks for the effort estimation. The Random Forest (RF) classifier is applied which will classify the parameters of the software.

There is a need to develop an algorithm which does not require any algorithm-specific parameters and teaching-learning-based optimization (TLBO) is such an algorithm. The TLBO algorithm is a teaching-learning process based on the effect of influence of a teacher on the result of learners in a class. The algorithm has two vital modes of the learning:
(i) Through teacher (known as teacher phase)
(ii) Through interaction with the other learners (known as learner phase)

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 4, April 2018

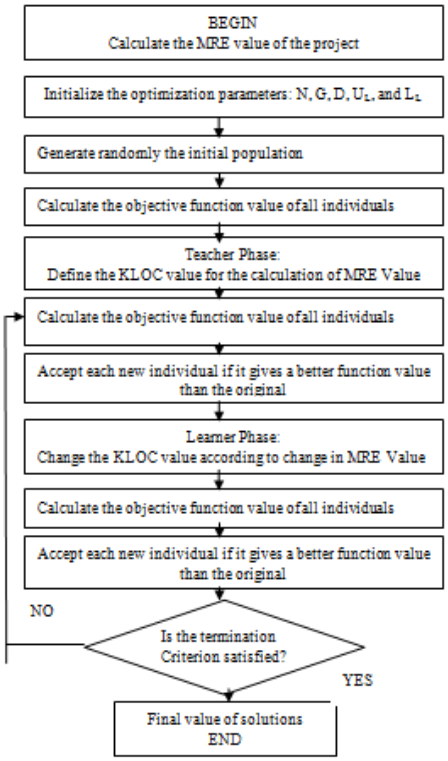

Figure 5. Flow chart of proposed methodology.

1) Initialization: The population X is arbitrarily initialized with a search space surrounded by matrix of N rows and D columns. N stands for the numeral of learners in a class i.e. "class size" or ''population size '' in this case. D is the amount of "subjects offered" to the learners, which is same as the dimensionality of the problem taken. The procedure being iterative is set to run for G maximum number of generations. The jth parameter of the ith learner (vector) in the primary generation is assigned values arbitrarily using the equation limits of design variables (upper, UL and lower, LL of each case)

$$x_{(i,j)}^{1} = x_j^{min} + \text{rand} * (x_j^{max} - x_j^{min}) \quad (1)$$

Where rand is used to represent a uniformly distributed over the range of (0, 1).

$$X_{(i)}^{g} = [\, x_{(i,1)}^{g}, x_{(i,2)}^{g}, \mathrm{K}, x_{(i,j)}^{g}, \mathrm{K}, x_{(i,D)}^{g} \,] \quad (2)$$

Where Eq. (2) shows the learning parameters of i^{th} learner of the iteration

2) Teacher Phase: The mean vector M^g that lists the mean value of the learners in the class for every subject at creation g is calculated. The mean vector M^g is calculated as following Eq. (3).

$$M^{g} = \begin{bmatrix} mean\left(\left[x_{(1,1)}^{g}, \mathrm{K}, x_{(i,1)}^{g}, \mathrm{K}, x_{(N,1)}^{g} \right] \right) \\ \mathrm{K} \\ mean\left(\left[x_{(1,j)}^{g}, \mathrm{K}, x_{(i,j)}^{g}, \mathrm{K}, x_{(N,j)}^{g} \right] \right) \\ \mathrm{K} \\ mean\left(\left[x_{(1,D)}^{g}, \mathrm{K}, x_{(i,D)}^{g}, \mathrm{K}, x_{(N,D)}^{g} \right] \right) \end{bmatrix}^{T} \quad (3)$$

This gives us the following equation:

$$M^{g} = [\, m_1^g, \; m_2^g, K \; m_j^g, K, \; m_D^g \,] \quad (4)$$

The given Eq. (4) gives the value of minimum objective function when the value is taken as the teacher.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 4, April 2018

$$X^{g}_{new(i)} = X^{g}_{(i)} + \text{rand} * (X^{g}_{teacher} - T_F^{Mg}) \quad (5)$$

The Eq. (5) gives the mean value of the learner towards the teachers.

3) Learner Phase: This phase is allied with the perception of the interaction of the learners with each other. The practice of mutual contact tends to amplify the information of the learner. Each learner tries to interrelate in an arbitrary fashion with other learners and hence helps in the process of knowledge sharing.

$$X_{new(i)}^{g} = \begin{cases} X_{(i)}^{g} + rand \times \left(X_{(i)}^{g} - X_{(r)}^{g}\right) & if\left(X_{(i)}^{g} < X_{(r)}^{g}\right) \\ X_{(i)}^{g} + rand \times \left(X_{(r)}^{g} - X_{(i)}^{g}\right) & otherwise \end{cases} \quad (7)$$

The Eq.(7) represents the value represented after the learner phase.The algorithm is terminated once the maximum number of iterations (G) are terninated.

X. RESULTS AND DISCUSSIONS

Results form the concrete proof for any research. After performing the research, following results have been generated.

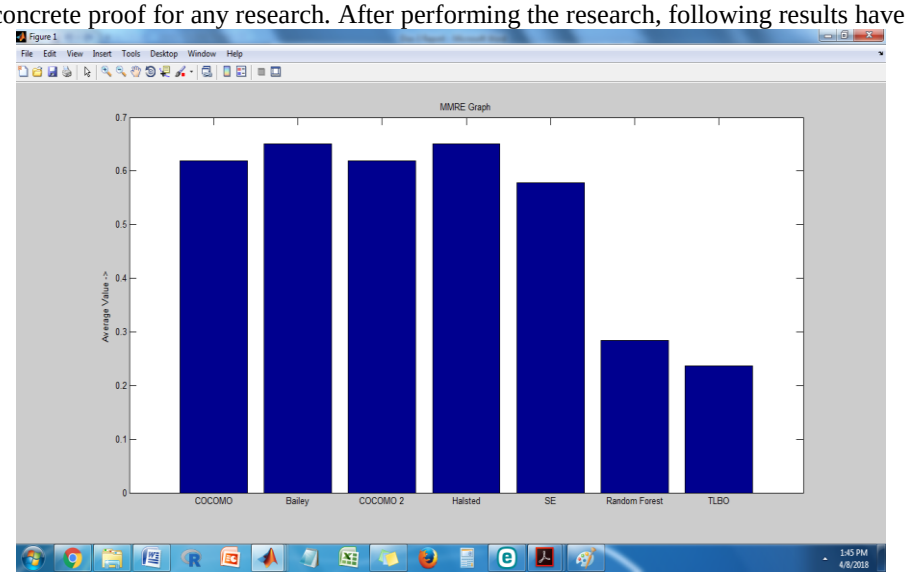

Figure 6 .Comparison of existing methods in the form of MMRE Graph

The output as well as the performance of the applied algorithms is measured in terms of different parameters like (MRE) Mean Relative Error and (MMRE) Minimum Mean Square Error and Execution Time. The parameters of the evaluation form the crisp proof of valid results.
The MRE is defined as below.

$$\text{MRE}_i = \frac{|Actual_i - Estimated_i|}{Actual_i}$$

The value of MRE is the measure to calculate the absolute percentage of error between actual and predicted effort for each given set of projects and i define the number of the projects in the iteration. Each value of MRE is used to calculate the goodness of fit.
The MMRE is used to find the average of MRE over all given set of projects.

$$\text{MMRE} = 1/n \sum_{i=1}^{n} MRE_i$$

Figure 6 shows a comparative analysis of the existing techniques and it shows that they have higher value of MMRE but the implication of TLBO has reduced the value of MMRE (as it is shown in the figure).

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 4, April 2018

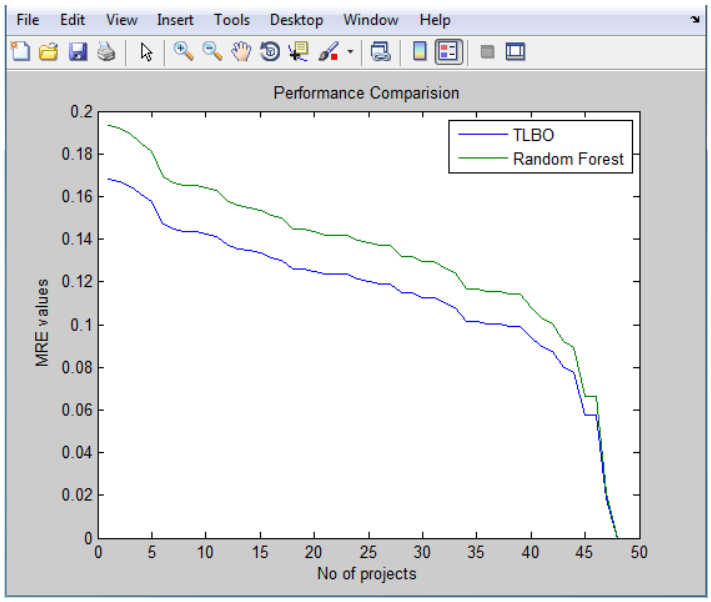

Figure 7. Performance analysis in terms of MRE values

As it is clear from the figure 7 that MRE value when using TLBO comes to around 0.167 which is an optimized value.

Parameter	Existing technique	Proposed Technique
MMRE	0.31	0.23
MRE	0.21	0.167
Execution Time	8.62 seconds	7.96 seconds

Table 1. Table showing the output in terms of the inputs specified as evaluation parameters.

XI. CONCLUSION AND FUTURE SCOPE

The contemporary role of software engineering experts is usually not discussed in papers on data mining for software engineering; making valuable acceptance of approaches developed in this field is rather complicated in practice. In order to fill in this gap, this research will try to proffer a discussion about the pivot role of software engineering experts when looking for the data mining approaches. As research is an ongoing process, the current work could be replicated with the number of different optimization issues that could be adaptive in nature. Next to with this, it could also integrate the application of test data in the study so as to authenticate results bearing in mind a proper validation practice. Other objectives such as maximization of probability of success of the portfolio, reduction of resource usages and the variation in it may be considered in future research as well. This can also further be expanded in terms hybrid and multiobjective TLBO which could be varied in terms of parameters and datasets.

ACKNOWLEGDEMENT

Foremost, I would like to express my sincere gratitude to my advisor Jaspreet Kaur Sahiwal for the continuous support of my research, for her patience, motivation, enthusiasm, and immense knowledge. Her guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my M.Tech study.

REFRENCES

[1] Q. Taylor and C. Giraud-Carrier, “Applications of data mining in software engineering”, Int. J. Data Analysis Techniques and Strategies,2010.
[2] Tao Xie, Suresh Thummalapenta, David Lo, Chao Liu. Data Mining for Software Engineering. In « IEEE Computer », 8, 42, August, 2009, 55-62.
[3] Thomas Zimmermann, A. Zeller, P. Weissgerber, S. Diehl. Mining version histories to guide software changes. In « IEEE Transactions on Software Engineering », 6, 31, June, 2005, 429–445.
[4] D. Schuler, T. Zimmermann. Mining usage expertise from version archives. In « MSR », 2008.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 4, April 2018

[5] Stefan Henss, Martin Monperrus, Mira Mezini. Semi-Automatically Extracting FAQs to Improve Accessibility of Software Development Knowledge. In « Proceedings of the International Conference on Software Engineering », 793 - 803, 2012,
[6] J. Han and M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, 2nd edition, 2006.
[7] R-Y. Chang, A. Podgurski, and J. Yang, "Finding What's Not There: A New Approach to Revealing Neglected Conditions in Software," Proc. 2007 Int'l Symp. Software Testing and Analysis (ISSTA 07), ACM Press, 2007, pp. 163-173.
[8] Satapathy, S. M., Acharya, B. P., & Rath, S. K. (2014). Class point approach for software effort estimation using stochastic gradient boosting technique. ACM SIGSOFT Software Engineering Notes, 39(3), 1-6.

BIOGRAPHIES

Gurtej Singh Ubhi, M.Tech Scholar, Department of Computer Science, Lovely Professional University, Phagwara, India.

Jaspreet Kaur Sahiwal, Department of Computer Science, Lovely Professional University, Phagwara, India.