*i*Journals: International Journal of Software & Hardware Research in Engineering (IJSHRE)
ISSN-2347-4890
Volume 10 Issue 3 March 2022

Literature Review of Software Engineering Fault prediction

Mustafa Zaki Mohammed[1]; Ibrahim Ahmed Saleh[2];

Department of Software, College of Computer and Mathematics, University of Mosul, Iraq[1,2]

E-mail: mustafa.20csp27@student.uomosul.edu.iq ; i.hadedi@uomosul.edu.iq[2]

ABSTRACT

During the software development life cycle, the testing phase is an important stage to verify the efficiency of the resulting product. With the increase project size and complexity, software testing becomes a time consuming and costly task and to address this issue we use Artificial Intelligence Techniques (AIT) which is a fruitful approach in the modern trend of delivering high quality software. It is effectively implemented to improve the outcomes of all phases of the Software Development Lifecycle (SDLC). This paper focuses more on artificial intelligence techniques for error handling to overcome the cost and time of testing and improve software quality. Error handling is done by detecting and predicting errors. Forecasting software errors is a fundamental activity in software development. This is because predicting and detecting errors before deploying the software achieves user satisfaction and improves the overall performance of the software. Moreover, anticipating software error early improves software adaptability to different environments and increases resource usage.

.

Keywords: Artificial Intelligence, Software Engineering, Test case optimization, Fault Detection, Error Prediction

1. INTRODUCTION

A software defect is an error, flaw, failure, or fault in a computer program or system that causes it to produce an incorrect or unexpected result, or to behave in unintended ways. Most defects arise from mistakes and errors made by people in either a program's source code or its design, or in frameworks and operating systems used by such programs [1].

Software mistake has a significant impact on software reliability, quality, and maintenance costs which show the fault tree analysis illustrate in figure (1). Error-free software is difficult to achieve, even when the software is used correctly, because most errors are concealed. Furthermore, establishing a software error prediction model that can anticipate errors early on is a hard mission in software engineering [2].

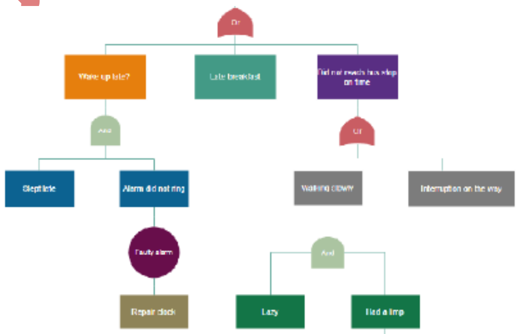

Fig 1: fault tree analysis

The application of computational intelligence techniques to solve software engineering problems is part of a relatively new field [3]. Due to the large test case, test case optimization has become a complex problem. Therefore, an important method of error handling is the prediction of software errors. This will predict the error in the early stage and sometimes before the project starts. This method gives time for the software engineer and lab to plan their stages early. Since the handling of error in later phase is exponentially high. Due to increase the agility in software delivery everyone is facing the shortage of time. The prediction technique will play a key role in guiding the error detection methods by indicating which parts require more testing. As a result, testing costs and time are reduced, and software quality is improved [3]. Many software defect prediction datasets, methods and frameworks are published disparate and complex, thus a comprehensive picture of the current state of defect prediction research that exists is missing. This literature review aims to identify and analyze the research trends, datasets, methods and frameworks used in software defect prediction research between 2000 and 2021.

The reset of paper section 2 is software quality, while section 3 is the error in software, section 4 Literature survey and finally conclusion.

2. SOFTWARE QUALITY

The primary procedures are Software Quality Assurance (SQA) and verification and validation (V&V) shown in figure (2), as they directly affect the quality of the software output. However, the word "product" will be expanded to include any artifact that is produced as a result of any process used to create the final software product. A full system specification, a software requirements specification for a software component of a system, a design module, code, test documentation, or outputs from quality analysis activities are all examples of products [4].

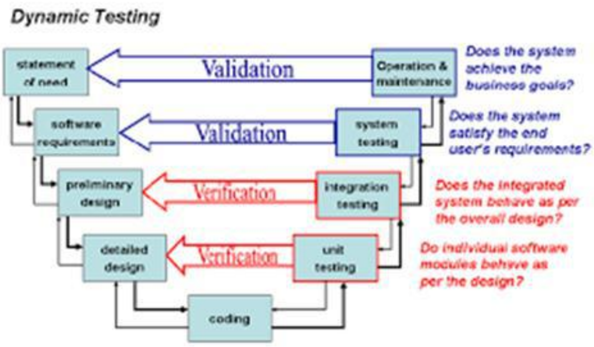
Dynamic Testing
statement of need
Validation
Operation & maintenance
Does the system achieve the business goals?
software requirements
Validation
system testing
Does the system satisfy the end user's requirements?
preliminary design
Verification
integration testing
Does the integrated system behave as per the overall design?
detailed design
Verification
unit testing
Do individual software modules behave as per the design?
coding

Fig 2: Verification and Validation V&V

The majority of quality treatments are expressed in terms of the end system's performance. The rationale for this expansion of the term "product" is because SQA and V&V may be used to assess intermediate and final goods. It may be applied to user documentation as well as intermediate products and code. User documentation is best written in tandem with code and can frequently force difficulties with requirements and code. Depending on the project organization, some testing may come within the development, SQA, or V&V processes. Testing is part of the V&V discipline, which necessitates testing operations from the start of the project. Because testing is addressed in both the SQA and V&V plans, Test management, planning, and documentation provide supporting strategies for testing. Component or module testing, integration testing, system testing, and acceptance testing are all examples of V&V testing V&V testing may involve examination of tools to be used in the project as well as testing of commercial off-the-shelf software (COTS) [4].

3. SOFTWARE PREDICTION FAULT

A software fault is a flaw in a software product that fails to fulfill a software requirement or end user expectation. To put it another way, a defect is a coding or logic fault that causes a program to malfunction or provide unexpected and wrong outcomes, type of fault which is facing most of software are shown in figure (3). The method of detecting problematic modules in software is known as software defect prediction. The provided end result should have as few flaws as possible in order to produce high-quality software. Early discovery of software flaws may result in lower development costs, less time and rework, and more dependable software. As a result, defect prediction is critical for high-quality software. In order to create a statistical defect prediction model, software defect prediction metrics are critical. Software companies can utilize defect prediction models to detect problematic modules in the early stages of development [5].

4. LITERATURE SURVEY

This section focuses on the fault prediction studies conducted by several writers. The authors selected many criteria for the survey including the metric set used, the methodology used, and the dataset used for fault prediction analysis. According to the findings of the fault prediction study, a significant number of academics and practitioners have used the Chidamber and Kemerer metrics suite for fault prediction.

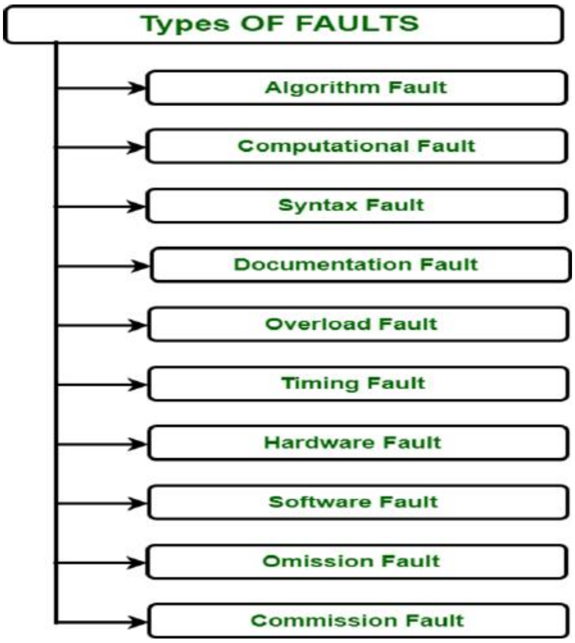

Fig 3: types of software fault

Rosli et al. 2011 [6] introduced a fault proneness prediction application based on the genetic algorithm. The application gets its values from an open-source software project, such as object-oriented metrics and count metrics values. The evolutionary algorithm takes the application's data as inputs to develop rules that are then used to classify software modules as defective or non-defective. Finally, use the genetic algorithm applet to visualize the results. At the same year, Catal et al. [7] suggest logistic regression (LR) as a predictor of software faults in seventeen investigations with various data sets. Nave Bayes is another extensively used predictor in software error detection (note). Eleven studies from various nations employed this algorithm in their software disciplines. It was C4.5, and its Java application was J48, which was another frequently used defect predictor. Use something similar to this. There were thirteen algorithms in total. Random forests (RF) are another indication mentioned frequently in Catal et al.'s study RF has been utilized in five studies as a group indicator.

Partha Sarathi Bishnu and Vandana Bhattacherjee.2012 [8] For software defect prediction, use quad tress-based k-means clustering. To demonstrate that clusters generated using the above approach have the highest gain values. To assess the total error rate and compare it to that of other approaches (NB, CS, CT, DA).

AhmetOkutan, O.T.Y. 2014[9], Introduced the most successful measures for defect prediction, such as Response for class (ROC), Line of code (LOC), and Lack of Coding Quality (LOCQ)Using Bayesian network classifier experimental. At the same year, Ezgi Erturk et al. [10] proposed a new method Adaptive Neuron Fuzzy Inference System (ANFIS) for the software fault prediction. Then for performing experiment they used PROMISE Software Engineering Repository dataset, and McCabe metrics are selected because they comprehensively address the programming effort. The results achieved were 0.7795, 0.8685, and 0.8573 for the SVM, ANN and ANFIS methods, respectively.

R. Malhotra 2015[1] presented a thorough study of software bug prediction strategies utilizing Machine Learning in. (ML). The research reviewed all works published between 1991 and 2013, studied and evaluated machine learning techniques for software bug prediction models, compared ML and statistic techniques, compared different ML techniques, and described the strengths and weaknesses of ML techniques. At the same year, While M. C. Prasad, L. Florence and A. Arya, [11] covered a variety of machine learning approaches as well as ML skills in software fault prediction. The research aided the developer in utilizing valuable software measurements and appropriate data mining techniques to improve software quality. where they used Classification Techniques such as Supervised, Un-supervised and Semi-supervised. and Laradji et al. [12] introduce in their study the effects of combination feature selection and performance aggregation learning algorithms defect rating. In this study, the authors suggest a new algorithm for group learning with two variables in order to increase the effectiveness of feature selection and prevent Negative effects of data imbalance.

Abdo and Darwish, 2018 [13] used the reshaping technique with three different types of group learners and eight grammar learners in their study. Learners were examined on seven different types of standard data sets provided by PROMISE Repository. The accuracy of most of the algorithms employed in the experiments shows that group learners use more than individual learners. At the same year, Ardimento et al. [14] proposed a multi-source machine learning strategy that relies on product and process metrics to predict defects. It

Researcher	Year	Subject
Rosli et al	2011	Introduced a fault proneness prediction application based on the genetic algorithm. The application gets its values from an open-source software project, such as object-oriented metrics and count metrics values.
Catal et al	2011	Used logistic regression (LR), Nave Bayes, Random forests (RF), as a predictor of software faults with various data sets.
Partha Sarathi Bishnu and Vandana Bhattacherjee	2012	use quad tress-based k-means clustering, For software defect prediction. to assess the total error rate and compare it to that of other approaches (NB, CS, CT, DA).
Ahmet Okutan, Olcay Taner Yıldız	2014	Introduced The most successful measures for defect prediction, such as Response for class (ROC), Line of code (LOC), and Lack of Coding Quality (LOCQ)
Ezgi Erturk et al	2014	introduced a novel approach for software defect prediction called Adaptive Neuron Fuzzy Inference System (ANFIS)
R. Malhotra	2015	Using presented a thorough study of software bug prediction strategies utilizing Machine Learning in (ML)
M. C. Prasad et al	2015	Using Data Mining and Machine Learning Techniques to Predict Software Defects Based on Software Metrics.
Ez Laradji et al	2015	Suggest a new algorithm for group learning with two variables in order to increase the effectiveness of feature selection and prevent Negative effects of data imbalance
Abdo and Darwish	2018	used the reshaping technique with three different types of group learners and eight grammar learners in their study. Learners were examined on seven different types of standard data sets provided by PROMISE Repository.
Ardimento et al	2018	proposed a multi-source machine learning strategy that relies on product and process metrics to predict defects
Kaur and Kaur	2018	used Software metrics such as Chidamber & Kemerer (C&K), Henderson & Sellers, and McCabe were used to estimate software problems. Demonstrate the use of random forest algorithms and effective packaging Software error prediction findings based on these metrics
Abdullah Bin Ateya et al	2020	smart unit tests are created to identify potential errors in the source code of the method under test. An Abstract Syntax Tree (AST) was prepared and constructed for the method source code

was put to the test on an actual dataset consisting of 183 releases from two well-known open-source software systems. The results of this investigation revealed that the proposed method for defect prediction was successful. and Kaur and Kaur,[15] used Software metrics such as Chidamber & Kemerer (C&K), Henderson & Sellers, and McCabe were used to estimate software problems. Demonstrate the use of random forest algorithms and effective packaging Software error prediction findings based on these metrics.

Abdullah Bin Ateya et al.2020 [16] smart unit tests are created to identify potential errors in the source code of the method under test. An Abstract Syntax Tree (AST) was prepared and constructed for the method source code. After the abstract syntax tree is generated, the syntax tree is analyzed to find a set of constraints and values that help predict software errors. These constraints usually come in the form of mathematical and logical expressions. The constraints are then resolved to evaluate the data values that lead to exceptions and cause errors at runtime. table (1) illustrate the summation of all lecturer reviewers.

5. CONCLUSION

In this paper introduce a literal of software engineering fault prediction. The major objective of this review is to characterize different techniques used predication of software fault. The literal review is used to helps the software designers to improve the software quality more and more, to

reduce the directing test effort, reducing cost of budget, to project become more reliability beside the quality of the software. In conclusion, a large measure of techniques has been designed for effective predicting or removing faults from software. Predication fault techniques are reasonably new to the range of fault, as well as there is unique technique for each application, also it must appropriate to the cost of fault predication

the customers want. Finally, there are not enable to design software project without faults, the software fault prediction will be important processes in the software system.

6. REFERENCES

[1]. Malhotra, Ruchika. "A systematic review of machine learning techniques for software fault

[2]. prediction." *Applied Soft Computing 27 (2015): 504-518.*

[3]. Hammouri, A., Hammad, M., Alnabhan, M., & Alsarayrah, F. (2018). Software Bug Prediction using Machine Learning Approach. In *IJACSA) International Journal of Advanced Computer Science and Applications* (Vol. 9, Issue 2). www.ijacsa.thesai.org.

[4]. Kumar, S., & Ranjan, P. (2017). A Comprehensive Analysis for Software Fault Detection and Prediction using Computational Intelligence Techniques. In *International Journal of Computational Intelligence Research* (Vol. 13, Issue 1). http://www.ripublication.com

[5]. P. Shen, X. Ding, W. Ren and C. Yang, "Research on Software Quality Assurance Based on Software Quality Standards and Technology Management," *2018 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2018, pp. 385-390,*

[6]. 1Pooja Paramshetti, & 2D. A. Phalke. (2015). *Software Defect Prediction for Quality Improvement Using Hybrid Approach*. (Vol.4, Issue 6).

[7]. M. M. Rosli, N. H. I. Teo, N. S. M. Yusop and N. S. Moham, "The Design of a Software Fault Prone Application Using Evolutionary Algorithm," *IEEE Conference on Open Systems, 2011.*

[8]. C. Catal, Software fault prediction: a literature review and current trends, *Expert Syst. Appl. 38 (4) (2011) 4626–4636.*

[9]. Software Fault Prediction Using Quad Tree-Based K-Means Clustering Algorithm. Bishnu, Partha S and Bhattacherjee, Vandana. 2012, IEEE Transactions on Knowledge and Data Engineering, VOL. 24, NO. 6, JUNE

[10]. Okutan, Ahmet, and Olcay Taner Yıldız. "Software defect prediction using Bayesian networks." *Empirical Software Engineering 19.1 (2014): 154-181.*

[11]. E. Erturk and E. A. Sezer, “A comparison of some soft computing methods for software fault prediction,” *Expert Systems with Applications, 2014.*

[12]. M. C. Prasad, L. Florence and A. Arya, "A Study on Software Metrics based Software Defect Prediction using Data Mining and Machine Learning Techniques," *International Journal of Database Theory and Application, pp. 179-190, 2015.*

[13]. I.H. Laradji, M. Alshayeb, L. Ghouti, Software defect prediction using ensemble learning on selected features, Inf. Softw. *Technol. 58 (2015) 388–402.*

[14]. A.S. Abdou, N.R. Darwish, Early prediction of software defect using ensemble

learning: a comparative study, *Int. J. Computer Appl. 179 (46) (2018) 29–40.*

[15]. P. Ardimento, M. Luca Bernardi, M. Cimitile, A multi-source machine learning approach to predict defect prone components, in: *Proceedings of the 13th International Conference on Software Technologies, 2018, pp. 272–279*

[16]. A. Kaur, I. Kaur, An empirical evaluation of classification algorithms for fault prediction in open-source projects, *J. King Saud University – Computer Inform. Sci 30 (1) (2018) 2–17.*

[17]. Abdullah Bin Ateya, H., Mohammed Saeed Banaeamoon, S., Author, C., & Bin-Ateya, H. (2020). Software Bug Prediction using Tree-Based Approach. In *International Journal of Engineering and Artificial Intelligence* (Vol. 1, Issue 2). http://www.ijeai.com