Journal of Positive School Psychology http://journalppw.com
2022, Vol. 6, No. 5, 5474–5481

Statistical Review of Dataset and Mathematical Model for Software Reliability Prediction Using Linear Regression

Mayuri H. Molawade[1], Dr. Shashank.D. Joshi[2], Dr. Rohini Jadhav[3]

[1]*Deparment of computer Engineering, BV(DU)COE, Research Scholar, Pune, India*
[2]*Deparment of computer Engineering, BV(DU)COE, Professor, Pune, India*
[3]*Deparment of computer Engineering, BV(DU)COE, Assisatant Professor, Pune, India*

Abstract

Depending upon early stage statistics if any software developer knows what will be the software reliability that will be very helpful for any developer to make any changes at early stages, so by thinking about this we are defining advance software reliability prediction model. As we are considering this prediction at early stage so that we need to use some of parameters and some of the constant values to predict accurate result as possible as. Here in this paper we have introduce some of mathematical expression to calculate reliability of each parameter to check efficiency of parameter we used ready dataset and also perform analysis using linear regression for machine learning.

Keywords— reliability index, prediction phase, Reinforcement phase, depth of inheritance tree, weighted methods per class, coupling between objects, line of code.

INTRODUCTION

This reliability model has parameters, the values of which are used to predict the software's reliability. Since the initial failure intensity and cumulative number of errors, as well as their values, are unknown, the model cannot be used to predict software reliability.

It would be very useful if the values of these variables were widely common to all types of software and could be conveniently determined based on any software attribute.

However, there is currently no accurate easy method available.

Both software reliability models currently use the approach of calculating the value of these parameters for the individual software being modeled using accurate data for that software. To put it another way, the value of these parameters is calculated using the programmed being model’s accurate results.

As a result of this fact, for reliability modeling, the performance of the software system is carefully monitored during system testing, and data on reliability is analyzed throughout testing and captured up to timeT.

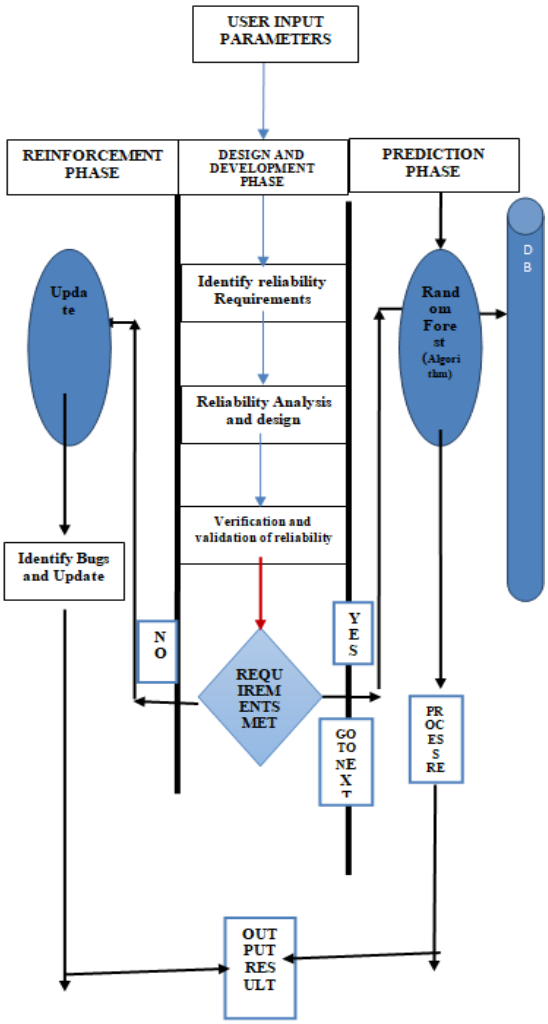
The value of these parameters is then determined using statistical techniques applied to the collected data. If the parameters' values are known, the software's reliability (in terms of reliability intensity) can be estimated. This mathematical techniques necessitate a significant volume of data; without it, the values of the variables are unknown.

SOFTWARE RELIABILITY PREDICTION

The architectural lifecycle can be broken down into three phases: design and development, reinforcement, and prediction.

Project proposals for components and structures are decided during the design and construction process to meet design specifications (from different aspects). Verification and evaluation was conducted towards the conclusion of the design and implementation process to ensure that the design solutions satisfy the specifications. Various stability-related engineering tasks, also known as reliability activities, are carried out during the lifecycle to ensure software's final reliability standard. Each reliability operation block in architecture represents a set of methods for ensuring reliability Identifying the reliability criteria is the first step in the design and construction process. Then, using reliability analysis and design methods, design strategies that meet the reliability criteria are determined. Verification and evaluation follow the selection of a design solution, with checks and design evaluations used to ensure that the design solution satisfies the reliability criteria. If the design solution's durability has been established, the prediction process will begin. Quality assurance methods are used in this step to ensure that no errors are incorporated into the prediction process, ensuring that the inherency is preserved. Finally, prediction phase, fault diagnosis and prognosis, and maintenance are needed to ensure the operational reliability of software.

Flow chart



DESIGN AND DEVELOPMENT INCLUDES

The first and most critical task in the software reliability prediction reliability engineering phase is to identify the reliability specifications. The reliability criteria are often expressed in terms of quantitative reliability indices in reliability engineering. The impact of reliability on various system attributes, such as availability, reliability (in a narrow sense), maintainability, and safety, can be measured using various reliability indexes. It is important to decide the goals for each class in order to define the reliability criteria, as seen below.

Safety Index

Loss Probability
Even Rate
Record rate

Availability index

Dispatch reliability or delay rate
Schedule Reliability
Transit Time
Turnaround time

Reliability index

Mean hour between failures
Mean time between failure and failure rate
Mean time between unscheduled removals

PREDICTION PHASE

• Testing is costly, but it is essential to produce high-quality applications. However, training should be stopped as soon as possible to avoid over-testing the device.

• Until the desired degree of device stability is reached, testing must come to an end.

• This desired level of reliability will be tested with a reliability prediction algorithm; if the model assumes that the required level of reliability can never be reached, the manager must consider whether or not to rewrite the programme components.

REINFORCEMENT PHASE

The term "reliability" refers to the likelihood of software. It is a failure-free process over a set period of time. It has the qualities of a complex system. The word dynamic is described in this context as a function of the number of programme failures. Getting rid of software flaws in those programmes When parts are used infrequently, they have no impact on reliability. It is important to decide their goal for each class as seen below in order to define the reliability criteria bug, and then to develop it using the knowledge engineering method.

Maintainability index

Mean time to repair or update

Here we use the aspect of reliability growth of a software which is defined below as reliability growth model

RELIABILITY MODEL

During the testing process, this model illustrates how device stability varies over time.

• As failures are detected when checking the system, the errors that caused them are corrected to increase programme stability during the testing and debugging phase.

•The computational reliability model is turned into a statistical model to predict reliability.

RANDOM FOREST ALGORITHM:

The random forest algorithm is a type of regression machine learning algorithm. The random forest algorithm is a coupling model made up ofb-regressive trees, classified as h(x,k),k=1, 2,...,b. The prediction of the random forest regressive model could be achieved by estimating the average ofb-regressive trees. After the MCMC generates the samples, the random forest model is built using these samples. The following is a summary of the random forest algorithm for reliability analysis setup procedure.

Step (1): Using the Bootstrap technique, create regression trees from MCMC samples by repeatedly abstracting b training sample classes. For b samples that are not chosen at each time of abstraction for training samples, an out-of-bag (OOB) group is formed as a study group to be examined.

Step (2): Make your regressive forest. Select randomlymtry (mtryk) variables as candidate branch variables from k variables among sub-branch points of each tree, then estimate their optimal branch variables using the branch-weight criterion.

Step (3): Each regressive tree branch continues to rise from top to bottom, recursively. Each tree's coefficient n tree-value is used as a regressive growth termination criterion.

Step (4): The regressive trees that are generated form a regressive random forest model. The OOB prediction accuracy, which is determined by the variance average of the examination collection of original data, can also be used to determine the prediction accuracy of the current model. Assuming that the sample size of OOB is bem,

MSEOOB=m∑i=1(yi−ˆyi)2m

R2RF=1−MSEOOBˆσ2y

Reliability prediction model,

a = fault consider at early stage

k = one constant

b1 =bug rate for previous training.

m (t) = reliability predicted for each parameter,

$$m1(t) = a/(1 + k.e^{-b1.t})$$

Depth of Inheritance Tree, The maximum distance in the inheritance hierarchy from the root to a given class. The maximum length inheritance path from a class to the root class is known as DIT.

$$m2(t) = a/(1 + k.e^{-b1.dit})$$

Weighted Methods per Class, WMC is classified as the total complexity of the class's methods. If all methods have the same level of complexity, it equals the number of methods. The number of each method's normalised complexity in a given class.

$$m3(t) = a/(1 + k.e^{-b1.wmc})$$

Coupling between Objects, The number of classes associated with a given class is represented by the CBO metric. Method calls, field accesses, inheritance, method arguments, return classes, and exceptions are all examples of couplings.

$$m4(t) = a/(1 + k.e^{-b1.cbo})$$

Line of Code, The LOC metric, which is built on Java binary code, sums the number of fields, procedures, and instructions in each method of the investigated class.

$$m5(t) = a/(1 + k.e^{-b1.loc})$$

By considering each and individual parameter reliability point at end will take average reliability to check for product reliability.

M(t)= Each parameter Reliability / number of parameter

To analysis above model we have done some of research , we have taken some values from one of research paper "DHARMENDRA LAL GUPTA1,2,* and KAVITA SAXENA2 Software bug prediction using object-oriented metrics Sa¯dhana¯ Vol. 42, No. 5, May 2017, pp. 655–669 , Indian Academy of Sciences" to check how our model behave on this values. Following are graphs of value analysis.

Depth of Inheritance Tree

Table 1. cases of DIT

	Input dit				output
cases	**a**	**K**	**b1**	**dit**	**m(t)**
case 1	1.01	0.5	0.043	1	0.682913
case2	2	0.4	0.043	3	1.479734
case3	2.07	0.9	0.043	1	1.111634
case 4	6.65	6	0.043	1	0.985551
case 5	1.532	0.01	0.043	3	1.518651
case 6	1.21	0.1	0.043	1	1.104225
case 7	0.9	0.1	0.043	1	0.821324
case 8	1.1	0.1	0.043	2	1.007547
case 9	1.33	0.1	0.043	3	1.22254
case 10	1.98	0.1	0.043	1	1.806913

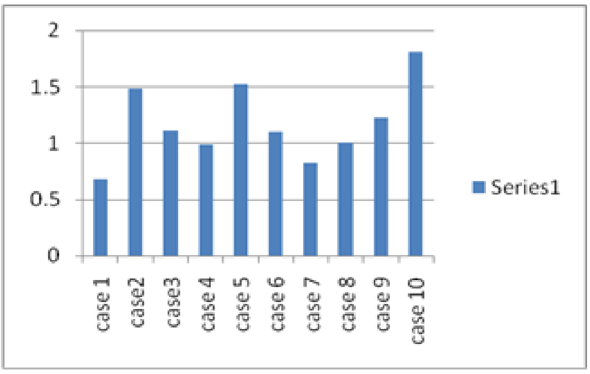


Fig 2. Graph for DIT

Weighted Methods per Class

Table 2. cases of WMC

	Input wmc				Output
Cases	**a**	**k**	**b1**	**wmc**	**M(t)**
Case 1	0.714	0.1	0.223	4	0.685887
Case2	1.532	2	0.223	5	0.925173
Case3	0.85	0.11	0.223	5	0.820404
Case 4	0.98	3	0.223	40	0.979607
Case 5	1.532	2	0.223	5	0.925173
Case 6	1.21	1	0.223	15	1.168774
Case 7	0.9	0.1	0.223	20	0.89896
Case 8	1.1	0.12	0.223	1	1.003635
Case 9	1.01	0.1	0.223	6	0.984175
Case 10	0.98	0.1	0.223	3	0.932245

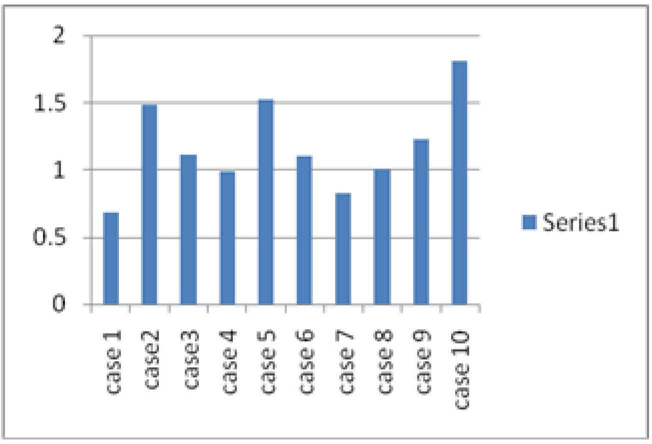


Fig 3. Graph for WMC

Coupling between Objects

Table 3. cases of CBO

	Input cbo				Output
cases	**a**	**K**	**b1**	**CBO**	**m(t)**
case 1	0.714	0.1	0.157	2	0.66539
case2	1.532	2	0.157	4	0.741033
case3	0.85	0.11	0.157	6	0.815045
case 4	0.98	3	0.157	2	0.307053
case 5	1.532	2	0.157	5	0.801124
case 6	1.21	1	0.157	8	0.94176
case 7	0.9	0.1	0.157	4	0.854401
case 8	1.1	0.12	0.157	2	1.011341
case 9	1.01	0.1	0.157	3	0.950641
case 10	0.98	0.1	0.157	1	0.902833

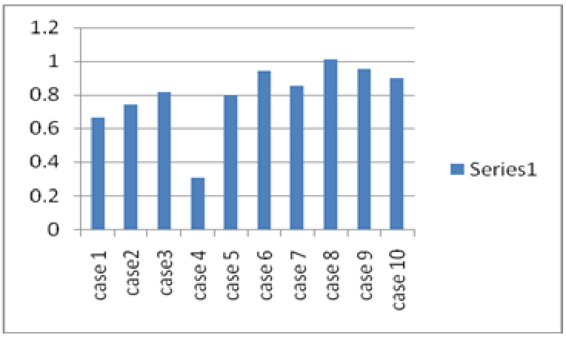


Fig 4. Graph for CBO

1. Line of Code

Table 4. cases of LOC

	Input loc				output
cases	**a**	**K**	**b1**	**LOC**	**m(t)**
case 1	0.714	0.1	0.232	624	0.714
case2	1.532	2	0.232	1020	1.532
case3	0.85	0.11	0.232	560	0.85
case 4	0.98	3	0.232	450	0.98
case 5	1.532	2	0.232	500	1.532
case 6	1.21	1	0.232	820	1.21
case 7	0.9	0.1	0.232	900	0.9
case 8	1.1	0.12	0.232	1073	1.1
case 9	1.01	0.1	0.232	400	1.01
case 10	1.11	0.1	0.232	200	1.11

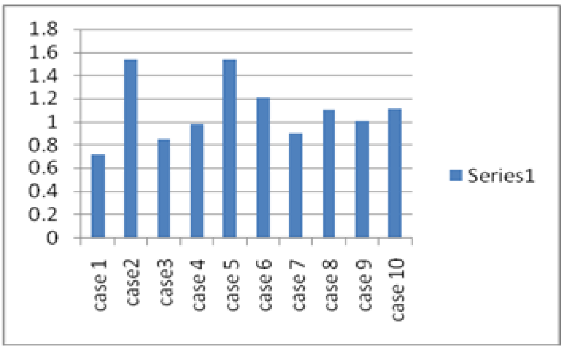


Fig 5. Graph for LOC

All reliability in one graph

Table 5. analysis of parameters

cases	m(t)dit	m(t)wmc	m(t)cbo	m(t)loc
case 1	0.899	0.685887	0.66539	0.714
case2	0.725	0.925173	0.741033	1.532
case3	1.112	0.820404	0.815045	0.85
case 4	0.986	0.979607	0.307053	0.98
case 5	1.519	0.925173	0.801124	1.532
case 6	1.104	1.168774	0.94176	1.21
case 7	0.821	0.89896	0.854401	0.9
case 8	1.008	1.003635	1.011341	1.1
case 9	0.928	0.984175	0.950641	1.01
case 10	0.894	0.932245	0.902833	0.98

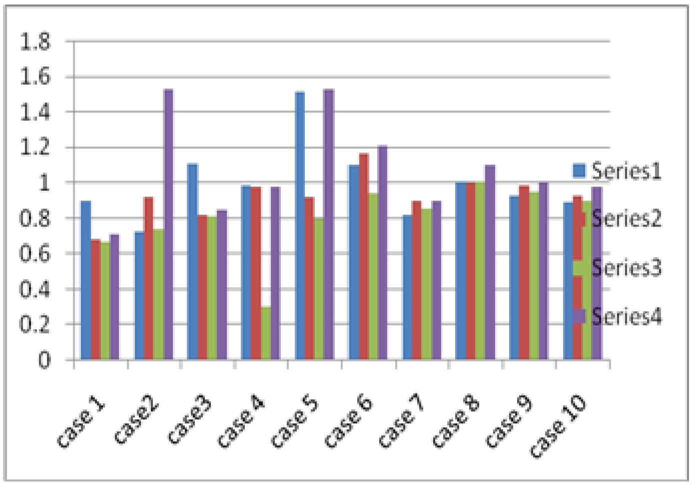


Fig 6. Graph for analysis

After analysis we have find out some of values give very good result in case parametric some very few of them give result very poor so we need to improve in that area. Also we have done linear regression to check that how many of them are lies as outlier and how many of them are present on slop.

Linear regression using above table values in statistics, **linear regression** is a linear approach to modeling the relationship between a scalar response and one or more explanatory variables The case of one explanatory variable is called simple linear regression.

So, here we have consider all reliability prediction values for Depth of Inheritance Tree, Weighted Methods per Class

Coupling between Objects, Line of Code parameters then we have calculated using linear regression for machine learning.

In this x is multiple cases and y is predicted values calculated as

$y = B0 + B1*x$

Here, B0 and B1 are consider as coefficients.

By considering above explanation we have plotted linear regression line graph for each parameter.

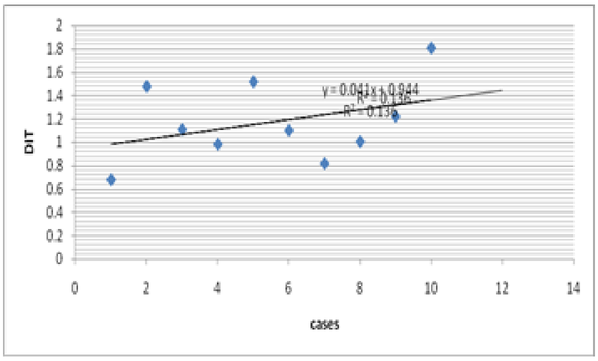


Fig7. Linear regression for DIT

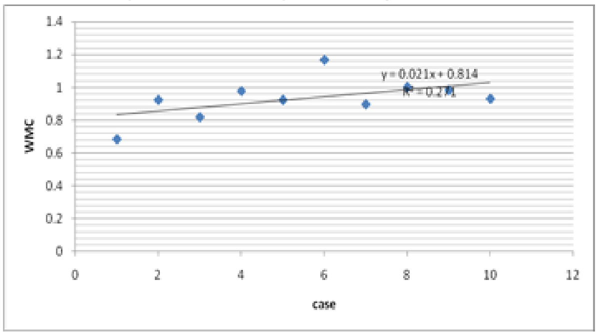


Fig8. Linear regression for WMC

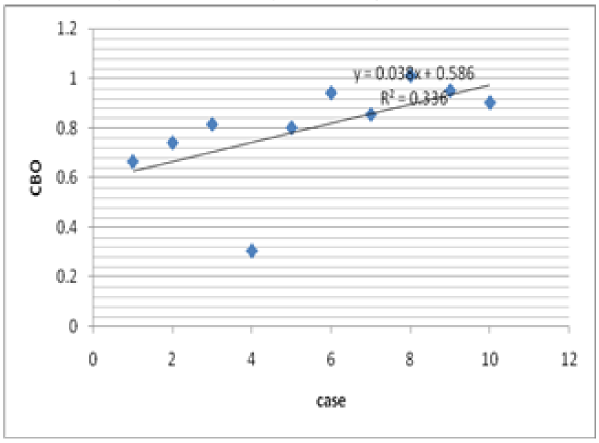


Fig 9. Linear regression for CBO

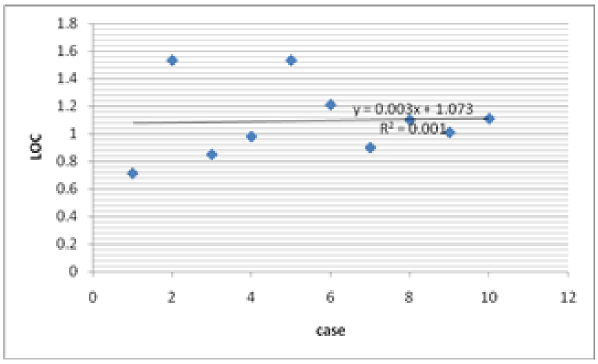


Fig 10. Linear regression for LOC

So here we can say that some of values are showing tremendous results but we need to develop this system for our own dataset this is just analysis of equation to check behavior of system.

For this stage of research we completed testing of our initial mathematical expression next we have to work on time variant as how these expression will work on time variant as well as other than these parameter how it will work.

References

1. Hossein A khavan “Power systems big data analytics: An assessment of paradigm shift barriers and prospects” Energy Reports 4 (2018) 91–100 30 November 2017
2. Javier Alonso “Predicting Software Anomalies using Machine Learning Techniques”
3. Ankur Choudhary “Software Reliability Prediction Modeling: A Comparison of Parametric and Non-Parametric Modeling”
4. Ashima Gupta “Prediction Of Software Anomalies Using Time Series Analysis – A Recent Study” International Journal of Advances In Computer Science and Cloud Computing, ISSN: 2321-4058 Volume- 1, Issue- 1, May-2013
5. Sultan H. Aljahdali “Software Reliability Prediction Using Multi-Objective Genetic Algorithm”
6. Soumya Joseph “Software Defect Prediction Using Enhanced Machine Learning Technique” International Journal of Innovative Research in Computer and Communication Engineering (*An* ISO 3297: 2007 Certified Organization)Vol. 4, Issue 6, June 2016
7. Philip Gross “Predicting Electricity Distribution Feeder Failures Using Machine Learning Susceptibility Analysis”
8. ArunimaJaiswal “Software reliability prediction using machine learning techniques” Int J SystAssurEngManag (February 2018) 9(1):230–244 https://doi.org/10.1007/s13198-016-0543y

9. RamakantaMohanty "Application of Machine learning techniques to Predict software reliability" 70 International Journal of Applied Evolutionary Computation, 1(3), 70-86, July-September 2010
10. Rita G. Al gargoor "Software Reliability Prediction Using Artificial Techniques**"** IJCSI International Journal of Computer Science Issues, Vol. 10, Issue 4, No 2, July 2013 ISSN (Print): 1694-0814 | ISSN (Online): 1694-0784 www.IJCSI.org
11. G.Krishna Mohan "Assessment and Analysis of Software Reliability Using Machine Learning Techniques" International Journal of Engineering & Technology, 7 (2.32) (2018) 201-205 International Journal of Engineering & Technology Website: www.sciencepubco.com/index.php/IJET
12. G. Martínez-Arellano "In-process Tool Wear Prediction System Based on Machine Learning Techniques and Force Analysis" Peer-review under responsibility of the International Scientific Committee of the 8th CIRP Conference on High Performance Cutting (HPC 2018)..
13. D. HemaLatha"A Development Model for Predicting Software Reliability Using Ant Colony Optimization Technique for Change Oriented Software Process" IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 19, Issue 2, Ver. I (Mar.-Apr. 2017), PP 57-64 www.iosrjournals.org
14. Vipul Vashisht "A Framework for Software Defect Prediction Using Neural Networks" Journal of Software Engineering and Applications, 2015, 8, 384-394Published Online August 2015 in SciRes. http://www.scirp.org/journal/jseahttp://dx.doi.org/10.4236/jsea.2015.88038
15. V**. Sangeetha"**A Survival Study on Software Failure Prediction Using Adaptive Dimensional Gene Model**"** International Journal of Computational Intelligence and Informatics, Vol. 6: No. 4, March 2017
16. Guru Prasad PANDIAN "A critique of reliability prediction techniques for avionics applications**"** Chinese Journal of AeronauticsReceived Date: 20 February 2017
17. AwniHammouri"Software Bug Prediction using Machine Learning Approach" (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 9, No. 2, 2018
18. TeeratPitakrata "Hora: Architecture-aware Online Failure Prediction" *The Journal of Systems & Software* 24 February 2017
19. Ajay Chandramouly "Reducing Client Incidents through Big Data Predictive Analytics"IT@Intel White Paper Intel IT IT Best Practices Big Data Predictive Analytics December 2013
20. Yoshinobu Tamura "Software Reliability Model Selection Based on Deep Learning with Application to the Optimal Release Problem"Journal of Industrial Engineering and Management Science, Vol. 1, 43–58.doi: 10.13052/jiems2446-1822.2016.003_c 2016 River Publishers. All rights reserved.
21. LanGuo "Robust Prediction of Fault-Proneness by Random Forests"
22. ManjubalaBisi "Software Reliability Prediction using Neural Network with Encoded Input"International Journal of Computer Applications (0975 – 8887) Volume 47– No.22, June 2012
23. Christian Ruiz "Improving Data Validation Using Machine Learning" United Nations Economic Commission For Europe Conference Of European Statisticians18-20 September 2018)
24. Jyoti Devi "A Review of Improving Software Quality using Machine Learning Algorithms" International

Journal of Computer Science and Mobile Computing A Monthly Journal of Computer Science and Information Technology ISSN 2320–088X IMPACT FACTOR: 6.017 IJCSMC, Vol. 6, Issue. 3, March 2017, pg.148 – 153

25. Jinyong Wang “Software Reliability Prediction Using a Deep Learning Model based on the RNN Encoder–Decoder”Reliability Engineering and System Safety 21 October 2017
26. Fu Yangzhen “A Software Reliability Prediction Model Using Improved Long Short Term Memory Network” 2017 IEEE International Conference on Software Quality, Reliability and Security (Companion Volume) 2017.
27. Ba J L, Kiros J R, Hinton G E. Layer normalization[J]. arXiv preprint arXiv:1607.06450, 2016.
28. Sharma, K., Garg, R., Nagpal, C. K., Garg, R. K.: Selection of optimal software reliability growth models using a distance basedapproach: Reliability, IEEE Transactions on,59(2): 266-276(2010)
29. Amin, A., Grunske, L., & Colman, A.: An approach to software
30. reliability prediction based on time series modeling: Journal of Systems and Software:, Volume: 86,Issue: 7,Pages: 1923-1932,(2013).
31. J. Alonso, J. L. Berral, R. Gavald`a, and J. Torres, “Adaptive on-linesoftware aging prediction based on machine learning,” in Procs. 40thIEEE/IFIP Intl. Conf. on Dependable Systems and Networks, 2010.
32. Mingxia Liu, LinsongMiao, and Daoqiang Zhang, “Two-Stage Cost-Sensitive Learning for Software Defect Prediction,” IEEE Transactions on reliability, Vol. 63, No. 2, June 2014.
33. J, Lee N, Baik J (2014) On the long-term predictive capability of data-driven software reliability model: an empirical evaluation.
34. Sy-Yen Kuo Michael R. Lyu Framework for modeling software reliability, using various testing-efforts and fault-detection rates Article *in IEEE Transactions on Reliability · October 2001*
35. DHARMENDRA LAL GUPTA1,2,* and KAVITA SAXENA2 Software bug prediction using object-oriented metrics Sa¯dhana¯ Vol. 42, No. 5, May 2017, pp. 655–669 , Indian Academy of Sciences
36. Raghuvanshi, K.K., Agarwal, A., Jain, K. *et al.* A time-variant fault detection software reliability model. *SN Appl. Sci.* **3,** 18 (2021). https://doi.org/10.1007/s42452-020-04015-z