International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8s
DOI: https://doi.org/10.17762/ijritcc.v11i8s.7226
***Article Received**: 24 April 2023 **Revised:** 10 June 2023 **Accepted:** 24 June 2023*

Quality Analysis of Software Applications using Software Reliability Growth Models and Deep Learning Models

[1]M. Jeevana Sujitha, [2]Dr. Kodukula Subrahmanyam
[1]Research Scholar, Department of Computer Science and Engineering,
Koneru Lakshmaiah Education Foundation, Vaddeswaram, AP, India
Email Id: jeevanasujitha.klu@gmail.com
[2]Department of Computer Science and Engineering,
Koneru Lakshmaiah Education Foundation, Vaddeswaram, AP, India
Email Id: smkodukula@kluniversity.in

Abstract: Finding the faults in the software is a very tedious task. Many software companies are trying to develop high-quality software which is having no faults. It is very important to analyze the errors, faults, and bugs in software development. Software reliability growth models (SRGM's) are used to help the software industries to create quality software products. Quality is the software metric that is used to analyze the performance of the software product. The software product which is having no errors or faults is considered the best software product. SRGM is also utilized to analyze the software quality based on the programming language. Deep Learning (DL) is a sub-domain in machine learning to solve several complex issues in software development. Finding accurate patterns from software faults is a very tedious task. DL algorithm performs better in integrating the SRGM with the DL approaches giving better results based on software fault detection. Many software faults real-time datasets are available to analyze the DL approaches. The performances of the various integrated models are analyzed by showing the quality metrics.

Keywords: Software reliability growth models (SRGM's), Deep Learning (DL), Faults, Errors, Exceptions.

I. INTRODUCTION

Software Engineering (SE) is the domain that analyzes the Quality of software by using various software functionalities. Software reliability growth models (SRGM) are used to find the progressive number of faults that occur at the time of software development, testing, and deployment. It is mainly defined as the failure-free software function for a particular period in a specific platform. SRGM aims to develop the software product without any errors and develops error-free software. SRGM mainly helps software companies to develop the best software product. These models show the relationship among the fault detection data and mathematical methods such as exponential functions. Quality is the software metric that can analyze the performance of the software product or application.

In software development, open source software has developed a great deal for better software. Based on the free availability of the operating system platform which is promoted by the overall world [1]. Usage of these open source software becomes more reliable for providing low expenses and security. This paper deals with the various SRGM models that provide error-free software with better quality. Generally, debugging is the process to find the errors in the programming code. This process is used to find and reduce the no of bugs or defects in code. Faults may occur in the software and hardware of the system. An error is the part of a computer program that leads to system failure. Errors also lead to failures in components. One component failure leads to the failure of all the other components.

Finding accurate faults or failures in real-time applications is a big task. The proposed SRGM models continuously solve the defects and develop quality software. This research is mainly focused on detecting the defects or failures or bugs etc using SRGM. Several Deep learning (DL) models are integrated with the SRGM to get accurate defect prediction. Various real-time datasets are also used to analyze the SRGM models with DL models. Improving more metrics and finding the quality of the project by using real-time software datasets with accurate defect prediction. Figure 1 shows the SRGM models that are used for detecting the software failures.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8s
DOI: https://doi.org/10.17762/ijritcc.v11i8s.7226
***Article Received**: 24 April 2023 **Revised:** 10 June 2023 **Accepted:** 24 June 2023*

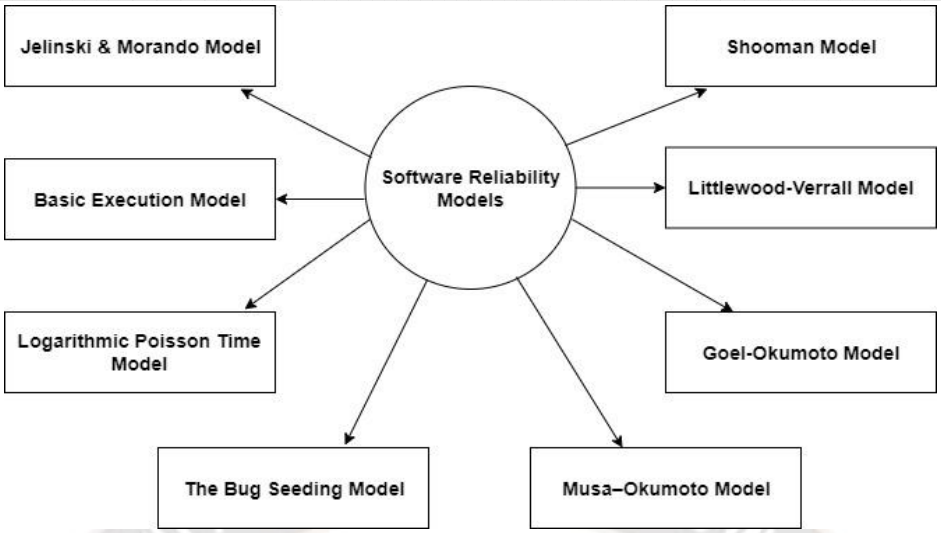

Figure 1: Types of SRGM Models

II. LITERATURE SURVEY

In this section various proposed algorithms by researchers, for detecting software faults(or) failures are discussed

C.- J. Hsu and C.- Y. Huang [2] proposed the Optimal weighted combinational models for programming unwavering quality assessment and investigation. Improved Genetic Algorithms (EGAs) are the mix of three upgraded weighted blends, to be specific weighted number-crunching, weighted mathematical, and weighted symphonious mixes. DS1-online ERP framework (WebERP) DS2-fair size programming project. DS3-Monitoring framework project. DS4-Rome Air Development Center venture. This structure is the mixes of SRGMs were proposed to work on the prescient precision of programming dependability assessment. The proposed framework is utilized to integrate an EGA with the drifting point chromosome, weighted piece transformation, and reconstructing instruments so that specialists can have an effective method for choosing the ideal number of consolidated models. A predetermined number of weighted tasks are integrated into EGA. It is noticed that the boundary assessments could turn out to be more muddled and dreary with the utilization of a rising number of joined models.

J. Li et al., [3] proposed the deformity forecast through Convolutional Neural Network (DP-CNN). This approach is applied to tera-PROMISE Repository datasets that are freely accessible storehouses gaining practical experience in computer programming research. A deformity expectation system called DP-CNN (Defect Prediction by means of Convolutional Neural Network) uses CNN for robotized include age from source code with the semantic and underlying data saved. Additionally, we utilize word installing and join the CNN-learned highlights with conventional carefully assembled highlights, to additionally further develop our imperfection prediction. More tests are directed more ventures, and stretch out our technique to other programming dialects like Python. It is promising to take on profound figuring out how to other computer programming errands, for example, code fruition and code clone discovery, which will be our future work.

A. N. Lam et al., [4] proposed the Software deformity forecast (SDP) model in light of LASSO-SVM. Tackling the unfortunate expectation precision of most SDP models is proposed. A SDP model joined with least outright worth pressure with determination technique and backing vector machine (SVM) calculation is proposed. The dataset is a constant dataset called a NASA informational index. The attributes of the product imperfection informational index, the LASSO technique is utilized to accomplish information dimensionality decrease. The product deformity information in the high-layered Euclidean space is reproduced in a low-layered space, and the complex construction of the product imperfection information in the high-layered Euclidean space is held, in this manner getting the comparing implanted map, subsequently understanding the information. The help vector machine (SVM) is utilized as the fundamental classifier, and the SVM is prepared with the diminished layered information. The SDP model of LASSO-SVM. Issues happen when cross-approval chooses the SVM ideal boundaries, that is to say, there might be different arrangements of boundaries relating to the most elevated confirmation characterization precision rate. The treatment measures, for this situation, are the following exploration bearing.

K. N. Rao et al., [5] proposed a clever under examining technique (USS) for effective programming imperfection examination of slanted disseminated information. USS

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8s
DOI: https://doi.org/10.17762/ijritcc.v11i8s.7226
***Article Received**: 24 April 2023 **Revised:** 10 June 2023 **Accepted:** 24 June 2023*

approach comprises of four stages: 1. Procuring information 2. Information pre-handling 3. Learning 4. Results. Component of Proposed Approach (I) Proposed approach used novel informational collection sifting system for viable commotion eliminate. (ii) Proposed approach used novel characterization calculation for better expectation. (iii) To utilize better assessment estimation boundary to obtain improved outcome. (iv) To diminish the product advancement cost, time and exertion. The proposed approach is applied on 16 continuous Datasets, for example, ar1, ar3, ar5, cm1, desharnais, jm1, kc1, kc1-deficient, kc1-top5, kc3, mc1, mc21, mw1, pc1, pc3 and reuse datasets. The proposed approach USS (under inspecting technique) for programming deformity investigation considers the product imperfection examination as an issue of class lopsidedness learning. This approach is intended to change over the first imbalanced dataset into a bunch of new datasets by proficient USS. The new datasets development is the primary danger to the build legitimacy. The boundaries of expectation models might be the danger to the inner legitimacy. Dataset quality might be the main danger to the outside legitimacy. Dynamic and dubious informational collections of programming process advancement in the ongoing climate can be broke down.

S. Jha et al., [6] Proposed a model for programming viability measurements expectation. This model used to identify and redress the issues of Software Development Life Cycle (SDLC). Remedial upkeep of programming is simply limited to finding absconds. It additionally considers receptive alteration of programming item after the imperfection has been found. This is done as such as to guarantee that product is as yet usable in a changed or evolving climate. The thought is to diminish the product remedial support exertion. In this way legitimate implementation of guidelines during each period of SDLC is obligatory. Huge Datasets Several research holes are distinguished by programming viability measurements expectation. From the recently distributed works and the analyses, obviously our proposed LSTM calculation outflanks the other AI calculations. It can moreover be seen that in the majority of different works, AI estimations disregard to show which estimations impact the item feasibility, though our LSTM calculations estimates this precisely. A fluffy profound brain framework is intgerated to work on the accuracy of the outcomes got by means of this strategy is tantamount to results gotten through different techniques. More number of datasets are required. Another strategy for accuracy is required. More SDLC is thought of.

Z. Ke et al., [7] proposed the numerous change-point model methodology. This work principally utilizes the Parr-bend with different CPs to portray the TE utilization in light of NHPP system and how to perform further programming dependability expectation and assessment. A few numerical properties of Parr-bend with numerous CPs and the proposed SRGM are likewise introduced and examined. In this work the outcomes are introduced from an exact investigation of the proposed models applied to two certifiable contextual investigations. From the consequences of Laplace pattern test, we have independently taken single CP and 2-CPs into the contemplations of investigations. The principal informational index (DS1) we utilized was accounted for by Dr K. Okumoto of Alcatel/Lucent Technologies and it was gathered from the trial of soundness The subsequent informational collection (DS2) was the System T1 information of the Rome Air Development Center (RADC) projects. This work utilizes the Parr-bend model with various CPs to portray the utilization of TE and how to perform further programming dependability examination. A few investigations are performed in light of two genuine informational collections, and assessment results show that our proposed model can further develop the expectation capacity of TE utilization and programming unwavering quality. What's more, in light of the proposed model, we likewise present a product financial strategy which gives a thorough examination of programming in view of cost and unwavering quality. The strategy can proficiently help project manager(s) and test team(s) in choosing when to quit testing for market discharge with impeccable timing. Still there are a few open issues considering present realities basically conceivable to gauge the number and recognize the area of various CPs as an element of time, which mirrors an adjustment of dissemination during the cycles of programming testing, investigating, and activity. The chose capability could have to require data about the disappointment information, for example, the absolute number of CPs in the disappointment information, what is the SDLC (like testing, troubleshooting, or activity), and the highlights of the phase(s).

Wang, J et al., [8] proposed the product dependability forecast utilizing a profound learning model in light of the RNN encoder-decoder. This model for the most part founded on the RNN encoder-decoder to anticipate the issue number in programming and survey programming dependability in programming testing. Consequently, the RNN encoder-decoder hypothesis embraced the profound NN model in view of the RNN encoder-decoder has four layers, in particular, one information layer, two secret layers, and one result layer. Considering that the scope of programming disappointment information values can be very huge and the information values are unevenly conveyed, we then scale programming disappointment information values into the scope of 0.1-0.9 to precisely foresee the issue number in the product. H. Wang et al., [9] proposed the new profound learning model to anticipate the deficiencies for better programming unwavering quality. To expand the presentation of proposed approach the

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8s
DOI: https://doi.org/10.17762/ijritcc.v11i8s.7226
***Article Received**: 24 April 2023 **Revised:** 10 June 2023 **Accepted:** 24 June 2023*

RNN encoder-decoder is utilized to anticipate the flaws in the product. NN models shows that better methodology contrast and existing models. A. N. Lam et al., [10] proposed a novel approach that utilizes the DNN to detect the bugs present in the real-time datasets. The proposed approach is the combination of rVSM and IR approach. The rVSM is the approach that collects the features from textual similarities among the bug reports and also from source files. This is applied to real-time datasets to fix the bugs from the existing approaches. This approach shows the high accuracy of detecting the bugs from the available datasets. A. N. Lam et al., [11] is the combined model which is the combination of the rVSM and IR approach, and features are extracted by using the HyLoc. Based on the previous history of bug fixing the proposed approach follows. The proposed approach achieved better performance compared with existing approaches. Kuldeep Singh Kaswan et al., [12] proposed the new SRGM model that handles the removal of faults from real-time applications. This approach analyzed the failures that occur at the time-based. At the time of development, the faults are identified. The proposed system analyzed the software faults and removes them from the code at the time of development. The proposed approach removes the software failures more compare with the existing approach.

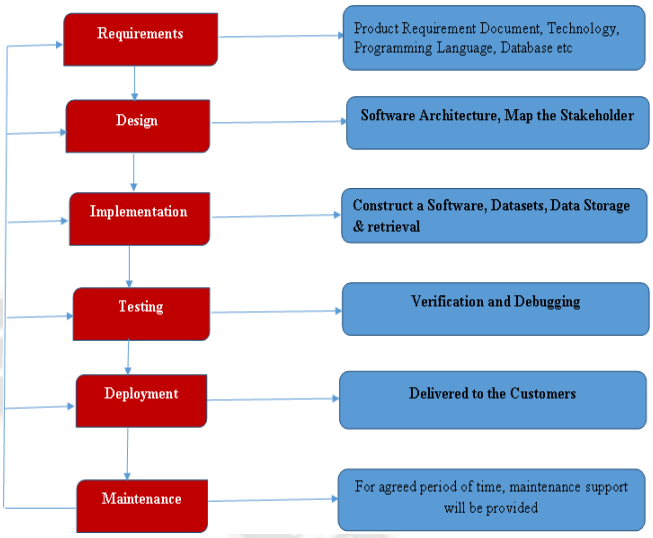

Figure 2: Software Development Process

Figure 2 shows the software development process based on the given steps. Every stage have some functionalities to develop and execute software project.

TABLE I: EXPLAINS SEVERAL DL AND SRGM MODELS

Authors	Proposed Model	Projects and Datasets	Project Description
I	Recurrent Neural Network (RNN)	jEdit and Apache Ant	This model mainly focused on detecting the defect-prone code blocks that are more specified to measure the reliability based on modules. To improve the performance of RNN algebraic tools are introduced to develop the reliability model accurately.
Yanli Shao et al., [16]	Cross-Project Defect Prediction (CPDP) and Cross-Company Defect Prediction (CCDP)	12 project datasets in NASA and PROMISE	Detecting the defects and modifying the defects gives improved software reliability. The proposed approach focused on redundancy and extracting the parameters by using principal component analysis (PCA).
Yiwen Zhong et al., [17]	CPDP-orient-object, semantic, and structural metrics (OSS)	AEEEM data set contains 5 open source projects	The proposed model is CPDP-OSS gives a better performance to detect the defect code blocks based on the available data. Many trails are applied to real-time datasets.
Zhang Xiaonan et al., [18]	Optimization PSO-LSSVM	3 Real time project datasets.	This approach focused on accurate prediction and analysis of software reliability. This shows the better precision and efficiently on small datasets.

III. BACKGROUND AND RELATED WORKS

A. Reviews of SRGM's

1)Jelinski and Morando (JM) Model

The JM model is also called as Markov model that has changed a lot for the existing JM model. This is also one of the Binomial models that detect the accurate debugging step to detect the bug and remove the bug [13].

The software failure rate at the j^{th} failure interval is given by,

$$\alpha(tj) = \phi[k-(j-1)] \text{where } j = 1,2,3,\ldots,k \quad (1)$$

$\phi =$ Proportionality constant indicating the failure rate provoded by every fault.

$k =$ the initial number of errors in the software.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8s
DOI: https://doi.org/10.17762/ijritcc.v11i8s.7226
***Article Received**: 24 April 2023 **Revised:** 10 June 2023 **Accepted:** 24 June 2023*

t_j = the time between (j − 1)th and (j)th failure.

Software Reliability function

$$= e - \phi[K - (J - 1)]t_j \qquad (2)$$

2)Basic Execution Time Model

This model is most popular model that can be used to improve the software reliability to the software projects. At the time of execution this model finds the failures based on the behavior of the code. The failure behavior is also called as non-homogeneous Poisson process, which means the associated probability distribution is a Poisson process whose characteristics vary in time.

Attributes used in the Basic Execution Model:

Failure intensity (λ): No of failures per time unit

Execution time (Γ): time when the program is running

Mean failures (ψ): mean failures we get in a time interval

In this model, the mean failures which we got ψ is indicated in terms of the execution time (Γ) as

$$\psi(\Gamma) = X_0 Y\left(1 - e^{-\frac{\lambda_0}{X_0}}\right)\Gamma \qquad (3)$$

3)Logarithmic Poisson Time Model (LPTM)

This is one among the software reliability models. This model is also works on the time based execution model that finds the faults present at the time of software development applications. LPTM represents the failures by using the following equations.

Time (t) =0 at the time of no failures identified. i.e., K (M (0) =0) =1.

The intensity of the failures are identified based on the decrease of failures.,

$\lambda(\Gamma) = a_0 a_1 \exp\left(\frac{-\mu(\Gamma)}{a_0}\right)$, where a_0 a_1 Is the starting failure intensity and a_0^{-1} is copied the failure.

The overall failures are identified within the time are $\Gamma, M(\Gamma)$, follows a Poisson Process.

4)Goel-Okumoto (GO) Model

The overall failures are identified by time t follows a Poisson distributed with the mean value function μ (t). this mean value has limited conditionsμ (0) = 0 and $\text{Lim}_{t\to\infty}\mu(t) = N < \infty$.

Several SRGM models are discussed in the above section. Some SRM models are time-based models that are based on the execution time model, i.e., the overall time taken for the processing of defect detection using actual CPU execution time. Based on the time the intensity of the failure is reduced, if time increases the failure intensity decreases. Some models remove the faults that are identified at the time of execution.

X(t)=ai(1-e^{-bt}),a>0,b>0

B)Evaluation Measure for Software Relaibility

The performance metrics are most widely analyzing the performance of various SRGM models and Deep Learning (DL) algorithms. Every parameter is very important to analyze the model and dataset. The goodness of fit is one of the performance analyzed metrics based on the applied mathematical model. The following are metrics that are used in various SRGM models. we will use the following constraints to evaluate and analyse the performance of the SRGM's

1)Mean Absolute Error (MAE):

Mean Absolute Error refers to the significance of difference between the prediction of an observation and the actual value of that observation. MAE takes the average of absolute errors for a group of predictions and observations as a measurement of the significance of errors for the entire group. MAE can also be referred as L1 loss function.

MAE=Actual value-Predicted value

2)Mean Squared Error (MSE): Mean squared error is also known as Mean Squared Deviation(MSD).It is the measure that evaluates the difference measure between observed value and Predicted value.When the model has no error ,the MSE value is also zero. As model has more errors the MSD value also increases.It is mainly used in Regression.

$$x = \frac{1}{n}\sum_{g=1}^{n}(xg - \overline{x})\,2$$

3)Root Mean Square Error (RMSE):RMSE is the standard deviation of Predicted values (or) errors.Standard deviation is a central tendency measure that describe how data is spread.It is calculated by the square root of variance.Variance is evaluated by the average of squared difference of mean.

$$\text{RMSE}=\sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

4)Prediction Error (PE): The overall difference among the observation and prediction of total failures at any time is initialized as i and it is represented as PE_i. If the PE_i value is low then the goodness of fit is better.

$$\text{PE} = \frac{\text{Calculated Value} - \text{Estimated Value}}{\text{Calculated Value}} \times 100$$

5)Bias: Bias is the metric that gives the average of PE's. If the bias value is low, goodness of fit is better.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8s
DOI: https://doi.org/10.17762/ijritcc.v11i8s.7226
***Article Received**: 24 April 2023 **Revised:** 10 June 2023 **Accepted:** 24 June 2023*

6)Variation: This metric mainly shows the standard deviation of PE and it is called as variation.

$$\text{Variation} = \sqrt{\left(\frac{1}{N-1}\right)\sum(PE_i - \text{Bias})^2}$$

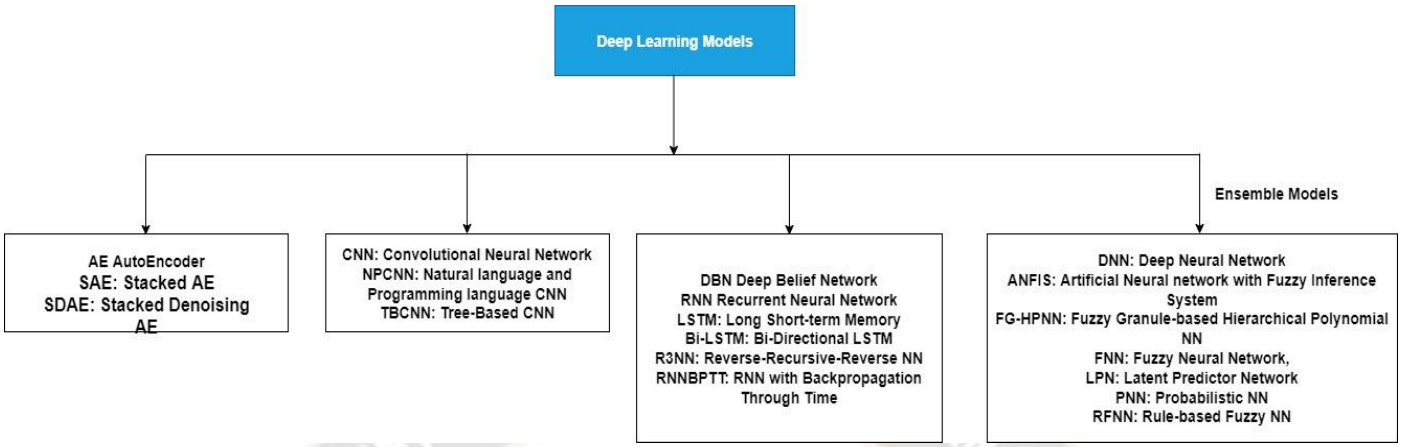

Figure 3: Deep Learning Models for Finding the Software Failures

C)Role of Deep Learning in Software Engineering

Deep Learning (DL) is efficiently achieved the better results in many applications. Effective training and testing of large and complex datasets are processed by the DL models. Some programs may fail after execution of code/script defects that decreases the quality of the software projects or applications. DL is the model which is used to process the software failure data with multiple layers. There are many DL models are present, according to the usage of datasets the specific DL models are used to process the complex failures dataset. In DL several pre-trained models are also used to process the failure data. But these pre-trained models are to be analyzed with the failure data. Using DL models gives quality software that provides error-free software projects. Using the DL models is new for the software reliability growth models (SRGM) to find accurate software failures from the real-time datasets.

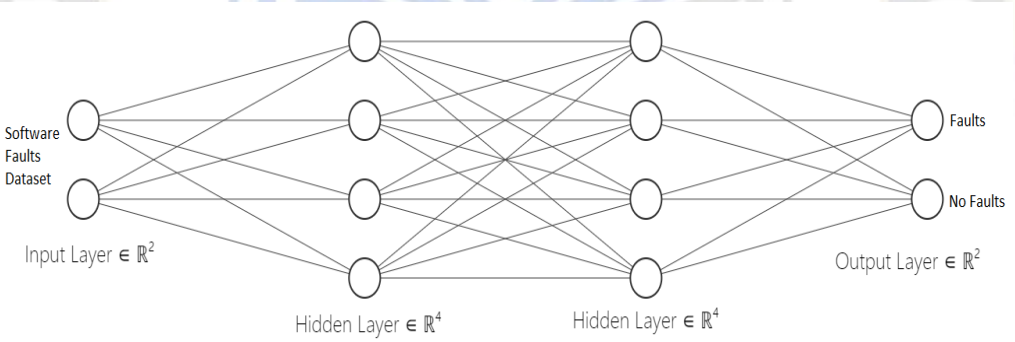

Figure 4: CNN Model for finding the Faults

Figure 4 explains the process of CNN model to find the faults from various real time datasets. This is the architecture used to find the accurate fault patterns by using the layers present in this model. The input layer mainly analyzes the given dataset and remaining layers take the input layer data and extract the features from the given dataset. The output layer classifies the faults and no faults by analyzing the data. Figure 5 explains the process of finding the failures at the time of software development. This is the step-by-step process that shows the training and pre-trained model, preprocessing technique, selection of the learning model, and analyzes the results. To increase the performance of proposed models an effective pre-processing techniques are integrated.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8s
DOI: https://doi.org/10.17762/ijritcc.v11i8s.7226
***Article Received**: 24 April 2023 **Revised:** 10 June 2023 **Accepted:** 24 June 2023*

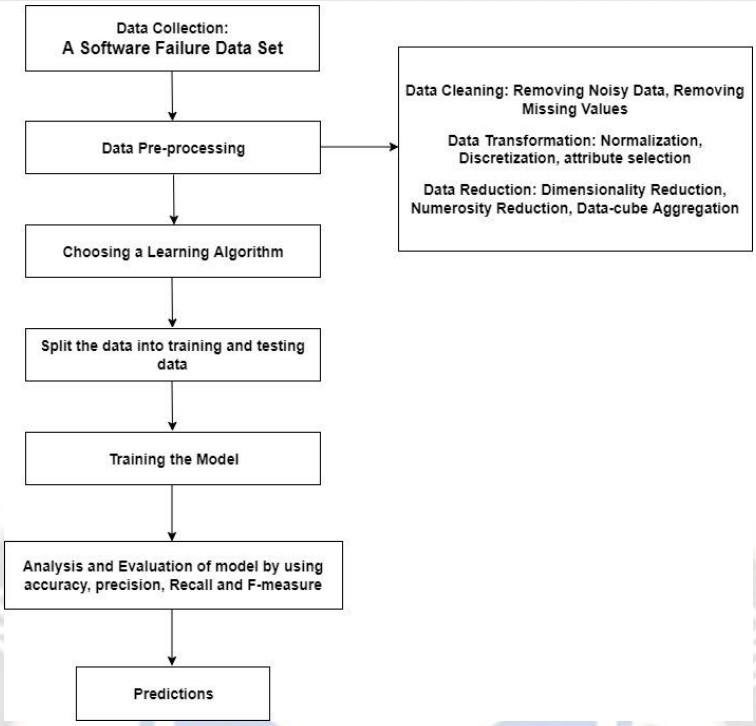

Figure 5: Flow of DL models for detecting the software failures data

IV. EXPERIMENTAL RESULTS

Finding the faults in various real time datasets gives the compatibility of several SRGM models. Some authors tried to develop the SRGM models to analyze the software failures. Deep Learning (DL) models give the accurate fault finding in real time software failure datasets. In this section, various real-time datasets are analyzed and gives the better fault prediction. Some researchers are focused on analyzing the threats in several software applications which are in the form of codes.

TABLE II: PERFORMANCE ANALYSIS OF VARIOUS SRGM MODELS

Authors	Proposed Model	Datasets	Performance
Suqi Zhang et al., [19]	Enhanced Whale Optimization Algorithm (EWOA) with Support Vector Machine (SVM)	E-Commerce transaction datasets	RMSE for SVM-97% and R_2 gives the 99.27
C. Lin et al., [20]	A new rate-based queuing simulation framework	Three real datasets from Apache and GNOME projects	Measuring the fixed bugs: 101.35 Sec
Maxime Frydman et al., [21]	AutSEC	Real-Time threat Datasets	Accuracy for detecting threats: 97.45
Chin-Yu Huang et al., [22]	SIQinU (strategy for understanding and improving quality in use)	Labeled Software project threat dataset	Average Effectiveness of 85.6%

V. EVOLUTION RESULTS

In this paper, the experimental results are analyzed from various software fault models. The performance is calculated by showing RMSE, PE, MSE etc. By using Python programming language the application is developed to predict the bugs and errors in the real time dataset.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8s
DOI: https://doi.org/10.17762/ijritcc.v11i8s.7226
***Article Received**: 24 April 2023 **Revised:** 10 June 2023 **Accepted:** 24 June 2023*

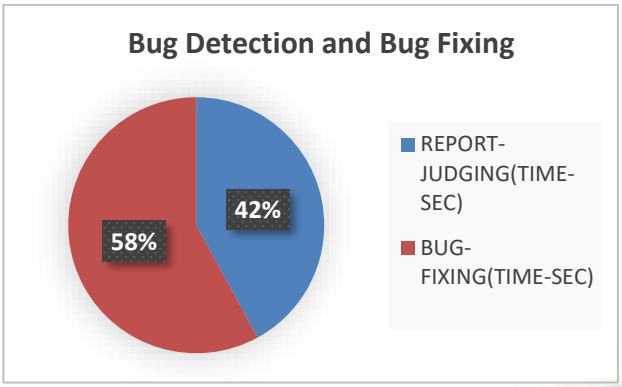

Figure 6: Bugs belongs to Bug Detection and Bug Fixing (Sec)

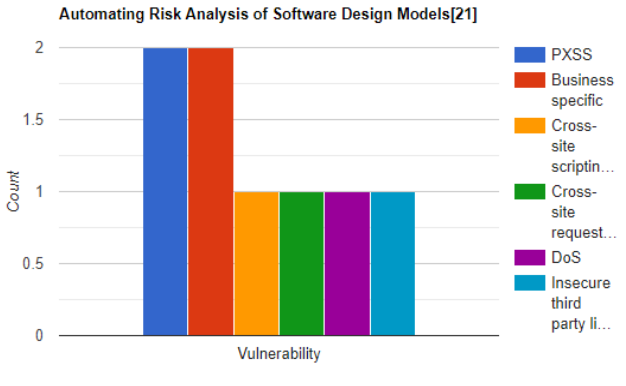

Figure 7: Total count of Vulnerabilities based on Types

VI. CONCLUSION

Software reliability growth models (SRGM) combined with the Deep Learning (DL) algorithms give the best fault detection in real-time datasets. Huge research has to be done by combining these two domains. Recently DL plays a major role in finding the issues in software reliability models and improving the detection of faults, failures, and bugs. Step-wise fault detection is done with these integrated models. Advanced pre-trained models, pre-processing techniques, and feature extraction techniques are used to find accurate bug detection. Generally, DL models require huge training for data processing and also for noise removal. These approaches also remove the faults by applying the interpreter-based error removal system. Several parameters are used to show the performance of various DL and SRGM models. An Ensemble software defect prediction is required to process the large and real-time datasets. In future, combined software reliability with machine learning (ML) and Deep Learning (DL) is required to overcome several issues in finding the software faults and failures in software development. This should be applicable to latest programming languages.

REFERENCES

[1] S. K. Jha, A. K. D. Dwivedi and A. Tiwari, "Reliability models and open source software: An empirical study," 2010 IEEE International Conference on Computational Intelligence and Computing Research, 2010, pp. 1-5, doi: 10.1109/ICCIC.2010.5705779.

[2] C.-J. Hsu and C.-Y. Huang, "Optimal weighted combinational models for software reliability estimation and analysis," IEEE Transactions on Reliability, vol. 63, no. 3, pp. 731–749, 2014.

[3] J. Li, P. He, J. Zhu and M. R. Lyu, "Software defect prediction via convolutional neural network", Proc. 17th IEEE Int. Conf. Softw. Qual. Rel. Secur., pp. 318-328, Aug. 2017.

[4] K. Wang, L. Liu, C. Yuan and Z. Wang, "Software defect prediction model based on LASSO--SVM," Neural Computing and Applications, pp. 8249–8259, 2021.

[5] K. N. Rao and C. S. Reddy, "A novel under sampling strategy for efficient software defect analysis of skewed distributed data," Evolving Systems, vol. 11, pp. 119–131, 2020.

[6] S. Jha et al., "Deep Learning Approach for Software Maintainability Metrics Prediction," in IEEE Access, vol. 7, pp. 61840-61855, 2019, doi: 10.1109/ACCESS.2019.2913349.

[7] Z. Ke and C. Y. Huang, "Software reliability prediction and management: A multiple change-point model approach", Qual. Rel. Eng. Int., vol. 36, no. 5, pp. 1678-1707, Jul. 2020.

[8] Wang, J.; Zhang, C. Software reliability prediction using a deep learning model based on the RNN encoder-decoder. Reliab. Eng. Syst. Saf. 2018, 170, 73–82.

[9] H. Wang, L. Wang, Q. Yu and Z. Zheng, "Learning the evolution regularities for big service-oriented online reliability prediction", IEEE Trans. Serv. Comput., vol. 12, no. 3, pp. 398-411, May/Jun. 2019.

[10] A. N. Lam, A. T. Nguyen, H. A. Nguyen and T. N. Nguyen, "Bug Localization with Combination of Deep Learning and Information Retrieval," 2017 IEEE/ACM 25th International Conference on Program Comprehension (ICPC), 2017, pp. 218-229.

[11] A. N. Lam, A. T. Nguyen, H. A. Nguyen and T. N. Nguyen, "Combining Deep Learning with Information Retrieval to Localize Buggy Files for Bug Reports (N)," 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE), 2015, pp. 476-481.

[12] Kuldeep Singh Kaswan, Sunita Choudhary, Santar Pal Singh, Anil Audumbar Pise, Simon Karanja Hinga, "A New Variant of JM Software Reliability Model", Scientific Programming, vol. 2022, Article ID 7257564, 11 pages, 2022.

[13] H. Joe and N. Reid, "On the Software Reliability Models of Jelinski-Moranda and Littlewood," in IEEE Transactions on Reliability, vol. R-34, no. 3, pp. 216-218, Aug. 1985.

[14] S. Campodonico and N. D. Singpurwalla, "A Bayesian analysis of the logarithmic-Poisson execution time model based on expert opinion and failure data," in IEEE Transactions on Software Engineering, vol. 20, no. 9, pp. 677-683, Sept. 1994.

[15] Hao Zhang, Jie Zhang, Ke Shi, Hui Wang, "Applying Software Metrics to RNN for Early Reliability Evaluation", Journal of Control Science and Engineering, vol. 2020, Article ID 8814394, 10 pages, 2020.

[16] Yanli Shao, Jingru Zhao, Xingqi Wang, Weiwei Wu, Jinglong Fang, "Research on Cross-Company Defect Prediction Method to Improve Software Security", Security and Communication Networks, vol. 2021, Article ID 5558561, 19 pages, 2021.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 8s
DOI: https://doi.org/10.17762/ijritcc.v11i8s.7226
***Article Received**: 24 April 2023 **Revised:** 10 June 2023 **Accepted:** 24 June 2023*

[17] Yiwen Zhong, Kun Song, ShengKai Lv, Peng He, "An Empirical Study of Software Metrics Diversity for Cross-Project Defect Prediction", Mathematical Problems in Engineering, vol. 2021, Article ID 3135702, 11 pages, 2021.

[18] Brian Moore, Peter Thomas, Giovanni Rossi, Anna Kowalska, Manuel López. Deep Reinforcement Learning for Dynamic Decision Making in Decision Science. Kuwait Journal of Machine Learning, 2(4). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/219

[19] Zhang Xiaonan, Yang Junfeng, Du Siliang, Huang Shudong, "A New Method on Software Reliability Prediction", Mathematical Problems in Engineering, vol. 2013, Article ID 385372, 8 pages, 2013.

[20] Suqi Zhang, Hsiung-Cheng Lin, Xinxin Wang, "Forecast of E-Commerce Transactions Trend Using Integration of Enhanced Whale Optimization Algorithm and Support Vector Machine", Computational Intelligence and Neuroscience, vol. 2021, Article ID 9931521, 12 pages, 2021.

[21] C. Lin and Y. Li, "Rate-Based Queueing Simulation Model of Open Source Software Debugging Activities," in IEEE Transactions on Software Engineering, vol. 40, no. 11, pp. 1075-1099, 1 Nov. 2014.

[22] Chaudhary, A. ., Sharma, A. ., & Gupta, N. . (2023). Designing A Secured Framework for the Steganography Process Using Blockchain and Machine Learning Technology. International Journal of Intelligent Systems and Applications in Engineering, 11(2s), 96–103. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/2512

[23] Maxime Frydman, Guifre Ruiz, Elisa Heymann, Eduardo Cesar, Barton P. Miller, "Automating Risk Analysis of Software Design Models", The Scientific World Journal, vol. 2014, Article ID 805856, 12 pages, 2014.

[24] Chin-Yu Huang, Hareton Leung, Wu-Hon Francis Leung, Osamu Mizuno, "Software Quality Assurance Methodologies and Techniques", Advances in Software Engineering, vol. 2012, Article ID 872619, 2 pages, 2012