International Journal of Latest Trends in Engineering and Technology
Special Issue April-2018, pp. 133-140
e-ISSN:2278-621X

SOFT COMPUTING TECHNIQUES FOR ENHANCING SOFTWARE RELIABILITY

Dhavakumar P[1], Shankar.S[2], Vikram Pandi M[3]

Abstract— Software is a major part of today's technical systems, which is developed in various fields. Reliable software is the primary and significant criterion for all software industries, developers and end users. There is a need to enhance quality software software model according to the type of software as well as the efforts needed to complete risk-less Software Development (SD) in a predictive way. In this paper, we propose a solution to the problem by using large number of soft computing techniques for proving the software reliability. Our main goal is to give failure free access software system in the entire environment. At the end, the task cumulative software risk will be calculated using fuzzy logic to improve the Software Reliability.
Index Terms— Software Risk, Reliability, Soft Computing, Fuzzy Logic, SD.

1. INTRODUCTION

In the Software engineering, Software reliability is an active research area over the past forty five years and still several challenges and transparent questions still exist. It_s a core part in software quality along with functional, usability, maintainability, serviceability performance and documentation. Software reliability is defined as the probability of failure-free operation of the software over a specified period of time in a specified environment [1]. Software Reliability is also an important factor which affects the system reliability. Software reliability is different from hardware reliability because it defines the design perfection, than manufacturing perfection. The high complexity of software is a major contributing factor of Software Reliability problems.

The study of software reliability can be categorized into three parts: modelling, measurement and enhancement. This modelling has matured to the point that useful results can be obtained by applying appropriate models to the problem. There are many reliability models, but no single model can capture a necessary amount of the software characteristics. Software reliability cannot be directly measured, because failure-free operations occurred during certain period of time in specified environment. Development process, failures and faults are found in all factors related to software reliability. Software reliability improvement is hard. The hard of the problem is identified from insufficient understanding of software reliability and in common, the characteristics of software [2]. Until now there is no good way to conquer the complexity problem of software. In the last few years many research studies have been carried out in this area of software reliability modelling and forecasting. The issues of software reliability are focused on soft computing techniques for developing and maintaining software systems to enhance it with proper measurement.

There are various soft computing modelling tools available to measure the reliability. Such as fuzzy logic models neural networks, Genetic algorithms (GA) based neural networks, Recurrent neural networks, Bayesian neural networks, and support vector machine (SVM) based techniques, and etc. In this paper we choose the fuzzy logic as a case tool to measure the reliability of various phases of software development. The measureable unit quantitatively represents without excessive limitations. Software quality is another important aspect for users, developers, and project managers. Various research persons have worked in developing suitable models that define software quality in different perspectives as described in ISO/IEC 9126 Model [3], Boehm_s Model [4], Dromey_s Model [5] and the FURPS Model [6]. Quality, not only describes and measures the functional aspects of the software, but also describes extra functional properties. In order to deal with fuzziness in quantifying the actual software parameter, real multi variable has been used.

1.1 Software Reliability: Software Reliability is the probability of failure free software. The software failures are introduced by the programmers, designers, system analysts, and managers during the different phases of SDLC. There are number of software reliability models which are used for different purposes. These models are explained below

(a) Jelinski Moranda Model - The Jelinski Moranda model is the earliest model in software reliability. It was given in 1972. It is one among the failure models. It assumes N software faults at the start of testing, failure occur purely at random and all the faults give equally to cause a failure during testing. It also assumes that the fix time is negligible and the fix for each failure is ideal. The software product_s failure rate improves by the same amount at each fix.

[1] Department of Computer Science and Engineering, Periyar Maniammai Institute of Science and Technology, Thanjavur.
[2] Department of Computer Science and Engineering, Periyar Maniammai Institute of Science and Technology, Thanjavur.
[3] Department of Computer Science and Engineering, Periyar Maniammai Institute of Science and Technology, Thanjavur.

(b) Littlewoods Model - It is similar to Jelinski Moranda Model but it assumes that different faults have different size. Large sized defects tend to be detected and set right earlier. As the number of errors is driven down with the progress in the test, this model is the average error size causing a law of deteriorating return in debugging.
(c) GoelOkumoto Imperfect Model - The J-M model assumes that the fix is insignificant and that the fix for each failure is perfect. It assumes perfect debugging .In practice this is not all times. In the process of fixing a defect new defects may be injected. This model proposed imperfect debugging model to beat the limitation of the assumption. In this model the hazard function during the interval between the (i-1)st and the failure is given

$$Z(ti) = [N - p(i-1)]\lambda \qquad (1)$$

Where N is number of faults at start of testing P is probability of imperfect debugging λ failure rate per fault
(d) Goel Okumoto Non Homogeneous Poisson Process Model - The NHHP model was given in 1979.It is concerned with modelling the number of failures experiential in given testing intervals. It defines the cumulative number of failure observed at time t. N (t) can be modelled as a non-homogenous poisson process with the time dependent failure rate.
(e) The Delayed S and Inflection S Models - With the assist of defect removal process Yamada said that a testing process consists of not only a fault detection process but also fault isolation process because of the time needed for failure analysis major delay can occur between time of first failure observation and the time of reporting. They offer the delayed S formed reliability development model for such a process in which the observed growth curve of increasing number of detected defects is S-shaped. The model stands on the no homogeneous poison process but with a different mean value function to reproduce the delay in failure reporting.

2. REVIEWOFLITERATURE

Even though ensuring a reliable software application as a whole is a very difficult task, a software system has two critically acclaimed user requirements, one is reliability and the other one is availability. ―Software reliability is define d as the probability of the failure-free operation of a software system for a particular period of time in a specified environment‖. Reliability of software system makes sense if non-performance of the system has cutting impact, and availability of the software system makes sense if downtime has severe impact. This traditional software reliability model thinks only timing and failure rate but this approach is imperfect as the failure rate should be associated to application complexity and test effectiveness as well [12]. Software unreliability is the result of defective design and architecture of the system, issues arises due to human failures.
Akira Hada (2011)describes in his research paper which presents various models to estimate reliability for a future profile with increased pressure using the current observations to develop a model for future reliability. For the foreseeable future the apparatus itself will not change, and thus the increase of the load or stress, that the system is exposed to, may decrease reliability and increase preservation. The model is required to determine the impact of these future profiles. There are several years of research conducted and their results on many model structures for the predictive software reliability were assessed and evaluated for reflection for the further research in the field of software engineering for the effective evaluation of the software reliability. Many problems and issues were reported over practicing these model arrangements that made analysis and design of such model a quite challenging process to complete [7, 8, 9]. For example:

1. Missing or incomplete data
2. Large number of variables or unused extra variables.
3. Strong co-linearity (cohesion) between/among the variables.
4. Complex linear/non-linear relationships between model variables.
5. Outliers and small sizes of the data sets used for the model evaluation.

Logging-fortified quality assurance - The incapability to reproduce a bug is one of the biggest frustrations for every software engineer. By enabling classification functionality, you can enable a better QA process; the QA person can augment the bug report with the connected log and thus give developers a better chance at reproducing the behaviour. Nonintrusive logging and tracing functionality is a cooperative tool in understanding system behaviour. The conventional implementation of the equivalent functionality is so unwieldy that systematic implementation rarely happens [10].
Cai et al. [13] advocated the growth of fuzzy software reliability models in situate of probabilistic software reliability models (PSRMs). Their disagreement was based on the proof that software reliability is fuzzy in nature. A demonstration of how to develop a fuzzy model to distinguish software reliability was also presented. Karunanithi et al. [14] carried out a detailed study to give details the use of connectionist models in software reliability growth prediction. It was shown through empirical results that the connectionist models adjust well across different datasets and exhibit better predictive accuracy than the well-known diagnostic software reliability expansion models.
Software quality involves a planned and methodical set of activities to ensure the effectiveness of software. It consists of various sub topics like software quality assurance, quality engineering and quality control. According to the IEEE 610.12 standard [11], software quality is a set of attributes of a software system and is defined as: 1.The degree to which a system, component, or process meets specified requirements. 2. The degree to which a structure, component, or process meets customer or user needs or expectations. 3. Quality also contains of the factors leading to the satisfaction of its requirements.

The quality of the software is measured in terms of its capability to fulfil the needs of the users and also its ability to achieve the developer_s goals. Quality is mostly studied by quality models. The quality model describes the set of characteristics, which are the source for establishing the quality requirements and for assessing software quality. In the present paper, the ISO/IEC 9126 Model [10] has been considered as the base model.

3. COMPUTING METHODOLOGY

Soft computing is an association of computing method that includes at its basic member of fuzzy logic, chaos theory, neuro-fuzzy, probabilistic and evolutionary computing [5]. Soft computing is a technique associated with the idea of natural and artificial. Several theory and implementations exists in many areas by this computing. In this section we have discussed the classification of existing soft computing techniques as below figure 1.

Neural Networks: According to Nigrin a neural network is a circuit collected of a very large number of simple processing elements that are neurally based. Each building block operates asynchronously, on local information; thus there is no overall system clock.

Fuzzy logic: Developed by Lotfi A. Zadeh, University of California in Berkeley. It is a multi valued logic that allows transitional values to be defined between conventional evaluations like true/false, yes/no and low/high, etc. [15,16]. The most important application area of fuzzy logic has been in control field. Fuzzy control having been successfully applied to numerous problem

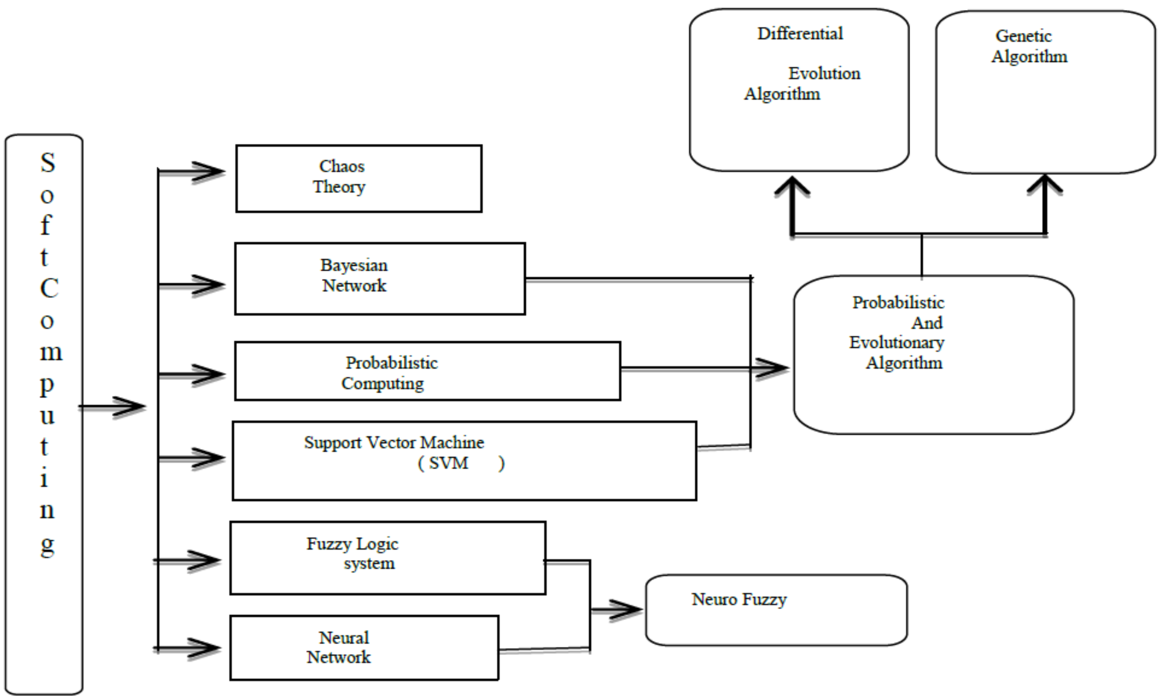

Figure 1. Soft Computing Methods

Evolutionary computing: Evolutionary computing can be viewed as an adaptation of a probabilistic approach based on principles of natural evolution [17]. It can also be defined as the stochastic search and optimization heuristic approach derived from the classic progress theory, which are implemented on computers in the majority of cases [18]. Evolutionary algorithms have been effectively applied to numerous problems from different domains, bioinformatics.

Bayesian network: Bayesian networks are graphical models for reasoning under uncertainty, where the nodes represent variables (discrete or continuous) and curves represent direct connections between them. Various applications such as the force of management style on statistical efficiency.

Chaos theory: A deterministic system is said to be chaotic whenever its evolution delicately depends on the initial conditions. This property implies that two trajectories promising from two different closes by initial conditions separate exponentially in the course of time. The necessary needs for a deterministic system to be chaotic are that the system must be nonlinear, and be at least three dimensional [19].

4. USABLE METHODOLOGY OF DIFFERENT SOFT COMPUTING TECHNIQUES

In recent years, many papers have been presented in various models for software reliability. The influence of the external parameters and other peculiarities of a software reliability growth model can be eliminated if we have a system that can develop its own model from the past failure history of the software system, which we can develop very simply by using Artificial Neural Networks and Fuzzy logic based systems.

4.1Neural networks

Neural networks are beginner's model of the biologic neuron system. It is particularly parallel distributed processing system prepared up of highly interconnected neural computing elements. Neural network has been applied for parameters assessment of the formal model and self-learning process in order to predict the future outcomes. Back-error propagation is one of the most widely used neural network paradigms and has been applied successfully in a broad range of areas [22]. Khoshgoftaar et al. [23] used the neural network for predicting the number of faults and shown static reliability modelling. The trained two neural networks; one with the complete set of principal components and one with the position of components selected by multiple regression model selection. Comparison of these models showed a better accepting of neural network software quality models. Ho et al. [24] investigated a modified Elman recurrent neural network in modelling and predicting software failures and then performed a comprehensive study of connectionist models and their appropriately to software reliability prediction. Tian and Noore [25] proposed an on-line adaptive software reliability prediction model using evolutionary association approach based on multiple-delayed-input single-output architecture, which showed better performance with esteem to next-step predictability compared to existing NN model.

Yu Shen Su et al. [26] proposed a model that uses the neural network approach to build a dynamic weighted combinational model. They evaluated the performances of the neural network models with some conventional SRGMs from three aspects: goodness of fit, forecast ability for short-term prediction and long-term prediction. Result shows that purposed model has more accurateness with both goodness of fit and the prediction ability compared to existing conventional models. Viswanath [27] proposed two models such as neural network based exponential training and neural network based logarithmic encoding for prediction of cumulative numeral of failures in software. He used execution time as the input and applied on four data sets. Result showed that its result is better that other statistical model. Su and Huang [28] discussed artificial neural-network-based techniques for software reliability estimation and modelling then further use of the neural network approach to make a dynamic weighted combinational model. The results obtained from the experiments show that the proposed model has a moderately accurate prediction capability. Sandeep Kumar Jain et al. [29] proposed a method to assess the reliability of the software consisting of system by using different neural network architectures, then estimate the faults prediction behaviour in the set of components over a cumulative execution time interval moreover this the prediction of faults is estimated for the complete software.

4.2 Fuzzy Logic System

Fuzzy Logic approach has been successfully used to solve selections of problems in modeling and recognition of nonlinear systems developed by Prof.LotfiA.Zadeh[21], Fuzzy Inference System is based on the fuzzy set theory, fuzzy if then rules, and fuzzy reasoning and is a popular computing framework. The basic idea is to model a method which can develop its own software reliability estimation model based on the history of earlier period failures of software_s. Thus such a model shall eliminate the need of any outside variables or any other dependencies to estimate software reliability [20]. A fuzzy model is a mapping between linguistic terms, attached to variables. Hence the input to and output from a fuzzy model can be either numerical or linguistic [30]. Cai et al. [31] discussed the development of fuzzy software reliability models in place of probabilistic software reliability models (PSRMs). It was based on the proof that software reliability is fuzzy in nature.

Khalafkhatatneh [32] model focused on particular dataset behaviour in predicting reliability. Focusing on particular dataset behaviour is performed to develop a precise model since the recent work focused on developing a model which can be more accurate. Reformat [33] proposed a multi technique knowledge extraction and comprehensive Meta model prediction system in the area of remedial maintenance of software. The system was based on evidence theory and a number of fuzzy-based models. Aljahdali et al. [34] investigated fuzzy logic on SRGM to estimate the expected software faults during testing process. This purposed model gives an prominent performance modelling capabilities. S. Chatterjee et al. [35] two fuzzy time series based software reliability reproduction have been proposed. The first one predicts the time between failures of software and the second one predicts the number of faults present in software. The purposed models are flexible, assumption free and very simple in computation.

4.2.1 Evaluation (Fitness) Criteria

The above analysis criteria used for evaluating the selected model and which had been very much highly used by the people of the software industry especially by the software engineering area are:

4.2.2 Root Mean Square Error (RMSE)

The Root Mean Square Error (RMSE)[14], which can be called as the root mean square deviation, RMSD is an evaluation tool and highly used measure for calculating the difference between the observed values and actual values as a result of the experiment based on some observations of the experiment. All the different stages of the experiments (residuals) are combined together to produce a single measure with the assist of this evaluation criteria. Generally, the RMSE of a model prediction with respect to the estimated variable Xmodel is distinct as the square root of the mean squared error and it is shown in the equation 2 given below:

$$'E = \sqrt{\sum_{i=1}^{n} \frac{(X\,obs,i - Xmodel,i)2}{n}} \tag{2}$$

Where Xobsis observed values/faults and Xmodel is model values at running time i. The RMSE values can be used to differentiate the performance of the different models and to validate the models. Also it will compare the performance of the model with other predictive models as a part of the comparative study. In most of the cases, we can use the Root Mean Square Error (RMSE) for validating the suitability of the selected model to assess the software reliability effectively.

4.2.3. Normalized Root Mean Square Error (NRMSE)

Normalized Root Mean Square Error (NRMSE)[14], Non Dimensional forms of the RMSE are helpful because often one wants to compare RMSE with different units, shown in the equation 3. There are two approaches: normalize the RMSE to the range of the observed data, or normalize to the mean of the observed data.

$$NRMSE = \frac{RMSE}{Max - Xobs,Min}, \qquad NRMSE = \tag{3}$$

4.3 Neuro Fuzzy Techniques

A neuro-fuzzy framework is a fuzzy framework that uses attacking in calculation got from or propelled by neural system hypothesis to focus its limitations (fuzzy sets and fuzzy rules) by preparing information tests. Neuro-fuzzy was proposed by J. S. R. Jang. Neuro Fuzzy System synergizes these two procedures by consolidating the human-like thinking style of fuzzy frameworks with the learning and connection is structure of neural systems. The several of Neuro fuzzy Systems are

- Mamdani based neuro Fuzzy inference system.
- Takagi Sugeno fuzzy inference system

A neural fuzzy framework makes utilization of neural systems to give fuzzy frameworks a sort of programmed tuning strategy, by not influencing their functionalities. There is a preparation process, where a neural system modifies its weights to minimize the mean square error between the yield of the system and the desired output. Mamdani fuzzy inference framework varies from the other variation takagisugeno fuzzy inference framework and has advantage of having likeness to the human subjective framework. KirtiTyagi et al. [41] proposed a model for estimating CBSS reliability, known as an adaptive neuro fuzzy inference system (ANFIS) that is based on two basic elements of soft computing, neural network and fuzzy logic. ANFIS model gives a more precise measure of reliability than the FIS model, as it reduces error from 11.74%, in case of FIS model, to 6.66% in ANFIS. The experimental model is compared with existing takagi sugeno based fuzzy model. The evaluation criteria of the proposed model are implemented in MATLAB. The Root Mean Square Error (RMSE) is computed for different membership functions. These computations are carried out for 3 given data set that is system code 1, 2 and 17 shows that the RMSE values of proposed model is less than the existing model. Therefore, Mamdani based fuzzy model is better in predicting software reliability. The following table 1 and 2 shows the data set, RSME comparisons of Takagi sugeno fuzzy and Mamdani fuzzy model

Table1: Result analysis of Takagi sugeno fuzzy model

Data set name	Model	RMSE
Real time command and control(1)	Takagi sugeno fuzzy	1.152
Real time command and control(2)	Takagi sugeno fuzzy	2.241
Retail	Takagi sugeno fuzzy	1.123

Table 2: Result analysis of Mamdani fuzzy model

Data set name	Model	RMSE
Real time command and control(1)	Mamdani based fuzzy	1.365
Real time command and control(2)	Mamdani based fuzzy	0.865
Retail	Mamdani based fuzzy	0.470

Neuro fuzzy techniques defined as the combinations of neural networks and fuzzy logic. It is used to access the data faster with accuracy, compatibility and performance. The neuro fuzzy technique is a backbone of soft computing. Soft computing techniques defined as the combination of neuro-fuzzy computing and derivative free optimization. The characteristics of soft computing are human expertise, biologically inspired computing models, new optimization techniques and numerical computation.

4.4 Genetic Algorithm (GA)

Genetic Algorithms (GAs) was developed by Prof. John Holland and his students at the University of Michigan during the 1960s and 1970s. Genetic algorithm can be used by characterize a solution to this problem as a genome (orchromosome) [36]. The genetic algorithm then produces a population of solutions and applies genetic operators such as transformation and crossover to develop the solutions in order to find the best one. The three most important aspects of using genetic algorithms are:

(1) Develop the objective function,

(2) Develop and implementation of the genetic representation, and

(3) Definition and implementation of the genetic operators.

Once these three have been defined, the generic genetic algorithm should work moderately well. Beyond that can try many different variations to improve performance, find multiple optimal (species - if they exist), or parallelize the algorithms [37].Genetic algorithms are machine learning and optimization systems, much like neural networks. However, genetic algorithms do not appear to suffer from local minima as poorly as neural networks do. Genetic algorithms are founds on the model of evolution, in which a population develops towards overall fitness, even though individuals expire. Evolution dictates that superior individuals have a enhanced chance of reproducing than inferior individuals, and thus are more possible to pass their superior traits on to the next generation. This continued existence of the fittest, criterion was first converted to an optimization algorithm by Holland in 1975, and is today a major optimization technique for complex, nonlinear problems.

5. SUMMARIZATION

In the past few years a number of software reliability models have been analysed, designed and evaluated. Soft computing plays an important role in the recent advancements in the software reliability growth models. Today these models included the application of different soft computing techniques such as Neural Network (NN), Fuzzy Logic, Neuro Fuzzy Logic (NFL), Genetic Algorithms (GA), thus the survey; we notice that the four soft computing techniques are used in different shapes with these models. We observed that Neural Network approach is more liked by the researchers in software reliability models. Fuzzy logic provides more accuracy than other soft computing techniques. This table data is important in case of comparison and selection of soft computing technique in terms of modelling capabilities.

6. DIFFERENT SOFT COMPUTING COMPARISON TECHNIQUE IN TERMS OF SOFTWARE RELIABILITY MODELS

Comparative analysis is very useful in case of optimal selection, user can view all possible choices on a single plate form and can select the best suited as the requirement. In Table 3, we have compared different

S.No	Tech. Used	Explain outputs	Suitability for small data sets	Suitability for large data sets	Can be redesigned for new data set	Reasoning process is visible	Applicability for complex models	Either known facts considered	Evaluation of processing time	Maintain -ability
1	Neural Network	N	N	Y	N	N	Y	Partially	Limited	Y
2	Fuzzy Logic System	Y	Y	Y	Y	Y	Y	Y	High	N
3	Neuro Fuzzy	Y	N	Y	Y	N	Y	Y	High	Y
4	Genetic Algorithm	Partially	Partially	N	N	N	Partially	N	Low	Y

Table 3 : Comparison of Soft Computing Techniques in terms of Modelling Capabilities.

parameters of soft computing techniques in terms of software reliability model such as original data sets, re-adjustments for new data set, process visibility, Transaction time, facts and outputs, Maintainability, product quality, etc. This comparison has outlined some parameters of modelling capabilities. From this table we observed that all the techniques explain its outputs and are applicable for complex models except genetic algorithm. Comparison revealed that only fuzzy Logic can be widely used for all the modelling capabilities. Various soft computing modelling such, Stimulated Annealing, Tabu Search, Antcolony and Cuckoo Search can also be used but suitable to only small data set capabilities. The rapid growth of soft computing techniques suggests that the impact of these algorithms will be used increasingly for software reliability models in the coming years. This table will help computer scientist who are keen to contribute their works to the field of software reliability.

6.1 Metrics for reliability prediction Today corporate company saves software metrics as sign of a maturing software development process. Industrialized software engineering data sets, particularly those related to system failures, are historically inflexible to obtain across a range of organizations. Novel methods are used to improve reliability prediction are actively being researched. For example, by evolving rich information from metrics data using a sound statistical and probability foundation, Moreover, conventional reliability models can be enhanced to incorporate some testing completeness or efficiency metrics, such as code coverage, as well as their traditional testing-time based metrics. The key idea is that failure detection related to the time that the software is under testing, but also what fraction of the code has been executed by the testing.

7. CONCLUSION

Software failure prediction is often impossible in real world. However we could observe this through character of work in process. In this paper we have analysed the various systematic existing approach to prove the software reliability using soft computing models. We emphasize the some soft computing model to prove the software reliability. The computing task would serve as a reference to both old and new includes neural network, Fuzzy logic, neural fuzzy and genetic algorithm.
From our result analysis, we observed that handling of software datasets in a variety of applications models are proved that the computation results using fuzzy logic are much better than the other computing models. Finally this model to support our understanding of current trends and guide the research flows. Also the result show that prediction of failure-free software would possible in software development. This research provides a best indication of prediction strength of developed fuzzy model for accessing the software reliability.

7.1Future work

Predict Failure-Free Software Access to improve the software reliability can extend this work to the other soft computing techniques to give better estimation of the software reliability at different stages of Software Development process. Prediction and sharpness of the Fuzzy Rule Generation for the Fuzzy Inference System can further be improved with decision making systems.

8. REFERENCES

[1] B W. Boehm, Software engineering economics, Englewood Cliffs, NJ: Prentice-Hall, 1981, pp. 50-100.

[2] S Amasaki, O. Mizuno, T. Kikuno, and Y. Takagi, ―A Bayesian Belief Network for Predicting Residual Faults in Software Products, Proceedings of 14th International Symposium on Software Reliability Engineering (ISSRE2003), November 2003.

[3] Software Engineering-Product Quality—Part 1: Quality Model, International Organization for Standardization, 2001, ISO/IEC 9126-1:2001

[4] B W. Boehm, J. R. Brown and M. L. Lipow, Quantitative Evaluation of Software Quality Proceedings of the 2nd International Conference on Software Engineering, San Francisco, CA, USA, October,1976, pp.592-605.

[5] J A. McCall, P. K. Richards, and G. F. Walters, Factors in Software Quality, Vol.I, II, and III, US Rome Air Development Center Reports - NTIS AD/A-049 014, NTIS AD/A-049 015 and NTIS AD/A-049 016, U. S. Department of Commerce, 1977.

[6] R G. Dromey, ― A model for software product quality, IEEE Transactions on Software Engineering, Vol.21, No.2, February, 1995, pp.146-162.

[7] J W. Park R, W. Goethert, ―Software cost and schedule estimating: A process improvement initiative, tech. report, 1994.

[8] K.M Atsumoto and K. Inoue, ―Experimental evaluation of software reliability growth models, in Proceeding of the IEEE of FTCS18, 1988, pp.148-153.

[9] Y K. Malaya, N. Karunanithi, and P. Verman, ― predictability measures for software reliability models in proceeding of the 14 the IEEE inter. Conf. Computer Software Applications, 1993, pp. 7-12.

[10] R Laddad, ―Aspect Oriented Programming will improve Quality, published by IEEE Computer Society 0740-745 ,2003.

[11] IEEE Standard Glossary of Software Engineering terminology, IEEE Std 610.12-1990.

[12] Chu, Y., &Xu, S. Exploration of complexity in software reliability. Tsinghua Science & Technology, 12, 266-269 (2007).

[13] K Y., Cai, C.Y., Wen, and M.L., Zhang, A critical review on software reliability modelling. Reliability Engineering and System Safety 32 (3), 1991, 357–371.

[14] N, Karunanithi, D., Whitley, and Y., Maliya, Prediction of software reliability using connectionist models. IEEE Transactions on Software Engineering 18, 1992, 563–574.

[15] Zadeh LA Fuzzy Algorithms. Information and Control 12: 94-102, (1968).

[16] Zadeh LA Fuzzy Sets. Information and Control 8: 338-353 (1965).

[17] Kuo W, Prasad VR An annotated Overview of System Reliability Optimization. IEEE Transaction on Reliability 49: 176-186(2000).

[18] Streichert F - Introduction to Evolutionary Algorithms. Frankfurt Math Finance Workshop, University of Tubingen, Germany(2002).

[19] Boccaletti S The control of chaos: theory and applications. Elsevier Science (2000) 329: 103-197.

[20] Quah, T. and Thwin, M.―Application of Neural Networks for Predicting Software Development Faults(2000) Using Object-Oriented Metrics Proceedings of Ninth International Conference on Neural Information Processing(ICONIP_02), vol. 5, pp. 2312-2316.

[21] Aggarwal G, Gupta VK Neural Network Approach to Measure Reliability of Software Modules: A Review. International Journal of Advances in Engineering Sciences (2013).
[22] Khoshgoftaar TM, Szabo RM, Guasti PJ Exploring the Behaviour of Neural-network Software Quality Models. Software Engg J (1995) 10: 89-96.
[23] Ho SL, Xie M, Goh TN A Study of the Connectionist Models for Software Reliability prediction. Computers and Mathematics with Applications (2003) 46: 1037- 1045.
[24] Tian L, Noore an On-line prediction of software reliability using an evolutionary connectionist model. Journal of Systems and Software (2005) 77: 173-180.
[25] Su SY, Huang CY, Chen YS, Chen JX An Artificial Neural-Network- Based Approach to Software Reliability Assessment. TENCON, IEEE Region (2005) 10:1-6.
[26] Viswanath SPK Software Reliability Prediction using Neural Networks.PhD. Thesis, Indian Institute of Technology Kharagpur(2006).
[27] Su YS, Huang CY Neural-network-based approaches for software reliability estimation using dynamic weighted combinational models. Journal of Systems and Software (2006) 80: 606–615.
[28] Jain SK, Singh MP Estimation for Faults Prediction from Component Based Software Design using Feed Forward Neural Networks. IJARCCE(2013).
[29] Aljahdali S, Debnath NC Improved Software Reliability Prediction through Fuzzy Logic Modeling. IASSE(2004).
[30] Cai KY, Wen CY, Zhang ML A critical review on software reliability modeling. Reliability Engineering and System Safety (1991) 32: 357-371.
[31] Khatatneh K, Mustafa T Software Reliability Modeling Using Soft Computing Technique. European Journal of Scientific Research (2009) 26: 147-152.
[32] Reformat M A fuzzy-based multi-model system for reasoning about the number of software defects. International Journal of Intelligent Systems (2005) 20: 10931115.
[33] Aljahdali S Development of Software Reliability Growth Models for Industrial Applications Using Fuzzy Logic. Journal of Computer Science (2011) 7:1574-1580.
[34] Chatterjee S, Nigam S, Singh JB, Upadhyaya LN Application of FuzzyTime Series in Prediction of Time Between Failures & Faults in Software Reliability Assessment. Fuzzy Inf Eng 3: 293-309.
[35] L A. Zadeh, ―Fuzzy Sets, Information and Control 1965.
[36] Adnan, W,A, Mashkuri, H,Y, ―An Integrated Neural Fuzzy System of Software Reliability Prediction, IEEE Software.