Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2020)
Vol.02/ No. 03
Pages: 126-135
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2020.3.002

Reliable Automated Software Testing Through Hybrid Optimization Algorithm

Dr. Subarna Shakya,
Professor,
Department of Electronics and Computer Engineering,
Central Campus, Institute of Engineering, Pulchowk,
Tribhuvan University,Pulchowk,
Lalitpur Nepal.
Email: drss@ioe.edu.np.

Dr. S. Smys,
Professor,
Department of CSE,
RVS Technical Campus,
Coimbatore, India.
Email id: smys375@gmail.com

Abstract: - Software testing is an important process in product development of software companies to ensure the product quality. The developed application must satisfy the customer needs and meets the industrial standards without any compromise. Thus, verification of products through manual test engineer and validate the product once it meets all the necessary requirements. The issues in manual testing is its high computation and analysis time, accuracy and reliability. In order to reduce the issues in manual testing automatic testing is introduced. Though it is also an application which requires parameters to test the given product. Efficient tuning of the product could be achieved through automatic testing. This proposed research work provides an optimized automatic software testing model through differential evolution and ant colony optimization as a hybrid model to achieve improved accuracy and reliability in software testing. Conventional models like artificial neural network and particle swarm optimization are compared with proposed model to validate the reliability of proposed model.

Keywords: - Software testing, Differential Evolution, Ant Colony Optimization, Artificial Neural Network, Particle Swarm Optimization.

1. Introduction

Software testing is the process used in software development companies to check the application quality to bring out an error free product to clients. The requirements of the clients and their suggestions are discussed to develop the product. Through proper validation and verification, the products must be delivered to the customers which reduces the pressure to the companies after product release. Growth reliability models in software testing is the current trend which provides better solutions to the needs of industries. Efficient automation provides better relation between verification and validation where the conventional testing process doesn't provide any such relationship as the verification and validation are performed by different persons. Real time applications require these automation process as a unified testing which mainly focuses on automation along with manual testing process. manual testing is a process which is used to check the system design for identifying the errors based on test cases. This process is repetitive and it requires a testing developer with adequate knowledge. In order to reduce the issues in manual testing, automated testing environment is developed with various levels of abstraction under various domains with wide range of diversity policies. The ultimate aim of automated testing by industries is to reduce the computation time and cost. Figure 1 gives an illustrative representation of software test cycle in detail.

ISSN: 2582-337X
Submitted: 20.06.2020
Accepted: 29.07.2020
Published: 04.08.2020

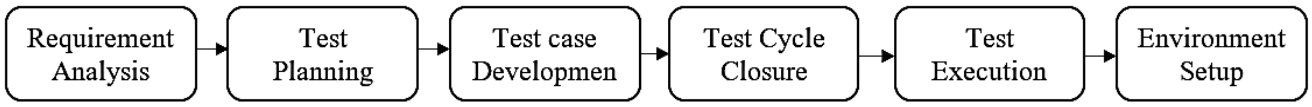

Fig.1 Software test cycle

The quality of software is defined based on its ability to run in different environments and the produce uniqueness. Based upon the user needs the product is developed in software development domain through software engineers. Once the application is developed based on the below parameters the product is verified and validated in testing environment such as

- Reliability
- Portability
- Usability
- Flexibility
- Testability
- Efficiency

In which reliability of the product defines the functions and their performance. Based on user requirements the product must be developed and the reliability level should be satisfactory to the developer and user. Testability describes about the product evaluation and the usability provides information about the user friendly and environment friendly characteristics of the product. Flexibility provides information about product adaptation for further upgradation. Efficiency depends on all the above features which describes about the product performance.

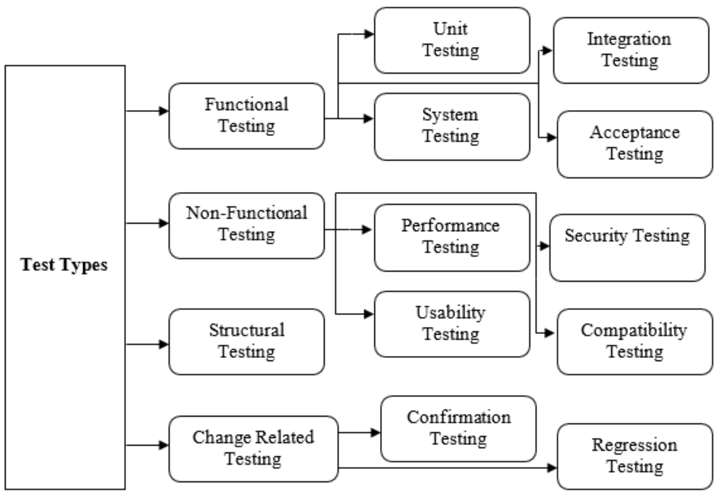

Fig.2 Software Testing Types

Software testing helps to improve the product design based on standard verification over static and dynamic environment. Using formal procedures and algorithms, verification is performed in static environment where in case of dynamic, the verification is performed at any stage through test cases. Compared to static, dynamic is much efficient and identifies the errors in product design. Figure 2 depicts the various types of testing process used in software test engineering. Early stage testing reduces the issues in the product by identifying the bugs and reduces the product failure upon implementation. Simple test and code process are referred in software testing which reduces the errors in each stage of the application. In case of verification and validation process, the right choice of product is analyzed in verification process where as the product characteristics are analyzed in validation process. It allows the user or the customer to check the developed application to satisfy the needs by correcting the issues.

The need of software testing is to reduce the bugs in a minimum cost as most of the testing process consumes almost 45-60% of the development cost. Based on the requirement, Software testing is broadly classified into manual testing and automatic testing. The automation process cycle starts from decision which describes the procedure for product evaluation. Followed by decision, acquiring tool is used to define the testing tool such as

ISSN: 2582-337X
Submitted: 20.06.2020
Accepted: 29.07.2020
Published: 04.08.2020

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2020)
Vol.02/ No. 03
Pages: 126-135
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2020.3.002

automation studio, Experitest, Testcomplete, etc., once the suitable tool is selected the entire script is included in the tool and it creates a design plan to develop the testing procedure. Execution is the last step in automation which executes the scripts based on the design and assess the results. Based on the script knowledge of the developers, software testing is further classified into black box testing and white box testing. If the testing is performed in the script itself then it is white box testing where the testing engineer must have prior knowledge about the scripts. Various factors are included in white box testing to examine the internal structure of the data from test cases. Figure 3 gives a classification in white box testing which is widely used in industries.

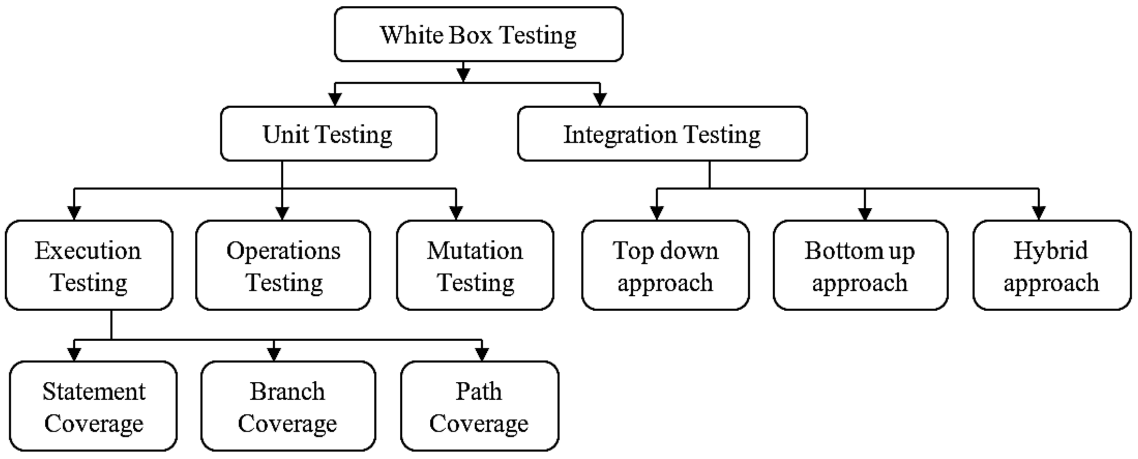

Fig.3 Classification in White Box Testing

Upon various classification in white box testing, execution testing and its types are widely used. In particular branch coverage is familiar and it provides a complete test results for every branch which is used to evaluate the developed application efficiently. The entire path is used to test in path coverage type and it is suitable for small scale applications. in case of statement coverage, statements used in the script are executed and produces faire results. The limitations of white box testing are its static coding model which increases the cost function, provides formal evaluation and doesn't consider the specifications of the developer and it is not suitable for all the testing engineers.

On contrary to the white box testing, black box testing doesn't require adequate knowledge about the testing script and it could be performed by anyone without any script knowledge. It performs testing in an external application using third party software such as Selenium, UI test builder, applitools eyes, etc., Black Box concentrates over different values of boxes and provides corresponding outputs. The correlation between the black box interface and engineer's expectation is the main process in testing. Figure 4 gives an illustration of processes involved in black box testing.

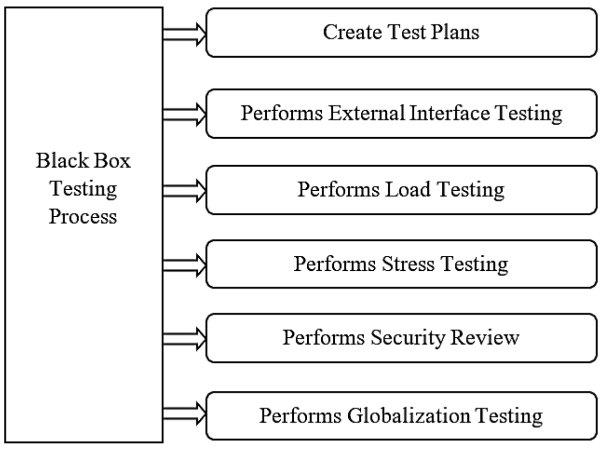
Black Box Testing Process
Create Test Plans
Performs External Interface Testing
Performs Load Testing
Performs Stress Testing
Performs Security Review
Performs Globalization Testing

Fig.4 Black Box Testing

ISSN: 2582-337X
Submitted: 20.06.2020
Accepted: 29.07.2020
Published: 04.08.2020

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2020)
Vol.02/ No. 03
Pages: 126-135
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2020.3.002

Software reliability engineering deals with reliability of the product which is widely used in leading industries to obtain solution for the user queries through applications. Also, it deals with quantitative measures of the product in terms of dependability and quality. By measuring, predicting and managing the product reliability is increases which satisfies the customer requirements. Almost 90% of the applications are planned and developed through this control process. The major challenges in reliability growth models are, every tester need to include an operational profile before testing the product and it must specify the operations, reduced number of failures defines the reliable product and finally, fault assumptions in software reliability models must be independent in nature. Since it is essential to define the test efforts which mentions the testing time and completion time. Software reliability provides solution to the engineers by predicting the application needs from starting to implementation. Generally, software engineering includes process such as fault prediction, detection and removal of errors, maximizing the reliability through effective measurements. More over software reliability performs product evaluation in terms of quantity and quality analysis, test schedule and its status and reliability measurements.

2. Related works

In order to obtain the issues in existing software testing process, a vast survey has been made based on reliability, classification accuracy. Various models are evolved in software testing, in this machine learning gains more attention due to its computation characteristics. Ajmer Singh.*et.al* [1] summarized various machine learning algorithms which is used for software fault prediction. Based on the object-oriented metrics such as learning function, features and efficiency, machine learning algorithms are compared in the proposed survey work. Research work provides an intense idea about machine learning models in software engineering. Alireza Haghighatkhah.*et.al* [2] also provides a systematic study of automation in software engineering. This literature work covers the automation from conventional methods to recent machine learning techniques in software engineering. Amir Elmishali. *et.al* [3] reported the issues in conventional software testing process through artificial intelligence model. The proposed model attains better reliability compared to conventional models. however, there is some limitations in the proposed model as it lags to evaluate the bugs for complex scripts.

Diana-Lucia Miholca. *et.al* [4] reported the issues in software defect prediction model through hybrid machine learning approach. The proposed model combines the relational association rules with ANN to obtain fault prediction model which attains better performance in terms of efficiency and reliability. Compared to defect prediction in software engineering, defect prevention is simple and easy to compute. Fuqun Huang.*et.al* [5] proposed such defect prevention mechanism through human error theories in software testing process. The research work defines the rules based on the common errors which could be performed by humans and the prevention mechanism are employed to obtain increases efficiency of the product. Helson. *et.al* [6] proposed a multi objective algorithm to improve the performance of software product. Parameters such as reliability, efficiency and classification accuracy are considered as multi objectives and all the parameter performance are improved in the proposed model compared to conventional models.

Fault tree analysis-based software testing is reported in Hua Wei Li. *et.al* [7] research model. compared to conventional models this fault tree analysis is simple and efficient which has better computation time. The performance of proposed model is better but when it is compared to machine learning models the fault tree models attains lesser performance. Jinyong Wang. *et.al* [8] reported the issues in software reliability models through RNN encoder and decoder. The proposed model has better performance in reliability for complex data sets. Kaliraj, Bharathi [9] proposed a reliability analysis models through component-based software system. The faults in the product are analyzed by dividing the entire script into individual components and then bugs are identified in the proposed model. This process provides more accurate results but the computation time is increased due to the dividing process. Kapil Juneja [10] proposed a neuro fuzzy based fault prediction model. The proposed research model has inter-version and inter project evaluation models which highly improves the testing efficiency. Peng Xiao. *et.al* [11] reported the issues in software testing through feedback-based prediction mechanism. The faults are experience and summarized by evaluating the product features to certain group of people. Based on their feedback the bugs are identified and corrected which increases the fault detection accuracy. But the proposed model consumes more time to summarize and analyze the feedbacks also the feedback may contain uncertainties which may produce adverse effect at some times. Various hybrid algorithms are evolved in software testing. Wasiur Rhmann. *et.al* [12] discussed various models in software fault prediction and software fault localization is discusses in Xiao-Yi Zhang. *et.al* [13] research model. Yuanxun Shao. *et.al* [14] presented a rule mining-based software defect prediction which has better performance in terms of prediction accuracy and efficiency. From the above survey it is observed that, still the research towards increased reliability is in progress. Since reliable model

ISSN: 2582-337X
Submitted: 20.06.2020
Accepted: 29.07.2020
Published: 04.08.2020

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2020)
Vol.02/ No. 03
Pages: 126-135
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2020.3.002

defines the quality of the application which improves the software performance. Based on this motivation, the research work proposed a hybrid model to improve the reliability in software automation process.

3. Proposed Optimization Model

This section provides the mathematical formulation of ant colony optimization, a nature inspired optimization formulated based on the ant food searching nature. The behavior of ant in searching food and identifies the food are considered and formulated as optimized path. This optimal path selection strategy is related to the proposed automation process. For this, initially the conventional ant colony optimization is analyzed and then proposed hybrid algorithm is implemented for the proposed research work. The nature of ant food searching behavior starts from the random fashion of search process and once the food is identified, the colony will observe the path through its left pheromones and reach the source. This process provides an optimal path for the ants to bring food from source to the colony. Figure 5 gives an illustration of finding optimal path in ant colony optimization.

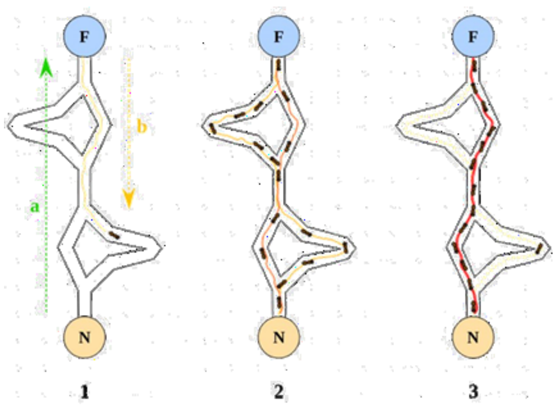

Fig.5 Ant Colony Optimization

The pheromone intensity is increased by continues travel of ants in the path, once the pheromone is evaporated the path is diminished so that the ants again starts to search another optimal path. The path selection is an attractive strategy in ant colony optimization since the evaporation nature of pheromone is maintained by the ants to obtain optimal solution which is related to various real time applications to obtain optimal solution. Conventional ant colony optimization considers the convergence parameter, data size and coefficients to measure the nature of ant. It considers the individual movement from a particular location to other location for searching food. The location change is mentioned as $\overrightarrow{h_\iota}$ and based on random distribution the stability is derived as

$$\overrightarrow{h_{j,\iota}} = \vec{h}_{i,j} + Real * Rand\ int[-1,1] \tag{1}$$

Based on the stability of the new position and the present position the best position is updated. If the stability of the present position is better than previous position, then it is updated as new position based on the equation given below

$$\overrightarrow{h_{j,\iota}}(t+1) = \vec{h}_{i,j}(t) + \left(\overrightarrow{h_{j,\iota}}(t) - \vec{h}_{i,j}(t)\right) * Rand\ int[0,1] \tag{2}$$

The process is repeated with frequent update of positions and the details between the present position and the new position is measured. The behavior of forward movement is given as

$$\overrightarrow{\mathrm{h}_{\mathrm{j},\iota}}(\mathrm{t}+1) = \vec{\mathrm{h}}_{\mathrm{i,j}}(\mathrm{t}) + \left(\frac{\overrightarrow{\mathrm{h}_{\mathrm{j},\iota}}(\mathrm{t}) - \vec{\mathrm{h}}_{\mathrm{i,j}}(\mathrm{t})}{\varphi_{\mathrm{i,t}}}\right) * \mathrm{Rand\ int}[0,1] \tag{3}$$

If the position of forward nature of ants obtains the best position which increases the convergence rate. Based on this interaction, the distance between the neighbors are identified based on Euclidian distance. To improve the convergence rate, the proposed model is designed with differential evaluation which helps to obtain

ISSN: 2582-337X
Submitted: 20.06.2020
Accepted: 29.07.2020
Published: 04.08.2020

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2020)
Vol.02/ No. 03
Pages: 126-135
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2020.3.002

the nature of distribution. Instead of random distribution function the differential evaluation obtains optimal path in software testing as reliability model. The minimum function is given as

$$\rho(z_{1,}z_2, z_{3,} \dots z_n) = \sum_{i=1}^{n}\left(\iint_{x^3}^{\alpha} \|Z_i - l\| * \varphi(i)/2\right) \tag{4}$$

The obtained ant colony optimization provides the connection between the vertex and provides direct relationship between the adjacent vertices to identify the errors in the script environment. This probability helps to assign the values for each transition and concludes the vertex into feasible or non-feasible models. Based on the pheromone, the transition of each elements is considered for feasible transition and updates the position and improves the decision-making process. The transition visibility of ant position is indicated through heuristic functions. The visited status provides information about existing ant travel states. Generally, these values are considered as zero for visited status and one for existing status. At last, the probability values provide the transition feasibility, heuristic information and pheromone values which are helps to obtain converged optimization algorithm with increases reliability in bug finding in developed product. Figure 6 depicts the process flow of proposed optimization model.

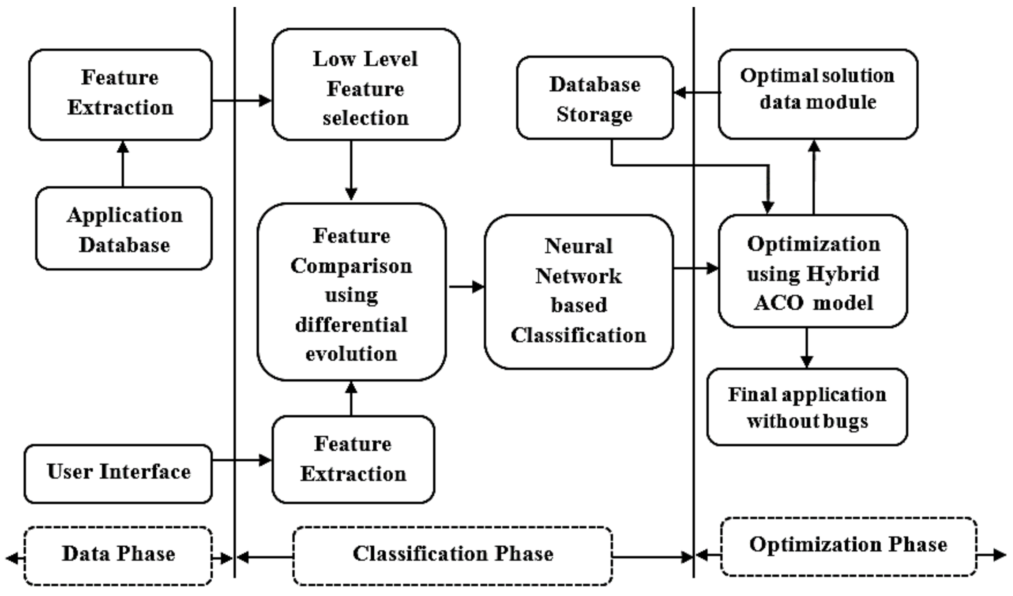
Feature Extraction
Application Database
User Interface
Low Level Feature selection
Feature Comparison using differential evolution
Feature Extraction
Neural Network based Classification
Database Storage
Optimal solution data module
Optimization using Hybrid ACO model
Final application without bugs
Data Phase
Classification Phase
Optimization Phase

Fig.6 Proposed Optimization model

The pseudocode for the proposed hybrid optimization model is summarized as follows.

```
------------------------------------------------------------------------------------------------
Initialize ant population size,
Based on continues trail generate random variables
Compare the strength and its stability factor
Select optimal strategy policy
Check and change the next preferred location
The parameters are $\varphi_{2,}\varphi_{3,\dots}\varphi_n$
        for $h_l = Rand\ int(\varphi_n l)$
                Select the random points $p^n, n = 1,2, \dots . h_n$
                        for n= 1 to $h_n$
                                computing the integer values $h_{best} = int(\varphi_4 h)$
                        if $h_{best} > 0$ then
                for $i = 1$ to $h_{best}$
move and create a suitable connection the set $Z_\sigma = z(x_\sigma^i, y_\sigma^j)$
If $Z_\sigma < Z_n$ then replace the current position into new position based on the function value.
------------------------------------------------------------------------------------------------
```

ISSN: 2582-337X
Submitted: 20.06.2020
Accepted: 29.07.2020
Published: 04.08.2020

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2020)
Vol.02/ No. 03
Pages: 126-135
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2020.3.002

4. Result and Discussion

The experimentation of proposed hybrid model is performed in MatLab 14.1 installed on a 3.5GhZ i3 processor with 4GB of RAM. To increase the complexity of the experimentation process, we have used three different data sets with different attributes. To validate the superior performance of proposed model conventional artificial neural network and particle swarm optimization models are used to compare the proposed model results. Table 1 gives the details of data set used for the experimentation process.

Table 1 Details of Dataset used in the experimentation

S. No	Dataset	Number of Records	Number of Attributes
1	PC1	1077	38
2	KC2	522	95
3	MC2	600	40

The confusion matrix for the proposed design is given in table 2. In this, true positive provides the details of predicted defects, false positive describes the non-defective items, false negative describes the defects quantity as non-negative values and true negative provides the details of defective classes.

Table 2 Confusion Matrix

	Defective	**Non-Defective**
Defective	True Positive	False Negative
Non-Defective	False Positive	True Negative

Based on the parameters such as precision, f-measure, mean, sensitivity and specificity the performance of all the three models are evaluated for all the three data sets along with the confusion matrix given in table 2. The mathematical formulations for parameters are given as

$$precision = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (5)$$

$$f - measure = \frac{2 \times precision \times Recall}{Precision + Recall} \quad (6)$$

$$mean = \sqrt{Specificity \times Sensitivity} \quad (7)$$

$$specificity = \frac{True\ Negative}{True\ Negative + False\ Positive} \quad (8)$$

$$sensitivity = \frac{True\ Positive}{True\ Positive + False\ Negative} = Recall \quad (9)$$

Table 3 provides the details of obtained parameters for all the three models for the given three data sets. It is observed from the results, the performance of proposed hybrid ACO model is much better than other models.

ISSN: 2582-337X
Submitted: 20.06.2020
Accepted: 29.07.2020
Published: 04.08.2020

Journal of Ubiquitous Computing and Communication Technologies (UCCT) (2020)
Vol.02/ No. 03
Pages: 126-135
https://www.irojournals.com/jucct/
DOI: https://doi.org/10.36548/jucct.2020.3.002

Table 3 Observed parameters for three data sets

S. No	Algorithms	Data Set 1			Data Set 2			Data Set 3		
		Mean	F-Measure	Precision	Mean	F-Measure	Precision	Mean	F-Measure	Precision
1	ANN	0.92	0.93	0.94	0.81	0.88	0.84	0.84	0.83	0.85
2	PSO	0.97	0.95	0.96	0.84	0.91	0.86	0.86	0.82	0.88
3	Hybrid ACO	0.98	0.99	0.99	0.95	0.95	0.93	0.90	0.89	0.90

The performance comparison in terms of sensitivity and specificity is depicted in figure 7. It is observed from the figure, that the proposed model attains better sensitivity and specificity values than artificial neural network and particle swarm optimization model.

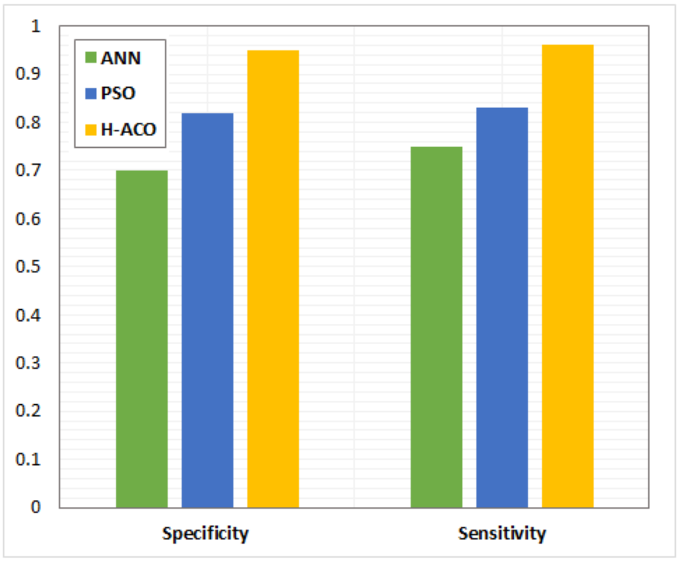

Fig.7 Comparison of specificity and sensitivity

The accuracy comparison between ANN, PSO and H-ACO is depicted in figure 8. It is observed that proposed model attains better accuracy compared to other models. The proposed model hybrid ACO model attains accuracy range of rate of 96.2%, but particle swarm optimization attains an average of 95.5% and artificial neural network obtains an accuracy of 92% which is 4% lesser than the proposed model.

ISSN: 2582-337X
Submitted: 20.06.2020
Accepted: 29.07.2020
Published: 04.08.2020

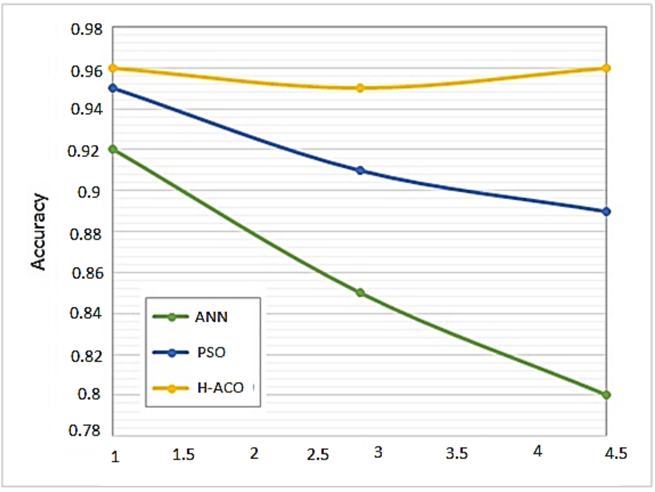

Fig.8 Accuracy Comparison

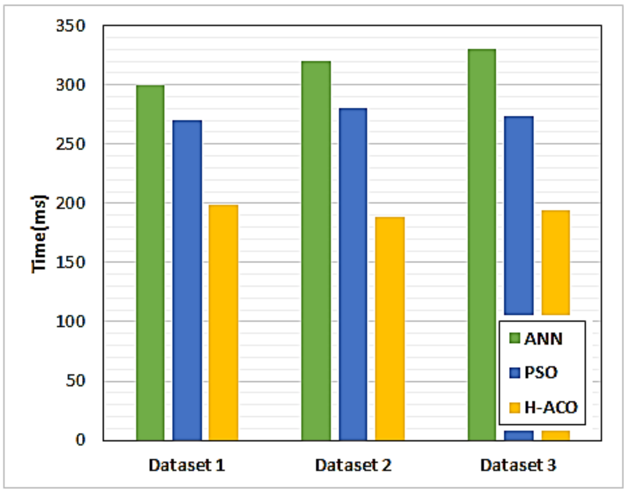

Fig. 9 Computation Time Comparison

Analysis of computation time is depicted in figure 9. For all the three data sets, all the three models are experimented. It is observed that the proposed model computes with in minimum time compared to other models. performance of ANN lags due to its poor memory capacity and PSO consumes more time due to its convergence parameters.

5. Conclusion

This research work proposed a hybrid ant colony optimization model for reliable software automation to achieve an efficient product development in software companies. Differential evaluation algorithm is used along with ant colony optimization model to measure the faults in the product. Instead of conventional distribution function in ant colony optimization process, differential evaluation is used to select the features. The selected features are trained in neural network model and then optimized through hybrid model. Performance of the proposed hybrid model is compared with artificial neural network and particle swarm optimization to validate the performances. The proposed model attains better reliability compared to other model by identifying the bugs in datasets with an accuracy rate of 96.2%. The further research could be progressed by improving the classification accuracy through two stage optimization process.

ISSN: 2582-337X
Submitted: 20.06.2020
Accepted: 29.07.2020
Published: 04.08.2020

References

1. Ajmer Singh, Rajesh Bhatia, Anita Singhrova (2018). Taxonomy of machine learning algorithms in software fault prediction using object-oriented metrics. *Procedia Computer Science*. 132. 993-1001.
2. Alireza Haghighatkhah, Ahmad Banijamali, Olli-Pekka Pakanen, Markku Oivo, Pasi Kuvaja (2017). Automotive software engineering: A systematic mapping study. *Journal of Systems and Software*. 128, 25-55.
3. Amir Elmishali, Roni Stern, Meir Kalech (2018). An Artificial Intelligence paradigm for troubleshooting software bugs. *Engineering Applications of Artificial Intelligence*. 69, 147-156
4. Diana-Lucia Miholca, Gabriela Czibula, Istvan Gergely Czibula (2018). A novel approach for software defect prediction through hybridizing gradual relational association rules with artificial neural networks. *Information Sciences*. 441, 152-170.
5. Fuqun Huang, Bin Liu (2017). Software defect prevention based on human error theories. *Chinese Journal of Aeronautics*. 30(3):1054-1070.
6. Helson Luiz Jakubovski Filho, Thiago Nascimento Ferreira, Silvia Regina Vergilio (2019). Preference based multi-objective algorithms applied to the variability testing of software product lines. *Journal of Systems and Software*. 151, 194-209.
7. Hua Wei Li, Ying Ren, Li Na Wang (2019). Research on Software Testing Technology Based on Fault Tree Analysis. *Procedia Computer Science*. 154, 754-758.
8. Jinyong Wang, Ce Zhang (2018). Software reliability prediction using a deep learning model based on the RNN encoder–decoder. *Reliability Engineering & System Safety*. 170, 73-82.
9. Kaliraj, Bharathi (2019). Path testing-based reliability analysis framework of component. based software system. *Measurement*. 144, 20-32.
10. Kapil Juneja (2019). A fuzzy-filtered neuro-fuzzy framework for software fault prediction for inter-version and inter-project evaluation. *Applied Soft Computing*. 77,696-713.
11. Peng Xiao, Bin Liu, Shihai Wang (2018). Feedback-based integrated prediction: Defect prediction based on feedback from software testing process. *Journal of Systems and Software*. 143, 159-171.
12. Wasiur Rhmann, Babita Pandey, Gufran Ansari, D. K. Pandey (2020). Software fault prediction based on change metrics using hybrid algorithms: An empirical study. Journal of King Saud University - *Computer and Information Sciences*. 32(4):419-424.
13. Xiao-Yi Zhang, Zheng Zheng, Kai-Yuan Cai (2018). Exploring the usefulness of unlabelled test cases in software fault localization. *Journal of Systems and Software*. 136, 278-290.
14. Yuanxun Shao, Bin Liu, Shihai Wang, Guoqi Li (2020). Software defect prediction based on correlation weighted class association rule mining. *Knowledge-Based Systems*. 196, 1-15.

Biography

Dr.S.Smys received his M.E and Ph.D degrees all in Wireless Communication and Networking from Anna University and Karunya University, India. His main area of research activity is localization and routing architecture in wireless networks. He serves as Associate Editor of Computers and Electrical Engineering (C&EE) Journal, Elsevier and Guest Editor of MONET Journal, Springer. He is served as a reviewer for IET, Springer, Inderscience and Elsevier journals. He has published many research articles in refereed journals and IEEE conferences. He has been the General chair, Session Chair, TPC Chair and Panelist in several conferences. He is member of IEEE and senior member of IACSIT wireless research group. He has been serving as Organizing Chair and Program Chair of several International conferences, and in the Program Committees of several International conferences. Currently he is working as a professor in the Department of Information Technology at RVS Technical Campus, Coimbatore, India.

Dr. Prof. Subarna Shakya is currently a Professor of Computer Engineering, Department of Electronics and Computer Engineering, Central Campus, Institute of Engineering, Pulchowk, Tribhuvan University, Coordinator (IOE) , LEADER Project (Links in Europe and Asia for engineering,eDucation, Enterprise and Research exchanges), ERASMUS MUNDUS. She received MSc and PhD degrees in Computer Engineering from the Lviv Polytechnic National University, Ukraine, 1996 and 2000 respectively. Her research area includes E-Government system, Computer Systems & Simulation, Distributed & Cloud computing, Software Engineering & Information System, Computer Architecture, Information Security for E-Government, Multimedia system.

ISSN: 2582-337X
Submitted: 20.06.2020
Accepted: 29.07.2020
Published: 04.08.2020