www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 6 Issue 3 March 2017, Page No. 20617-20620
Index Copernicus value (2015): 58.10 DOI: 10.18535/ijecs/v6i3.40

Evaluation of Software Testing Techniques Using Artificial Neural Network

V.Sathyavathy
Assistant Professor,
KG College of Arts and Science.

ABSTRACT
Software Industry plays a vital role in the current environment, it is very essential to minimize the fault in the existing software products. In SDLC life cycle, software testing becomes very much important which finds faults in the software that increases the efficiency of software. Most of the cost is occupied by the software testing process, it is product very essential to implement the automation technique that reduces the cost that increases the software reliability. In the automated testing process, various intelligent methods are used in order to avoid the manual process. Automation testing tools can be implemented to increase the quality of the software.

Keywords: Software Testing, Intelligent Testing, Automated Testing, Software Reliability.

I.INTRODUCTION

Software testing in software engineering process is an important component to identify the faults and rectify the errors. Reliability of software becomes a major part in the software product. It is impossible to find all combination of inputs and generate the test case accordingly. Error detection and correction are to be performed by various testing process models and techniques. Many testing techniques such as unit testing, integration testing, acceptance testing, regression testing, load testing, black box testing and white box testing. Various methods are to be found to decrease the time and effort of the software testers. Several researches say that automated and intelligent methods or tools lead the testing process cost effective.

Test automation is a way to make the testing process extremely efficient. The testing team can be strategically deployed to tackle the tricky, case specific tests while the automation software can handle the repetitive, time consuming tests that every software has to go through. One of the best aspects of automation testing is that the testing software is reusable. Not only that, but with every new test and every new bug discovery, the testing software directory can be upgraded and kept up-to-date. Test automation provides a persistent platform for your testing needs. The tests for which automation is usually implemented are extremely tedious. These techniques varied between artificial intelligence and statistical methods.

Figure.1 illustrates the classification. The following methods can use to automate software testing process, this is necessary to know what are testing process phases to understand how these methods can automate the testing process.

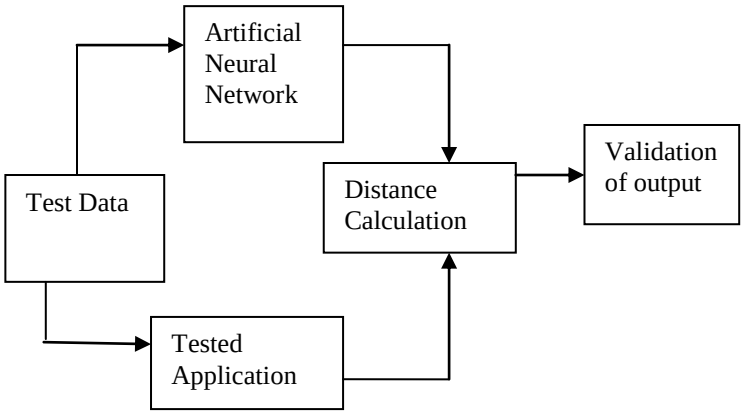

Figure 1: Classification of automated testing process

II. SOFTWARE TESTING PHASES

Testing process can divide into various phases in the following categories. Based on the classification, the testers must identify the current problem before moving to the next problem and by which phase it can be automated.

1. Software's Testers Environment

The software testers must identify the relationships and interactions between the software and its environment. All the interactions are performed through the software and the human resources. This phase can automate that stimulate the interfaces.

2. Test case Selection

Below mentioned are the parameters, depending on the application.

- Test case with different set of data
- Test case with different browser
- Test case with different environment

- Test case with complex business logic
- Test case with different set of users
- Test case that Involves large amount of data
- Test case has any data dependency
- Test case requires Special type of data

Each application is divided into modules. The test cases are identified for each module that are automated based on the parameters. Break each application into modules. For each module, analyze and try to identify the test cases which should be automated based on the parameters.

3. Measuring the test process

The test process determines how to test, which in turn Determines how effective and efficient the testing is, which in turn contributes greatly to how effective and efficient your development is Managing the testing process is essential for delivering, maintaining, and improving testing and development value

III.AUTOMATED SOFWARE TESTING METHODS CLASSIFICATION

Methods and associated structures and systems for automating software test procedures so as to enable automated black box and white box testing techniques to be performed in an automated manner.

The present invention further provides for test sequences to be easily maintained in synchronization with corresponding changes in the underlying source code files. In particular, structural and functional testing techniques are described in a test language of the present invention in a manner which it is combined within comments of the programming language used to implement the software product. Such comments are ignored by software development tool sets.

1.Test automation process

Testing tools of the present invention parse the source code files to extract structure and functional test sequences defined in the comments therein. The test sequences that are extracted are then performed by the test tool in combination with a software testing tool for the purposes of applying stimuli both for black box testing of the software product as a whole and for white box testing of internal functions and modules of the software product.

ANN (Artificial Neural Networks) is designed to resemble the structure and information processing capabilities of the brain. The architectural component of a neural network is computational units similar to the neurons in the brain. The neural network is a massive parallel information processing system that utilizes distributed control to learn and store knowledge about its environment.

The two inherent factors that influence the superior inherent capability of the neural network are its parallel distributed design and its capability to extrapolate the learned information to yield outputs for inputs that are not presented during generalization. These characteristics of the neural networks allow complex problems to be solved.

2. Evaluation of Test Scenario

A software tester needs a method to generate the output for the specific input for a test case. They compare the existing output with the actual output; if the outputs are not same then the fault is detected. To detect the fault testers need the oracle for automation of testing. The Oracle is a free source of expected output to detect the fault. Test oracle must accept every input specified in software specification and should give the correct result.

3. The process of using automated oracle

In this study, an ANN used to approximate this behavior. Then, this model can use as automated Oracle for generating correct outputs. Because ANNs have a suitable capability to modeling continues deterministic functions, this method of approximation has a good accuracy if is deterministic and without ambiguity.

IV.DESCRIPTION OF METHODOLOGY

The testing methodology can be viewed as a black-box testing approach

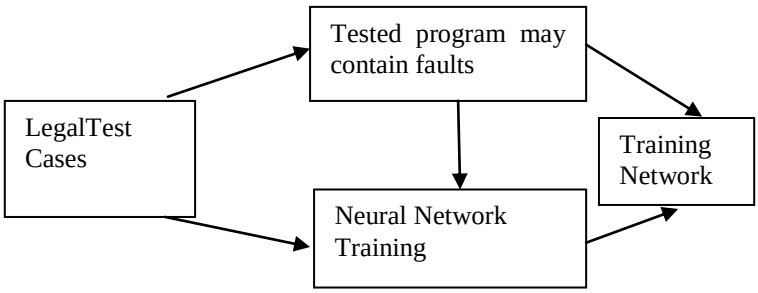

Figure 2: Training Phase and Evaluation Phase

Each test case can be provided as the input to a new version of the tested program to the ANN.

For each input, the comparison tool calculates the distance between the ANN output and the corresponding value of the application output. All the outputs are numbered from zero to one. The values are computed in one of the three intervals within the range. The comparison tool reports that the program output is correct. If the network prediction is different from the application output and the distance between outputs are very large (it falls into the category between the high threshold and 1.0), a program defect (wrong output) is indicated by the comparison tool. In both cases, the network output is likely to be correct and it is accurate enough to evaluate the correctness of the application output. However, if the distance is placed into the range of interval between the low threshold and high threshold, the network output is considered unreliable. In the last case, the comparison tool reports a program fault only if the network prediction is identical to the application output.

The comparison tool is used as an evolutionary method of comparing the results from the neural network and the results of the tested versions of the credit card approval application. An objective automated approach is required to ensure that the results have not been affected by external factors. This in effect replaces the human tester,

who may be biased by having prior knowledge of the original application.

The comparison tool uses the output of a neural network and the tested application output. The output is taken as the correct difference between the value of the code for each output and the value within the application. The absolute value of the corresponding output is equal to 1.0 if the expected and actual outputs are same[1].

The two types of outputs are handled differently by the comparison tool, as there are four possible combinations for a binary output and five for a continuous output. The analysis of a continuous output differs in one part with respect to a binary is equal to 1.0 if the expected and actual output are same.

V.USING A NEURAL NETWORK IN THE SOFTWARE TESTING PROCESS

Table1: Each output has a defined category

	Tested Application output	
ANN Output	Correct	Wrong
Correct	1	2
	True Positive	True Negative
Wrong	4	3
	False Positive	False Negative

V.DESCRIPTION OF THE EXPERIMENT

The experiment is divided into three parts, first one is the application instance, the second one is the artificial neural network and the third one is the tool for comparison that calculates the distance between the two outputs. The requirement specification for the credit card approval application is used to provide the necessary input and output attributes and placing the injected faults. The design of the neural network is partially dependent on the format of the application.

In the training phase and evaluation phase, the input data for the training of the neural network and the tested program are generated randomly. The data is given as the inputs that are fed into the tested program that generates the output for each application. The back propagation algorithm is used to pass the input to a two layer network and output vectors which is training instance. In the next phase, the tested program is created by making change at a time to the original version of the program. The data that is to be tested are then passed to the artificial neural network and tested program.

Using the comparison tool processes the output program that is tested to find whether output is correct or not. Test cases are to be generated to begin with the process of experiment to begin[2]. The attributes are created as the input with the specification of the program being tested; the outputs are generated by executing the tested program. The outputs are the binary and continuous, where the binary outputs are assigned a value of 0 or 1.

The output of each example which are continuous in different manner is placed into the correct interval With the range of possible values and the number of intervals used. The data set are used to process the data for training the artificial neural network[8]. The entire data set is presented to the training network with the number of epoch being specified. When the number of epochs has been reached then the back propagation algorithm gets concluded. To predict the output, regression tests are being conducted in which the network used as the oracle to predict the correct output.

Table 2: Instance of Credit card Approval

Name of the Attribute	Data Type	Attribute Type	Details
Serial ID	Integer	Input	Unique for each customer
Citizenship	Integer	Input	0-India 1-Others
State	Integer	Input	0-Chennai 1-Others
Region	Integer	Input	0-6 for different regions in India
Income Class	Integer	Input	0-if income p.a<10K 1-if income p.a<10K 2-if income p.a<10K
Age	Integer	Input	0-Female 1-Male
Marital Status	Integer	Input	0-Single 1-Married
Credit card Amount	Integer	Output	>=0
Credit Approved	Integer	Output	0-No 1-Yes

For example, customer 2 is not an Indian citizen, does not live in Delhi, is 18 years of age, is male, lives in region 2, has an annual income greater than 5,000, and is single with one dependent. Credit has been approved for this client for an amount of 1,000. Since a neural network can be trained only on numeric values, all categorical attributes (citizenship, state, and so on) were converted to numeric form.

The data in the training phase consists of 1000 possible test cases; the additional 500 test cases can be included for evaluating the expected versions of the application that follow the same format. The current dataset is larger than the previous that has the sufficient data to find the faults in the tested program

VI.EXPERIMENTAL RESULTS

The faults are identified and fed into the loaded credit card program. The application program output can be checked whether it is correct or not based on the faults that are generated. The Artificial Neural Network is used for checking the output in which the faults will be listed that is been tested with the possible test cases. Performance of error can be identified Normalized mean square by square rooting the ANN output square of the program.

VII.CONCLUSIONS

In this paper, Real time application can be tested with automation and provides mutation testing that generates the faulty versions of the original program.

Then the comparison tool is used to identify the correctness of the results that are identified based on the absolute difference between the two outputs. The neural network is to be one of the best models for testing the software application by providing the dataset that has the wide coverage of possible inputs.

The back propagation algorithm behind the model is very tedious method that is capable of performing generalization and one of its attributes ensures that that the network can be updated by learning new data. The trained network can be used to classify the new data as the software is trained to simulate is updated.[4] The Artificial Neural Network is capable of performing evolving versions of software. The benefits of this approach are to find the faulty coverage of the inputs. The drawbacks of this approach are to fully satisfy with the larger possible combinations of inputs and outputs. This approach can be used as the base for the future experimentation of the application project.

One of the applications is the generation of test cases that are used to generate the faults[3]. The dataset used by the comparison tool can be modified for one or more set of threshold values or by the process of fuzzification.The method can be further emprically evaluated by identifying more number of faults into a tested application.

References

[1] Voas JM, McGraw G. Software Fault Injection; 2012
[2] Choi J, Choi B. Test agent system design. In: 1999 IEEE International Fuzzy Systems Conference Proceedings; August 22–25, 2012.
[3] G. McGraw, "Building Secure Software: A Difficult but Critical Step in Protecting Your Business," Cigital, White Paper, available at: http://www.cigital.com/whitepapers/
[4] Weyuker E, Goradia T, Singh A. Automatically generating test data from a Boolean specification.IEEE Transactions on Software Engineering 2013;SE- 20(5):353–363.
[5] DeMillo RA, Offutt AJ. Constraint-based automatic test data generation. IEEE Transactions on Software Engineering 1991; SE-17(9):900–910.
[6] Anderson C, von Mayrhauser A, Mraz R. On the use of neural networks to guide software testing activities. In: Proceedings of ITC'95, the International Test Conference; October21–26, 2015.
[7] Khoshgoftaar TM, Szabo RM.Using neural networks to predict software faults during testing.IEEE Transactions on Reliability 2011; 45(3):456–462.
[8] Khoshgoftaar TM, Allen EB, Hudepohl JP, Aud SJ. Application of neural networks to software uality modeling of a very large telecommunications system.