Artificial Neural Network Approach for Software Product Line Testing

Ashish Saini, Rajkumar, Satendra Kumar

Abstract: Software Product Line Testing (SPLT) is an immensely important task because it confirms the validity of a product while it is a time consuming and costly process. To minimize the time and check the validity confirmation of a product, a method is required to decide whether a product of an SPL is faulty or not. A tester can decrease the actual cost of testing and its maintenance cost as well as time by using such efficient method. In this paper, we concentrate on an idea, where we use an Artificial Neural Network (ANN) technique, which works to test a software product line. In this technique, the backpropagation algorithm is used to train a Neural Network (NN) based on the set of test cases of the product's actual version. The trained network treats as a black-box testing approach, in which two parts are presented for an algorithm. SPL product's validity is measured by the distance between actual and faulty outputs.

Index Terms: Testing, Software Product Line, Artificial Neural Network, Test Cases, Feature Model, Product Line, Configurations.

1 INTRODUCTION

Software Product Line (SPL) is a very optimistic approach to software product advancement that allows organizations to develop number of comparative products in similar application space. SPL decreases the products development and maintenance cost, moreover increases the productivity which occur by artifacts reuse, and also reduce the market time. SPL artifacts like as architectures, test cases, components, models, documents and requirement specifications are the metamorphic artifacts of core assets. These artifacts are sufficient to develop non identical products from a particular set of core assets by utilizing and recognizing commonality between the products in a product line. According to Software Engineering Institute, “An SPL is a software-intensive system that share common, managed features to satisfy the certain needs of a specific segment of the market and developed in a prescribed manner using a common set of core assets”[1][2]. Companies like Toshiba, Bosch, General motors etc. already applied SPL engineering approaches effectively, in developing intensive system. Industries (small or large scale) want to expand their business and also capture the market in very less time. To grow their business, companies want to create different and unique products according to the requirement of customer or market needs. Now a days, industries are moving to adopt the SPL concept. But an issue that always persists with SPL approach that, the combinations of features of licit and unique artifacts are work properly or not. So, testing is a better way to check the product and their artifact’s combinations. Testing is an expensive, time consuming and a tedious process but, it control the quality of a product. Testing of an SPL is a challenging and significant practice for any organization’s goals. These goals are to improve the quality, reduce the time and cost of implementation of products. An SPL testing should follow some characteristics like [3] [4]:

- In development process, testing should be done instantly as possible for each artifacts.
- Must follow the property of reusability regarding assets of product line.
- The test method should manage a large number of test cases, because SPL will produce a test case exponent as the features increases.
- If probably changes occur in an SPL, there should be no effect on test artifacts.
- An approach should test the artifacts combinations and final product too.
- Testing methods should be structured whereby, it could achieve the aim of traceability.

Number of researchers are articulated numerous testing methods to test product line and improve testing costs [4][5] discuss these later sections, but no any researcher uses the neural network to test the Software product line and its artifacts.

2 ARTIFICIAL NEURAL NETWORK

BACKGROUND

The arrangement of Artificial Neural Networks (ANNs) is similar to the organization and information processing capabilities of the brain. An ANN composed by a huge number of massive interrelated processing elements called neurons, which works in parallel fashion to solve a specific task. In a neural network neuron consists one or more layers and connections between neurons have related synaptic weight. The synaptic weight on the connections of neurons encodes the knowledge of the network. Every neuron used in the network, used for calculations occurring in the network, which contributes to the training or whole learning process of neural network. Each neuron has its own memory and the output of every neuron depend, on its value in memory and the signals attain at that neuron [6]. Intelligence power of a neural network depends on the collective behavior of neurons, where limited operations performed by each neuron. Thus, the neural network is a large-scale parallel information system, which uses a distributed system to learn and store the information of its environment. There are two components which affect, the ability of neural network computation. First One is designing of

- Ashish Saini is currently pursuing Ph.D. at Department of Computer Science in Gurukula Kangri Vishwavidyalaya, Haridwar, India, PH-8791158699. E-mail: raashishsaini@gkv.ac.in
- Rajkumar is an Assistant Professor at Department of Computer Science in Gurukula Kangri Vishwavidyalaya, Haridwar, India. E-mail: rajbhatiagkv@gmail.com.
- Satendra Kumar is currently pursuing Ph.D. at Department of Computer Science in Gurukula Kangri Vishwavidyalaya, Haridwar, India, PH-7248096172. E-mail: satendra04cs41@gmail.com
- (This information is optional; change it according to your need.)
parallel distribution and another one is the ability to expand the learn information which comes from the output for those inputs that are not present during the training. Thus, with these characteristics, a neural network can solve complicated tasks. Some complex tasks like, Data classification, function approximation and data mining that can be solved with the help of the neural network. Our approach works in that manner, where a trained network produces different outputs when representing a network with input signals. In phase of training, for the training algorithm, the weight of the network can be upgraded with a set of training examples. Every instance occurs in the organization of two components, for the first input and the second output. When the complete set of training example present to the network then one pass (forward or backward) or epoch is concluded. Thus, a neural network works as an input-output mapping system. Each computational neuron of neural network made up of three features: set of synaptic weights, each for interconnections; summation unit which is used to sum, the multiplication of input signal and corresponding weight; an activation or squashing function that keep bound the output in a range, referred by user. Demonstration of a neuron depicts in Fig.1 [18]. Where, for neuron i the input signal is $x_i$; $w_{ij}$, weight associated between neuron and input signal. Now linearly combine the Multiplication of weights with input signals. The obtained resultant value is used for output of the activation function.

The standard training approach backpropagation which works for multi-layer feed forward neural networks. In an algorithm input signal passes forward and the erroneous signal passes backward with the help of network. A multilayer neural network works efficiently, feedforward resolves complex and general tasks while backpropagation is a systematic method for training [7][8]. Objective behind this idea to propose an automated testing method to test the combination of artifacts as well as the end product of a product line. Further this paper structured as follows: section 2 discuss some research work published by researchers in concerning subject of research. Outline of propose idea is defined in section 3. Description about the testing methodology discuss in section 4. In section 5, we conclude the paper with future work.

3 RELATED WORK
Much research work has been done in the SPL testing area. The V Model divides the testing into three categories: system, unit and integration testing [9]. On the other side, the W model is similar to the V model, which divides the testing into two sub-models’ domain and application testing [10]. McGregor [11] clarified key challenges regarding testing in SPL and with this author proposed combinatorial testing for SPL, which works for both system as well as integration testing. Author confirmed that combinatorial test design does not test entire interactions of features, besides this test cases covers all those features that can have n-way interaction. Ferreira et al. [12], presents a mutation approach to test the product line, this approach selects the product and test the fundamental part of product entitled feature model, and also test features. This mutation-based testing enhances the possibility to find the faults where pairwise testing not capable to do this, and confident that the final products match the market requirements. To derive test case, Bertolino and Gnesi proposed a testing methodology named Product Lines Use Case Test Optimization (PLUTO), which uses category partition method and use cases in the form of a model where dependency considered in between use cases and test cases and results yields in natural language. But, PLUTO methodology struggle with the order of testing. PLUTO works in both the phases of the product line, domain and application engineering respectively [13]. Besides this, V activity diagram [9] uses branch coverage norms to test the test cases and consider testing order as advancement, it is the extended version of UML activity diagram and does not consider the advantage (i.e. dependency between cases) of PLUTO. To meet the requirement and enhance the quality of products and product line, Ali and Ramadan [14] introduced an improved approach which uses PLUTO and V activity diagram advantages as discussed above. This approach maintains the testing order as well as dependencies between use cases and provides the result in form of complete test. An increase in testing cost is a key issue in SPL. In order to reduce test costs, Akbari [3] focuses on integration testing and proposes a feature-based approach called PINE (Prioritized Integration testing in software product line) in which the author prioritize the integration test cases. In this method, the author reuses the artifacts of domain engineering and, to prioritize test cases use pruning techniques for feature models. Method also offers the ability to partitioning features into sub-tree of features. It provides many favourable results also like decrease cost of integration testing and identifies integration faults in domain engineering when works on certain case studies. Patil and
Prakash [15] presented an improved combinatorial testing approach, testing with a combinatorial approach considers a promising method. They enhance the combinatorial testing with approach IPOG (In-Permanent-Order-General) which is a test case generating method. To perform combinatorial coverage, they incorporate that method with a neural network which provides the most appropriate test case scenario. New approach suitable to find that how much combinatorial previously performed and how many fresh test cases are required in existing test cases, through which suitable coverage of testing could be achieved. There are many more papers that discussed the testing of the product line. But, after survey those papers it came to our notice that, no author has shown any method in which the neural network has been used for testing of product line.

4 METHOD’S OUTLINE

In the above section, we have seen that no author has discussed any such methods of using the neural network approach for software product line testing. In this section, we will introduce a method for testing a product line with the help of a neural network.

The proposed method is summarized as follows:
1. Design/Select the feature model of SPL.
2. Generate configurations of an SPL product.
3. Use the configuration to generate a dataset.
4. Apply a neural network approach.
5. Lead to errors or the correct form.

An overview of the proposed method presented in the form of a flowchart in Fig.3.

4.1 Create Feature Model
4.2 Generate Configurations
4.3 Generate Dataset
4.4 Apply Neural Network
4.5 Make Changes in Test Cases

Figure 3. Flow Chart of Proposed Method

On the other hand, creating a new version (faulty dataset) of the actual dataset and output generated using this new version of the dataset. Then, using the comparison tool to measure the distance between both outputs, representing the distance, we can determine the effectiveness of the product. If the product is valid, the process will end and the existing configuration is valid, but if the product is not valid, we make changes in the existing test case or configuration.

5 PROPOSED SPL TESTING METHODOLOGY

The proposed methodology might seem to be like a black-box approach. This methodology works into two stages, one is training that depicted in Fig.5. Where randomly generated valid configurations take as an input with their corresponding dataset, on the other side generate configurations, which may
contain defects and create a dataset for corresponding configurations. The actual dataset is created and trained by the neural network, from which a trained network is obtained, and this trained network can act as an automated tool for testing the upcoming versions of the product, and this process known as regression testing. The second phase is the evaluation phase, as shown in Fig. 6, where the actual dataset is used as input to the trained neural network and misconfiguration dataset for testing. For each input, the distance between the output of the ANN and the value corresponding to the faulty configuration is measured. The validity of a product depends on the distance calculated, and if the calculated distance coincides with the test case output, then our product is valid. Furthermore, if the output of the test case does not coincide with the calculated distance, then the network reports of an invalid product. The validity of any product depends on the distance calculated by the network which can be measured by any comparison or distance rule, which compares the result given by ANN to the results of faulty test cases. With the help of an automated method, we can ensure that outside factors have not affected the results. Apart from this, the condition of biasness may occur if a tester already aware about the original product or output.

6 CONCLUSION AND FUTURE WORK
Testing of an SPL is a complex task, either testing the product's effectiveness or verifying the functional combination for that product. Performing this manually can be time-consuming and much costly. To decrease the time and cost of SPL testing, we propose a method in which an artificial neural network acts as an automatic predictor for SPL testing. In the proposed method, a dataset is generated from a test case and then used as input to the training, and a dataset is generated for the wrong configuration. The neural network is applied to the dataset and calculate the difference between the outputs is found. After the difference is obtained and compare with the faulty configuration, it can be said that the product is valid or not. The neural network approach might be suited to product line testing, and implementing this approach in real-time examples or as a result experiment will be our future work.

REFERENCES


