A Single-Network ANN-Based Oracle to Verify Logical Software Modules

Seyed Reza Shahamiri, Wan M.N. Wan Kadir and Suhaimi Ibrahim
Department of Software Engineering
Faculty of Computer Science and Information System
Universiti Teknologi Malaysia
Skudai, Malaysia
admin@rezanet.com, wnasir@cs.utm.my, suhaimiibrahim@utm.my

Abstract—Test Oracle is a mechanism to determine if an application executed correctly. In addition, it may be difficult to verify logical software modules due to the complexity of their structures. In this paper, an attempt has been made to study the applications of Artificial Neural Networks as Single-Network Oracles to verify logical modules. First, the logical module under test was modeled by the neural network using a training dataset generated based on the software specifications. Next, the proposed approach was applied to test a subject-registration application; meanwhile, the quality of the proposed oracle is measured by assessing its accuracy, precision, misclassification error and practicality in practice, using mutation testing by implementing two different versions of the case study: a Golden Version and a Mutated Version. The results indicate that neural networks may be reliable and applicable as oracles to verify logical modules.

Keywords—software testing; automated test oracles; logical software modules; artificial neural networks; mutation testing

I. INTRODUCTION

Software testing is the process of evaluating the software behavior and checking whether it operates as expected in order to improve its quality and reliability. Since the testing process can be highly time and resource consuming [1], complete testing is almost impossible; thus, testers use automatic approaches to facilitate the process and decrease its costs [8].

Test Oracle is a mechanism to determine whether an application executed correctly. It is a reliable source of how the Software Under Test (SUT) must operate [17]. They also expected to provide correct output(s) for any input(s) specified in software specifications and a comparator to verify the actual behavior automatically. Automated test oracles are helpful in providing an automated software-testing framework.

Logical modules are one of the important structures that exist in most of the software applications. However, they may be difficult to be verified because they could be too complicated. This complexity often caused by various conditions and several paths in their structures that may cause them to behave differently. Moreover, providing an oracle for such modules requires programmers and testers to understand the application domain completely. Consequently, automated oracles may decrease the cost of testing such structures significantly, as they may save a great deal of manual effort.

After test cases are executed and results of the testing are generated, it is necessary to decide whether the outputs are accurate. To verify the behavior of the SUT, correct outputs need to be compared with the results generated by the software. The results generated by the SUT that need to be verified are called actual outputs, and the correct results, which are used to evaluate actual outputs, are called expected outputs [11]. The process of finding correct and reliable expected outputs is called oracle problem [12]. According to [13], oracle information and oracle procedure are building-blocks of each test oracle. The former is a source of expected outputs and the last is the comparator.

There are some challenges to develop an automated test oracle [16]. First, it could be difficult and expensive to provide the output domain. In general, expected outputs are generated manually based on software specifications, domain specialist information and programmers knowledge of how the software should operate (i.e. direct verification [12]). Another challenge is how to map the input domain to the output domain. Since manual mapping may decelerate the testing process significantly, and an automated oracle is impossible without automatic mapping, it is highly recommended to handle the mapping challenge. The proposed approach tries to address the mapping challenge to test logical software modules. The final challenge is the automated comparator. The comparator needs to consider some tolerance instead of directly compare the actual and expected outputs. We defined a set of thresholds to define the comparison tolerance and adjust the precision of the proposed oracles.

In this paper, we aim at investigating the application of Artificial Neural Networks (ANNs) as Single-Network Oracles to verify logical modules. A Single-Network Oracle uses an ANN to simulate the SUT in order to map the input to the output domain. The logical rules being verified were simulated by the ANN using a training dataset generated based on the software specifications and domain expert knowledge. Then, we evaluated the proposed oracle using mutation testing by applying it to test a subject-registration application. First, a mutated version of the application was provided and injected with some faults. Then, a fault-free version of the application was developed as a Golden Version to evaluate the capability of the proposed oracle to