УДК 004.052.3

Tulchak L. V.

Senior Teacher of Foreign

Languages Department, VNTU

Noskovenko Y. P

Undergraduate of VNTU

USING A NEURAL NETWORK IN THE SOFTWARE TESTING PROCESS

Vinnytsia National Technical University

Анотація:

У даній роботі розглянуто перспективи застосування нейронних мереж у процесі автоматизації тестування програмного забезпечення.

Ключові слова: тестування програмного забезпечення, нейронні мережі, інтелектуальне тестування, автоматизоване тестування, надійність програмного забезпечення.

Abstract

In this paper considered the prospects of using neural networks in the process of software testing automation.

Keywords: software testing, neural networks, intelligent testing, automated testing, software reliability.

Introduction

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 [1].

Research results

When it is necessary to check the operation of the entire software as a whole, none of the automatic methods can even come close to manual testing. However, it has a significant drawback, human resources are limited.

This is due to the following problems:

1) the complexity of writing tests does not depend linearly on the complexity of the software;

2) there is no randomness in the behavior of the tests.

Thus, it is necessary to:

– simplify the creation or automatically create a test coverage;
– add randomness to the behavior of the test script [2].

The software, in its essence, represents a function in a multidimensional space. In such a mathematical model, it is easy to apply fuzzy logic methods, that is, neural networks.

The neural network is a mathematical model based on the cognitive processes of the brain. Just like the brain, a neural network consists of nodes (neurons) and variable connections between them (the weights of the neural links).

Neurons are divided into 3 groups according to their properties: input neurons, output neurons and hidden layer neurons (computational nodes). In fact, a neural network is a special way of specifying a function.

The learning process is quite simple. A training sample is prepared – a set of pairs of input and output vectors, with known and correct values. Elements of this set satisfy the following property – independence, therefore, the sample can be expanded, and its elements can be rearranged in any order. The training sample is passed through a neural network, and as a result of this process, the weights in the neural links change in such a way as to define a function that satisfies the training sample. Obviously, the quality of the work of a neural network depends on the quality of its training.

A test sample is prepared similarly to the training sample, but on different values. If the test result is unsatisfactory, it is better to train the network or change its structure.

A well-trained neural network is able to predict the result by a set of input values. Together with this additionally, it solves the classification problem if classes in the input set are pre-defined, or the clustering problem, that is, the identification of classes in the set.

Thus, the neural network in the test task is able to solve both subtasks at once, due to the mechanism of the prediction of the result [3,4].

A multi-layer neural network is trained on the original software application by using randomly generated test data that conform to the specification. The neural network can be trained within a reasonable accuracy of the original program, though it may be unable to classify the test data 100 percent correctly. In effect, the trained neural network becomes a simulated model of the software application. When new versions of the original application are created and "regression testing" is required, the tested code is executed on the test data to yield outputs that are compared with those of the neural network. The new versions do not change the existing functions, which means that the application is supposed to produce the same output for the same inputs. A comparison tool then makes the decision whether the output of the tested application is incorrect or correct based on the network activation functions.

But there are a few technical questions:

- the amount of generated test coverage for unit testing (since in the trivial case, you will have to create all possible input value rebounds);
- the degree of randomness and the way it is set to simulate user work (because a set of random actions is not the best way to test);
- the structure of the neural network remains unique for each task, which prevents its mass distribution [5].

Conclusions

Proceeding from the above, neural network technologies together with widely used methods of automated testing are quite capable of improving the results of automated testing in testing problems of complex software systems.

The neural network should be a promising method of testing a software application provided that the training data have a good coverage of the input range. 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.

REFERENCES

1. Choi J, Choi B. Test agent system design. In: 1999 IEEE International Fuzzy Systems Conference Proceedings; August 22–25, 1999.
2. Voas JM, McGraw G. Software Fault Injection; 1998.
3. 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; October 21–26, 1995.
4. Boehm, B.W. Software Engineering Economics, Prentice Hall, USA, 1981.
5. Kirkland LV, Wright RG. Using neural networks to solve testing problems. IEEE Aerospace and Electronics Systems Magazine 1997;12(8):36–40.

Носковенко Юлія Павлівна – студентка групи 2СІ-14б, факультет комп’ютерних систем і автоматики, Вінницький національний технічний університет, м. Вінниця, e-mail: angel479870@gmail.com

Науковий керівник: ***Тульчак Людмила Володимирівна*** — старший викладач кафедри іноземних мов, Вінницький національний технічний університет, м. Вінниця

Noskovenko Yulia P. – Faculty of computer systems and automation, Vinnytsia national technical University, Vinnytsia, e-mail: angel479870@gmail.com

Supervisor: ***Tulchak Liudmyla V.*** — Senior Lecture, Chair of Foreign Languages, Vinnytsia National Technical University, Vinnytsia