UDC 3338.01
DOI: 10.20535/SRIT.2308-8893.2021.4.03

RESEARCH OF INTELLIGENT METHODS OF SOFTWARE TESTING

T.H. KAZIMOV, T.A. BAYRAMOVA, N.J. MALIKOVA

Abstract. This article presents the examination of several techniques and tools used in the automated software testing process. Considering the ever-growing importance of software testing, several possible implications of implementation of artificial intelligence into this area are also discussed. The main objective of this study is to examine the field of test automation by categorising related test activities, to which artificial intelligence tools can be applied for increased efficiency, and evaluate the impact of the application. The main software testing methods are white-box, black-box, and grey-box methods; an effort has been made to determine a connection between the given testing methods and artificial intelligence methods. A brief summary of several artificial intelligence engine tools used to automate testing was also provided. Lastly, the possible future benefits from usage of AI in software testing was investigated..

Keywords: software testing, automated testing, artificial intelligence.

INTRODUCTION

The increasing complexity of software systems require modern testing methods that can matchthe complexity of these systems to ensure defendable verification of these systems. Today, software complexity has reached the heights where making mistakes is an inevitable part of the development process. Much progress has been achieved in test and verification techniques, however software development still requires considerable effort in testing before delivery to the customer. Software testing is using test methods to eliminate the errors at different stages of the software life cycle.

Testing is an imperative stage in the Software Development Life Cycle (SDLC) [1]. The qualityand suitability of the software is determined by the extent of fulfilment of its intended purpose. This is evaluated and verified at the testing stage. Hence, testing stage approximately comprises 50% of software development time [2].

In SDLC, the software is considered unfinished if it is untested, therefore the sooner an error isdetected the cheaper it becomes to fix [3]. Generally, the intent of program testing is not to prove that the software is error free, but rather to demonstrate the viability of the software before installation. Software testing is an important research area, and therefore it has recently experienced significant advances.

Software testing can be manual or automatic. Manual testing does not require knowledge of any testing tool, it requires a lot of effort and more human labor. Participation of humans in the testing process does inevitably result in introduction of errors, and it is important to identify these errors. The use of human labour in software testing is highly unfavourable due to increased execution speeds, insufficient scope of software testing, and requirements in manpower [4].

Following Fig. 1 demonstrates systematic and automated testing capability to reduce huge costs. Using traditional testing methods in software testing is considered inconvenient due to poor execution speed, insufficient test coverage, and additional manpower. All of the listed issues are easily eliminated by the implementation of automated testingmethods. Automated testing is a process using software separate from the software under test to control test execution and compare actual results with expected results [5].

Automation tools are used to automate many sections of manual testing, but they do not cover all sections [6]. Automated testing usually saves time, a tester can efficiently run large numbers of tests in a short period, and testing that would be difficult to do manually can be automated. At the same time, test automation saves money and effort, improves the quality of testing tasks, and also helps to improve the reliability of software [7]. Test automation requires an experienced tester with knowledge of automation tools and software under test.

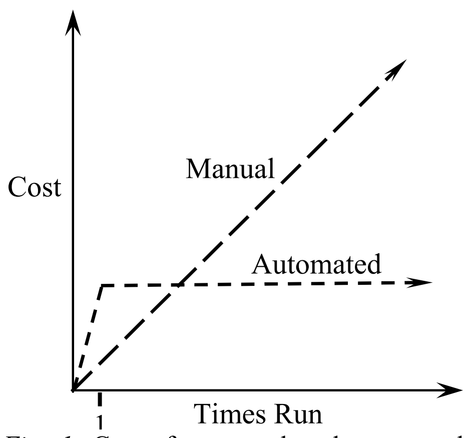

Fig. 1. Costs for manual and automated testing

Automated testing has its advantages and disadvantages (Table 1) [8].

T a b l e 1 . Advantages and disadvantages of automated testing

Advantages	**Disadvantages**
Improves accuracy and faster error detection compared to manual testing	Choosing the right tool takes a lot of effort, time and development plan
Saves time and effort by making testing more efficient	Knowledge of testing tool required
Increases test coverage as multiple test tools can be used concurrently, allowing parallel testing of different test scenarios	The cost of buying a testing tool and, in the case of reproduction methods, testing maintenance is slightly more expensive
Automation test script repeats	Writing Automated Test Scripts Requires Skill

The determining factors for successful and effective software testing projects are: (a) choosing and using the right test method, and (b) choosing and using the right test automation tool.

It is impossible to carry out automated testing without the use of good tools. Recently, the application of artificial intelligence and machine learning in testing has been rising. Artificial Intelligence has had a major impact on software engineering and has found wide application in software testing in recent years. Many experts in the field of software testing and quality assurancehave pointed out the necessity of integration of artificial intelligence into the software testing processes. Organizations that carry out software testing using artificial intelligence models and techniques have stepped into a new era, increasing the competitive advantage by producing applications exceeding the set expectations. Fundamentally, the most basic component of all software is data; input and output, source codes, websites, and databases are all examples of suchdata. Artificial intelligence algorithms and techniques excel at the efficient processing of big data.

Artificial intelligence and machine learning algorithms have found widespread application in real-life industries such as commercial, industrial, and digital services, however, their usage in the software testing methods is relatively new. The complexity of the software testing methods and techniques have limited the practical application of artificial intelligence to academic usage only. In theory, it is estimated that the highest level of automated testing will be achieved with the help of artificial intelligence. The main objective of this research is to determine how artificial intelligence will be used and how much it will contribute to the field.

METHODOLOGY

Testing methods play a crucial part in widening the scope of the test and in determination of strategies used in the test result analysis [5]. Several test methods are available to determine various qualitative aspects of the software application. Although it is not possible to use all testing methodsin testing of any given software, the testing specialist may select and apply more than one testingmethod based on the requirements of the test, the type of the software being tested, the monetary and time constraints. The desired result is achieved more easily if more testing methods are beingapplied [6]. The methodologies used in the software testing are described below [7]:

- **White box testing.** This method is also referred to as open-box testing, white-box analysis,and open-box analysis. During the testing process, the internal workings and structure of the system are known. It is a very effective problem detection and resolution method, dueto early identification of the issues preventing the escalation resulting in costs and time. In white-box testing method, the tester must have full access to the source code in order to identify the part of the device that is causing the undesirable outcome. Application of this method is therefore limited to the cases where the tester has full comprehension of the interaction between software components. Therefore, a skilled tester is required to carry out this testing method. It is the most time-consuming method, preventing its application to large systems and networks. It is most commonly used in web services applications [8].
- **Black-box testing.** The internal workings of the black-box testing device are unknown tothe user and are not accessible. This method can be used without any knowledge of the internal working mechanisms of the device and at various stages of the testing process. It only examines the key aspects of the system and therefore the scope of this approach is limited. However, it is the fastest testing approach among the listed methods. The main goal of this method is to determine the extent by which the specified requirements are metby the system. This method ensures that input and output data are as expected [9]. The most ubiquitous method is fuzzing (purposefully inputting incorrect information to determine the errors) [8].
- **Gray-box testing.** This method is a combination of white box and black box testing methods. It combines the advantages of both black and white box testing. It uses built-in data structures and algorithms to compile test scripts more

than black box testing and lessthan white box testing. This method is only relevant when doing integrated testing between two or more code modules written by different developers, provided the interfaces are open for testing [10]. Gray-box testing is considered the most unbiased testing tool as it does notrequire the tester to enter internal source code. However, the scope is limited due to the lack of source code. This method is used to test an application that has partial knowledge of the fundamental aspects of the system. Some forms of gray box testing are regressiontesting (checking for new software changes) and model testing (checking the architectureand design of an application) [8].

The methods used in machine learning fall into two categories: supervised machine learning andunsupervised machine learning. There is also a reinforcement machine learning method. Based on“Google Scholar”, the most popular machine learning method used in software testing in the last 3 years is control and reinforcement methods. Amplification is the most popular machine learning method for “Scopus Elsevier” and “Web of Science”. During the last three years, artificial intelligence algorithms and machine learning methods have been increasingly applied to “black- box” testing methods. Although not as popular as the two methods mentioned above, unsupervised learning is very important in black box testing. Black box testing is the most widely used method of artificial intelligence. The most popular AI algorithms are "clustering" (a general uncontrolled learning method), ANN (a method used in controlled and reinforcement learning), and GA (a method commonly used to reinforce learning). Artificial intelligence is not widely used in white box and gray box testing [11].

REVIEW OF THE SCIENTIFIC LITERATURE ON THE SUBJECT OF RESEARCHING AI METHODS FOR TESTING

In recent years, various parametric and non-parametric software reliability models have been developed in the software testing process to assess software reliability. Despite this, no single model can accurately predict the number of software bugs under all testing conditions. Modern software is developed with many features and sizes, and assessing the reliability of software is an extremely challenging task. In [17], research and study of these testing features show that a deep neural network (NN) model can have suitable prediction capabilities. This study uses a repetitive NN (RNN) codec-based deep learning model to predict the number of software errors and assess software reliability. Experimental results show that the proposed model has better forecasting characteristics in comparison with other parameters and NN models. The authors show that the deep learning neural network model has better prediction performance than other models. This model is more reliable in predicting software reliability.

In [18], a new idea of using machine learning to automatically generate test suites is proposed. The NEAT (Neuroevolution of Augmenting Topology) algorithm was used to automatically create new test suites or to improve the coverage of already created test suites. This method automatically generates test suites for white box testing.

In [19], artificial neural networks were used for regression testing. For the optimal selection of the highest priority regression checks, test management systems were used, which allow to determine the priorities of test cases depending on the corrections made to the product code, based on a neural network model. The collection and analysis of data on changes in the code from the version control system is used as input data for an artificial neural network. The output in such a model is the results of the regression tests, that is, the fact that an error was found or that the product is performing as expected. Thus, the training of the neural network is based on real data obtained from the results of running tests at the software development stage. A trained neural network is able to prioritize test cases and optimize resources for regression testing.

In [20], the authors presented optimization models that ensure the correct selection of values for various attributes of test plans, test cases, defect reports and other documents generated during testing. Described the structure of a neural network and the concept of its application for the most accurate selection of attribute values.

Swarm intelligence is a new area in optimization, and researchers have developed various algorithms that simulate the behavior of various flocks of animals and insects such as ants, termites, bees, birds, and fish. These algorithms are used to reduce the time and cost of testing in general and regression testing in particular. Two such widely used algorithms for prioritizing test cases are the ant colony algorithm and the bee colony optimization [21].

In [22], the authors proposed using UML state diagrams and Ant Colony Algorithms to generate test data. The advantages of the approach they propose are that it directly uses standard UML artifacts created in software design processes, and also automatically generates a valid test sequence that is not redundant and achieves the stated coverage criteria. An important aspect of this algorithm is parallelism: several solutions are simultaneously built that exchange information during the process and use information from previous iterations.

In [23], the authors carried out an experiment. In this experiment, the optimal testing process for regression testing is determined using the multipurpose ant lion optimization (MOALO) algorithm. This algorithm is completely suitable for identifying test cases of mutants. To gain access to software quality through regression testing, they used mutant test cases. Ensuring the availability of mutants is seen as a multi-purpose modernization problem that can be addressed by the MOALO calculation. The experimental results showed that the proposed MOALO method provides better and more effective results than other methods.

In [24], the authors proposed a fault coverage regression system using the Bee Colony Optimization (BCO) algorithm. The idea is based on a natural bee colony with two types of worker bees that are responsible for the development and maintenance of the colony: scout bees and forage bees. The BCO algorithm developed for a suite of fault coverage regression tests is based on the behavior of these two bees. The algorithm was designed for failure coverage in order to achieve maximum failure coverage in the smallest units of execution time for each test case. This algorithm automates the process of prioritizing the test suite according to the bee colony optimization criteria. Using this algorithm, you can automatically generate a test case and test your application efficiently, with less cost and time.

For quality control of video games, the software testing process is essential. Gaming environments are complex and interactive systems. These environments can include level geometry, interactive objects, player-controlled and non-player characters, and so on. Depending on the number and complexity of the levels, manually testing them can be significant. This problem can be solved using artificial intelligence (AI) in testing. In [25], the authors investigated the most promising and current artificial intelligence applications for video game testing, which can serve as a reference for anyone getting started in the field. Automated testing of video games using AI is a relatively new area of research, often without an established terminology and structure.

ARTIFICIAL INTELLIGENCE TOOLS USED IN SOFTWARE TESTING

The application of artificial intelligence to the software testing process has led to radical changesin this field. Currently, the process of developing and testing the program continues to evolve. During software development, testers must ensure that customer satisfaction is achieved. For thisreason, the use of artificial intelligence bots in the testing process is inevitable. This section provides examples of tools that apply the capabilities of artificial intelligence and machine learningin automated testing. During the process, some tools are used without machine learning. However,these tools are not yet considered scientifically reliable as they are more relevant to companies' marketing activities. The most commonly used AI tools in software testing are:

- "Mabl". Boston-based startup "Mabl" has introduced a program that simplifies the functional testing process for developers using machine learning. However, users no longerhad to manually type large lines of code. Instead, developers should show the workflow they want to test in the program and let "Mabl" do the rest. "Mabl" will perform these testsand adapt to even the smallest interface changes. It can detect software bugs, "JavaScript"errors, visual changes and more. This makes testing as simple as possible for users. Co- founders of "Mabl" are Izzy Azeri and Dan Belcher, who were founders of Stackdriver (formerly acquired by Google). Not surprisingly, "Mabl" runs on Google's Cloud platform. In short, "Mabl" makes the continuous testing process both easy and measurable with its advanced artificial intelligence technology [12].
- "Eggplant". This tool is not model based, it is an amazing artificial intelligence tool for testing software using different concepts. "Eggplant" is a powerful tool that uses different complex algorithms to achieve different goals. While testing the software, the tool performs the following [13]:
  - ✓ Detects errors: the tool looks for common patterns that may indicate the presenceof errors.
  - ✓ Evaluates the scope: The tool performs comprehensive analysis in terms of information, movement and status.
- "Applitools" is a tool that visually monitors software applications using an advanced algorithm. "Applitools" serves professionals and teams especially in "DevOps", "Digital Transformation'', "Quality Assurance" and many more. The tool identifies possible errorsand flaws in the implementation of the program [14].
- "Sealights" is also a Web-based testing tool for developers and quality assurance professionals. This tool is indispensable for testers as it takes a long time to perform repeated tests. It offers machine learning technology that evaluates vehicle codes and testsperformed. At the same time, with this tool, testers can learn

the true scope of tests. The tool performs different types of tests in terms of performance, functionality, unit testing and more [14].

- "Testim". With this test tool, the user can automatically save and code test cases. The tool has auto-correction features that reduce the amount of manual work required when changing the target. It also allows the user to use different datasets and browsers simultaneously by performing visual verification on the tested pages. The features of AI focus on the ease of use of these steps and the reuse of the steps between different test scenarios. As the number of tests performed increases, more reliable results are obtained [15].
- "Functionize". Uses machine learning and artificial intelligence to accelerate the creation, diagnosis and maintenance of tests. One of the best features of this tool is that you don't have to think much before you do the test. Only those who want to be tested should write in plain English. The "NLP" engine will then process and automate the individual steps ofa test plan written in English. Thousands of tests are performed in a few minutes with thistool [16].
- "TestCraft". Also used to monitor web applications. The role of AI technology is to automatically handle changes in the application, eliminating service life and costs. The biggest advantages of "TestCraft" are that testers are automated using a drag-and-drop interface, can create selenium-based tests, and run simultaneously in multiple browsers andworkspaces. Tests can be run in parallel in different environments and in differentbrowsers. No coding skills are required to perform this test [16].

THE POTENSIAL IMPACT OF AI ON SOFTWARE TESTING

Based on our research, we can conclude that the ongoing trends in artificial intelligence in thefield of software testing are very promising and that AI will play a more important role in the development of this field in the future. It is not a coincidence that companies have already startedto invest in this sector. Following Fig. 2 demonstrates the potential impact of AI on software testing. In the first phase of the development path, AI will take on automation tasks that require decisions that a person can make in less than a second. In the second phase, resolution of higher- level test tasks may still require human intervention or intervention. Tasks such as "Test creation", "Usage testing", "Security testing" and "External cases" require more thought. Third, as technologyevolves and machines learn these high-level tasks on the go, AI will be able to handle these activities and tackle problems that require a deeper context.

Significant contributions to artificial intelligence in the field of software testing in the near future(4 to 8 years) based on research, analysis and prediction are [17]:

- Artificial intelligence software testing will become an independent industry and play an important role in the field of information technology. Artificial intelligence is expected to replace quality assurance and test engineers. Quality assurance professionals and test engineers will play a new role in organizing and monitoring results.
- Artificial Intelligence will manage the software testing process and will cover all stages oftesting without human intervention and error, from test preparation to planning, executionand reporting.

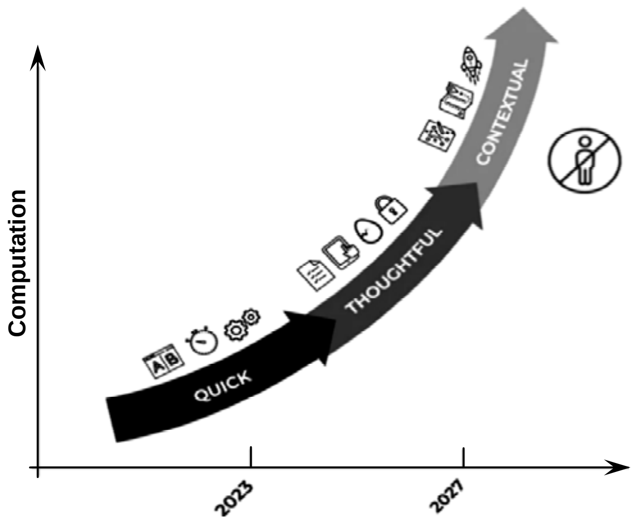

Fig. 2. Development of AI in software testing in the near future

- The application of artificial intelligence will provide more accurate results in the testing field than traditional testing methods and shorten the development time of the program. Especially since it is difficult to meet software requirements and meet deadlines, artificialintelligence is expected to fill this gap and alleviate the problem by reducing the required trial time.
- AI will eventually have specialized tools to effectively test "Cloud Computing", "Big Data" and other future technologies. The introduction of new technologies will also bringinnovation to software testing based on artificial intelligence. Because artificial intelligence will play an integrative role in generating the test data required for a particularproduct.
- AI can apply predictive analytics by examining other similar products and services to betterunderstand the new features customers need.
- Organizations will use artificial intelligence algorithms and techniques to improve their product offerings and improve the quality of services provided.
- The predictive analytics of artificial intelligence will play an important role in detecting allpossible test scenarios, making software products more robust, reliable and exceeding customer expectations.
- New AI-enabled software testing tools will be innovative, flexible and smart. This will allow users to get more accurate results.
- "Machine learning", "Deep learning", "NLP" and other fields of artificial intelligence are considered pioneers of most of the technologies around us. As mentioned earlier, the application of artificial intelligence is automated. It will demonstrate the great power of software testing and improve the software development and testing process in a new era ofinnovation and agility.

CONCLUSION

At a time of increasing demand for quality software testing, it is important to stimulate researchin this area, continually recap new achievements, and come up with different ideas to facilitate rapid development. Studying and successfully using the methods used in this area will help testersto perform testing more effectively and thus improve the quality of the program. Obviously, delivering quality software is the ultimate goal of any software project. We cannot be sure of the quality level in a program without measurement. Therefore, various test methods are used to measure quality. The article examines different testing techniques that can be used to measure different quality characteristics. We concluded that artificial intelligence is the most widely used method for black box testing, and all three machine learning methods (controlled, uncontrolled and augmented), namely "clustering" are widely used.

In addition, the result is expected to be an integral part of artificial intelligence software testing.Testers will play a new role in making AI models and algorithmic methods smarter. Artificial intelligence has the ability to automatically analyze complex data using intelligent models and algorithms. Thus, artificial intelligence will manage most of the test areas and provide more accurate results. Automating software testing will improve the quality of software and have a huge impact on customer experience by delivering seamless applications and solutions. At the same time, artificial intelligence algorithms will be applied to new technologies (Cloud Technology, BigData) and better results will be obtained.

REFERENCES

1. A. Dennis, B.H. Wixom, and D. Tegarden, *Systems Analysis and Design with OOP Approach with UML 2.0*; 4th edition. USA: John Wiley &amp; Sons, Inc., 2009, 691 p.
2. G.J. Myers, C. Sandler, and T. Badgett, *The Art of Software Testing*; 3rd edition. Canada: John Wiley &amp; Sons, Inc., 2012, 420 p.
3. A. Dennis, B.H. Wixom, and R.M. Roth., *Systems Analysis and Design*; 5th edition. USA: John Wiley &amp; Sons, Inc., 2012, 594 p.
4. D.R. Graham, "Testing, Verification and Validation", *Int. J.*, vol. XVI, pp. 1069–1101, 1979.
5. D. Huizinga and A. Kolawa, *Automated defect prevention*. Hoboken, N.J.: Wiley-Interscience, 2007, 454 p.
6. M. Polo, P. Reales, M. Piattini, and C. Ebert, "Test automation", *IEEE Software*, vol. 30, no. 1, pp. 84–89, 2013.
7. M.A. Umar, "Comprehensive study of software testing: Categories, levels, techniques and types", *International Journal of Advance Research, Ideas and Innovations in Technology*, vol. 5, no. 6, pp. 32–40, 2019.
8. U. Mubarak Albarka, and C. Zhanfang, "A Study of Automated Software Testing: Automation Tools and Frameworks", *International Journal of Computer Science Engineering (IJCSE)*, vol. 8, pp. 217–225, 2019.
9. L. Luo, *A Report on Software Testing Techniques*. Pittsburgh, USA, 2003, 20 p.
10. O. Carol, "Why We Need New Software Testing Technologies", *Conference At-A-Glance PNSQC World Trade Center 121 SW Salmon St. Portland*, pp. 226–248, 2019.
11. I. Jovanovic, "Software Testing Methods and Techniques", *IPSI BgD Trans. Internet Res.*, vol. 5, no. 1, pp. 30–41, 2009.
12. E. Khan, "Different Forms of Software Testing Techniques for Finding Errors", *Int. J. Comput. Sci.*, vol. 7, no. 3, pp. 11–16, 2010.
13. M.E. Khan and F. Khan, "A comparative study of white box, black box and grey box testing techniques", *International Journal of Advanced Computer Science and Applications*, vol. 3, no. 6, pp. 12–15, 2012.

14. R.S. Pressman, *Software Engineering: A Practitioner's Approach*; 6th edition. Chapter 14: Software Testing Techniques, &amp; Associates, Inc., 2005, 402 p.
15. F. Redmill, "Theory and Practice of Risk-based Testing", *Software testing*, vol. 15, no. 1, pp. 3–20, 2005.
16. R. Lima, A.M. Cruz, and J. Ribeiro, "Artificial Intelligence Applied to Software Testing: A Literature Review", *15th Iberian Conference on Information Systems and Technologies (CISTI)*, pp. 6–7, 2020. Available: https://doi.org/10.23919/cisti49556.2020. 9141124.
17. J. Wang and C. Zhang, "Software reliability prediction using a deep learning model based on the RNN encoder–decoder", *Reliability Engineering & System Safety*, vol. 170, pp. 73–82, 2018.
18. H.L.P. Raj and K. Chandrasekaran, "NEAT Algorithm for Testsuite generation in Automated Software Testing", *IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 2361–2368, 2018.
19. A.D. Danilov and V.M. Mugatin, "Solving the optimization problem for regression testing using a neural network approach", *Modeling, optimization and information technology*, **8** (1), pp. 1–8, 2020.
20. I.S. Polevshikov and R.A. Faizrakhmanov, "Automated control of testing software systems using neural networks", *Engineering Bulletin of Don*, no. 4 (51), pp. 94–105, 2018.
21. M. Hossain, S. Abufardeh, and S. Kumar, "Frameworks for Performing on Cloud Automated Software Testing Using Swarm Intelligence Algorithm: Brief Survey", *Advances in Science, Technology and Engineering Systems Journal*, vol. 3, no. 2, pp. 252–256, 2018.
22. H. Li and P.L. Chiou, "Software Test Data Generation using Ant Colony Optimization", *International conference on computational intelligence*, pp. 1–4, 2004.
23. A. Tripathi, Sh. Srivastava, H. Mittal, Sh. Sinha, and V. Yadav, "Multi-Objective ANT Lion Optimization Algorithm Based Mutant Test Case Selection for Regression Testing", *Journal of Scientific & Industrial Research*, vol. 80, pp. 582–592, 2021.
24. K. Karnavel and J. Santhoshkumar, "Automated Software Testing for Application Maintenance by using Bee Colony Optimization algorithms (BCO)", *Information Communication and Embedded Systems (ICICES), 2013 International Conference on IEEE*, pp. 327–330, 2013.
25. I. Zarembo, "Analysis of artificial intelligence applications for automated testing of video games", *Environment Technologies Resources Proceedings of the International Scientific and Practical Conference*, pp.170–174, 2019.
26. M. Mikael, "Utilizing Artificial Intelligence in Software Testing", *Metrolopolia University of Applied Science*, pp. 46–47, 2019. Available: http://urn.fi/URN:NBN:fi:amk-2019120123754
27. Eggplant Software. Available: https://www.eggplantsoftware.com/
28. A. Jones, *Artificial Intelligence Tools for Software Testing*. Available: https://www.rtinsights.com/artificial-intelligence-tools-for-software-testing
29. R. Subramanian, "How AI is transforming software testing", *Selenium Conference, Chicago, 2018*. Available: https://youtu.be/pMd1L1IZrxk
30. *5 Popular AI-powered tools for test automation*. Available: https://www. nextgenerationautomation.com/post/ai-powered-tools
31. H. Hourani, A. Hammad, and M. Lafi, "The Impact of Artificial Intelligence on Software Testing", *IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT)*, pp. 568–569, 2019. Available: https://doi.org/10.1109/ jeeit.2019.8717439

Received 30.09.2021

INFORMATION ON THE ARTICLE

Tofig H. Kazimov, ORCID: 0000-0001-9245-6731, Institute of İnformation Technology of ANAS, Department of Software Engineering, Azerbaijan, e-mail: tofig@mail.ru

Tamilla A. Bayramova, ORCID: 0000-0002-8377-3572, Institute of Information Technology of Azerbaijan National Academy of Sciences, Department of Software Engineering, Azerbaijan, e-mail: toma_b66@mail.ru

Nazakat J. Malikova, ORCID: 0000-0001-9617-0554, Institute of İnformation Technology of ANAS, Department of Software Engineering, Azerbaijan, e-mail: naranara_68@mail.ru

ИССЛЕДОВАНИЕ ИНТЕЛЛЕКТУАЛЬНЫХ МЕТОДОВ ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ / Т.Г. Кязимов, Т.А. Байрамова, Н. Дж. Меликова

Аннотация. Рассмотрено несколько методов и инструментов, используемых в процессе автоматизированного тестирования программного обеспечения с учетом постоянно растущей важности тестирования программного обеспечения; обсуждены некоторые возможные последствия внедрения искусственного интеллекта в этой области. Основная цель исследования — изучить область автоматизации тестирования путем категоризации связанных тестовых действий, к которым могут быть применены инструменты искусственного интеллекта для повышения эффективности, и оценивания влияния приложения. Основные методы тестирования программного обеспечения — методы белого ящика, черного ящика и серого ящика; сделана попытка установить связь между данными методами тестирования и методами искусственного интеллекта. Предоставлено краткое описание нескольких инструментов двигателя искусственного интеллекта, используемых для автоматизации тестирования. Изучены возможные будущие выгоды от использования искусственного интеллекта при тестировании программного обеспечения.

Ключевые слова: тестирование программного обеспечения, автоматизированное тестирование, искусственный интеллект.

ДОСЛІДЖЕННЯ ІНТЕЛЕКТУАЛЬНИХ МЕТОДІВ ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ / Т.Г. Кязимов, Т.А. Байрамова, Н.Дж. Меликова

Анотація. Розглянуто кілька методів та інструментів, які використовуються у процесі автоматизованого тестування програмного забезпечення. З огляду на постійно зростаючу важливість тестування програмного забезпечення обговорено декілька можливих наслідків запровадження штучного інтелекту в цій сфері. Основна мета дослідження — вивчення галузі автоматизації тестування шляхом категоризації пов'язаних тестових дій, до яких можуть бути застосовані інструменти штучного інтелекту для підвищення ефективності, і оцінювання впливу програми. Основні методи тестування програмного забезпечення — методи білого ящика, чорного ящика і сірого ящика; зроблено спробу встановити зв'язок між цими методами тестування і методами штучного інтелекту. Подано короткий опис декількох інструментів двигуна штучного інтелекту, які використовуються для автоматизації тестування. Вивчено можливі майбутні вигоди від використання штучного інтелекту для тестування програмного забезпечення.

Ключові слова: тестування програмного забезпечення, автоматизоване тестування, штучний інтелект.