Artificial Intelligence in Software Test Automation: A Systematic Literature Review

Dhaya Sindhu Battina

Sr. Data Engineer & Department of Information Technology

CA, USA

Abstract— *The main aim of this paper was to review how artificial intelligence works in software test automation. When it comes to software engineering, artificial intelligence (AI) has had a significant influence, and software testing is no exception. With artificial intelligence (AI), the goal of software test automation may be closer than ever before. To some extent, the paradigm has changed during the previous two decades [1]. Everything about the testing process has been a positive experience, starting with manual testing and progressing to automated testing, where Selenium is acknowledged to be one of the best test automation tools. As a result, in today's high-speed IT landscape software testing must come up with fresh testing approaches that are based on solid research. The emergence of AI-based testing has been very beneficial for this aim [1]. A computer's ability to learn without human involvement may be fully simulated by AI algorithms and machine learning (ML). While AI and ML entail the construction of distinct and unique algorithms to access data and learn from it by identifying patterns to make conclusions, these predictions are intended to be employed in software testing to their full potential [1].*

Keywords: *Artificial intelligence, automation, Software test automation, software engineering, AI systems*

I. INTRODUCTION

The significance of technology in our professional and personal life constantly changes at a breakneck rate, and we must keep up with it. The digital revolution is now affecting every part of life, from household appliances to virtual reality headsets. To an international audience, companies create applications utilized by hundreds of thousands or perhaps million worldwide [2]. The agile quick delivery methodology is used by the vast majority of those companies, resulting in fresh launches every two weeks on average. These programs must be thoroughly tested before each launch to provide the best possible experience for the end-user. Manual testing cannot keep up at that rate. Every company, no matter how big or little, views software and application testing as a critical phase in the development cycle. There are several important components of a program that are validated by this process [2]. To be sure, manual testing gets more inefficient, time-consuming, and expensive as software expands and new functionality is introduced. Test automation, which automates critical processes & operations in detail to boost the quality & effectiveness of human testers, is increasingly being added to prevent such concerns.

In software development, the usage of artificial intelligence (AI) is still in its development, and the amount of autonomy is still considerably lower than observed in more mature fields of work including such self-driving systems or voice-assisted control, but it is still moving towards autonomous testing. In software testing tools, AI is being utilized to make the software development lifecycle simpler for the team working on the product. In software development and testing, artificial intelligence (AI) may be used to automate and minimize the number of dull and laborious operations that must be performed manually [2,3]. We must make certain that the test is always performed with an empty cart before adding any products. This helps to avoid distorted findings and adheres to sound automation principles. "Maintenance" is the largest issue with test automation. As software complexity rises, we must write more tests to keep up. It's because of this that we're swamped with testing and maintenance. It takes a lot of time and effort to debug and resolve tests that fail. Recent research shows that maintaining tests takes roughly 40% of the time spent by testers [4].

II. PROBLEM STATEMENT

The main problem that this paper will address is to review artificial intelligence in software test automation. Automated software testing has a major difficulty, which will be examined in detail in this paper. As a result of a lack of intelligence and premature human involvement, today's technologies are forced to cope with ineffective test runs [5]. As a result, there will be no way to identify test mistakes, code flaws, or other important obstacles in the testing environment. When it comes to testing, artificial intelligence makes it possible for users to move over their current challenges and improve their productivity. Instead of prioritizing the feature's testing, most teams just assign that task to whoever happens to be available at the moment [5]. To test software effectively, we need testers that are inquisitive about the product and have a critical perspective. These testers should question the product and assess it objectively.

III. LITERATURE REVIEW

A. An Overview of Artificial Intelligence for Software Testing

Software testing is a critical step in ensuring the application's customers are satisfied. Test automation is carried out in a controlled manner in which an application is monitored under specified situations, allowing testers to gauge the threshold and potential dangers associated with the software's deployment [6]. In software testing, artificial intelligence (AI) aids in the prevention of application fail-overs that might be costly to both the program and the company in the long run. Tests using Artificial Intelligence are becoming more important as AI becomes increasingly prevalent in our daily lives. Even with autonomous driving automobiles, there is still the risk of human lives being endangered if the car's intelligence isn't working correctly and it makes the incorrect judgment or responds too slowly [6]. An introduction to the advantages and requirements of

artificial intelligence in software testing will be provided in this article. Robotic testing is in charge of carrying out routine activities more accurately and quickly.

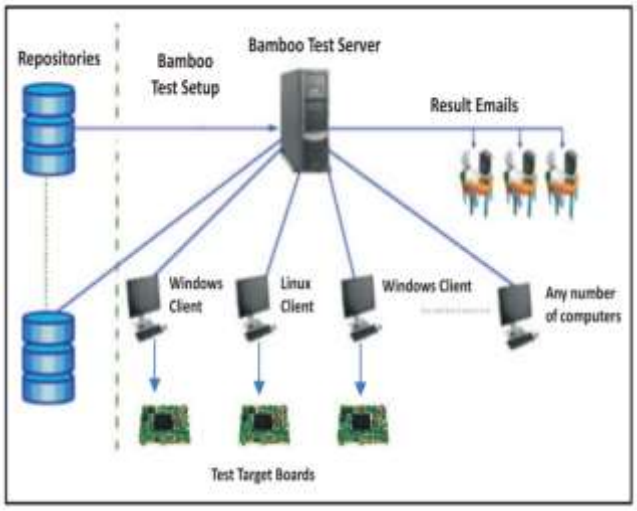

Figure I: Framework for software test automation

B. How is Artificial Intelligence Shaping the Dynamics of Software Testing?

We're leaning more and more on AI to make the application more secure (AI). We may be handing over much of the testing to AI as it becomes more automated. So, instead of human-driven testing, we're heading towards a situation where robots execute test scripts in place of people [6]. Machine learning and self-improvement will need some human input, but it will be minimal. Hence, the creation of a group focused on the Grand Dream of Testing has become critical, where everything is automated without human interaction and technologies provide superior testing than existing application test teams [6,7]. Consider going one step further and imagining an environment where software can test, diagnose, and cure itself on its own.

C. Artificial Intelligence-enhanced Software tools

Model-Inference driven testing (MINTest) is used for software test automation and may be used to generate test cases using the C4.5 method. It describes itself as a framework for unit and integration testing on its website [7]. It is implemented for the Linux operating system (OS). Using AutoBlackTest (Automated Black-Box Testing), the supervised learning method known as QLearning is implemented. The tool's primary purpose is to generate GUI test cases automatically. GitHub says it's only compatible with IBM Rational Functional Tester running on Windows [7,8]. It's impossible to determine if the program works with Windows OS versions higher than 8.1 and JRE based on the evidence currently accessible.

AimDroid is a GUI testing platform for Android apps created by Google. Exploration of the app's activity is used for automated testing. Using the program, tests may be run and results reported to the user. Fusing was employed as an AI enhancement [8]. One of AimDroid's drawbacks and concerns is that the smartphone must be rooted: the user of the device is given root access.

To fix GUI test breakage, Vista makes use of computer vision technology from the past. A successful test is recorded in the web-based GUI. It is possible for Vista to restore test scripts that fail on a subsequent version of an application by comparing their current status with what was documented before. In specific Selenium scripts, the program presently supports repairing Java scripts. To write GUI tests, you may use the Sikuli Test, an algorithm that is efficient that lets you use the visual notation (such as an image of an element to help identify it on the screen) while utilizing visual notation. The program makes automated testing easy for the users by using computer vision. Sikuli Test was created to run on any operating system. Because of this, it may be used to test personal computers, online, and mobile (Android) apps [8]. The tool seems to be actively being developed as SikuliX at the moment.

Testilizer can generate test scenarios for software applications leveraging SVM from Selenium scripts that are already in existence. It starts with Selenium tests and may create additional test cases for application states that haven't been reached yet. It's necessary to have Crawljax installed before running any tests on that system [9]. Automating Android GUI tests is made easier using SwiftHand's GUI test automation features. This approach is used to explore the model of the graphical user interface (GUI) of the program under test [10]. SwiftHand then makes use of it to produce the necessary inputs for inspecting the software's previously unvisited levels. SwiftHand is compatible with Linux and Mac. The tool's GitHub source has comprehensive installation and uses instructions [11].

D. The Importance of Artificial Intelligence in the Software Testing Process

In the field of software development, software testing is a critical step in the process. Nevertheless, due to the lack of resources and time, programmers are often unable to perform comprehensive testing (a test technique in which all conceivable data configurations are tested) on applications. To automate recurring patterns, we want a system capable of intelligently recognizing areas that will be developed and more concentrated. The greatest time, money, and resources go into software testing. In addition, developers are looking for speedier releases, and Artificial Intelligence is a good solution [12]. A human tester adds unnecessary expenses and energy since 80 percent of testing consists of repeating tests that the software already has. Artificial Intelligence may assist automate these procedures more effectively. To discover application difficulties, it would be a good practice to use both cognitive abilities and AI automation to create unique and novel software systems. Repetitive labor should be automated using Artificial Intelligence, such that just 80% of testing processes need human creativity and thinking [12].

When it comes to creating smarter and more efficient applications for end-users, artificial intelligence algorithms may be a huge assist in the testing sector. However, understanding how Artificial Intelligence (AI) may be used wonderfully is critical. Artificial intelligence algorithms that access automation in the same way as an actual human would. The next step is to identify the parts of the process where Artificial Intelligence may improve performance, and then use a machine learning or learning techniques algorithm to do so [13]. An algorithm that encourages the process, aids testers in finding the most problems in the shortest time, and increases the reliability and accuracy of the application is a plus [13]. Developers may then utilize the results to iterate on the product and learn from their mistakes.

E. The advantages of using artificial intelligence in software testing

Fig i: The benefits of integrating Artificial Intelligence in Software Testing

i. Enhanced Precision

When doing frequent manual software testing, even the most seasoned tester is prone to make errors. To aid with this, automated software tests conduct the same or repeated tasks properly every time, ensuring that accurate findings are recorded every time. The time saved by automating manual testing allows the testers to work on more complex features and new automated tests.

ii. Beyond Manual Testing's Restrictions

It is almost hard for even the largest software development/quality assurance organizations to conduct controlled web application testing involving 1000+ users. One may mimic tens, hundreds, or thousands of virtual users using automated testing, and these people can then be combined with a web-based application, program, or network [13].

iii. Benefits for developers and testers.

The developers can detect issues faster by using shared automated tests before submitting them to the QA team. Tests may be performed periodically anytime the source code is modified, checked in, and failures can be reported to the team or the developer. These kinds of features provide developers more confidence and save them time at the same time.

iv. Increasing the total number of tests conducted

It's possible to do more tests with more depth and breadth thanks to automated software testing. Automated software testing may examine the contents of memory and files, as well as internal program modes and datasets, to evaluate whether or not the software is working as it should under certain conditions. Overall, software test automation allows for the execution of over 1000 distinct test cases in a single test run, providing coverage that is not feasible with human software testing [14].

v. Time and money saved equates to a quicker time to market

Manually performing such tests may be time- and money-consuming when software tests are redone after every modification in the source code [15]. However, once they've been written, automated tests may be reused indefinitely at no extra charge and a significantly faster rate. Software testing may be done in a matter of hours rather than days, saving time and money.

IV. FUTURE IN THE U.S

The automation of software testing is advancing rapidly in the United States. The software industry's long-term prospects. The usage of tools to assist with testing will be a part of test automation. According to an InfoWorld study, 88% of firms automate 50% or more of their tests, which results in quicker testing cycles, 71% higher test coverage, and 68% better problem detection. Many American organizations are aiming to expand their automated testing portfolios as Agile and DevOps adoption increases [15,16]. Test automation has grown by 85% in the last two years, according to app developer Magazine. Open Source software technologies freely accessible on the market today have contributed to this increase. As a decentralized architecture for resilient and flexible cybersecurity management, cybersecurity mesh is a term to remember. As a result of cybersecurity mesh, the perimeter may be refocused on securing people or things rather than just their physical location. System scalability and adaptability will be determined via this kind of testing. The goal of artificial intelligence (AI) in software testing is to make testing more intelligent and efficient. The use of artificial intelligence (AI) and machine learning (ML) helps automate and enhance testing [16]. Software testing using artificial intelligence (AI) saves time and allows teams to work on more difficult tasks, such as developing inventive new features. The phrase "Mobile First" and providing the user with a mobile platform via mobile internet, hybrid mobile apps were popular throughout the world around 5 years ago. Artificial intelligence (AI) is the newest fad (AI). In self-driving vehicles, voice recognition, machine vision, healthcare, finance, and now test automation, hardly a day goes by without someone or an article announcing some form of AI progress [17].

V. ECONOMIC BENEFITS

As AI advance at a fast rate in the US, the advantages to the economy will only grow. To address many technical issues in the fields of healthcare, driverless vehicles, search engines, predictive modeling, and a lot more testing, companies like Apple have begun spending more on AI. It has an impact on all businesses, large and small. By 2030, AI is anticipated to boost the world economy by a whopping $15.7 trillion. The application of artificial intelligence (AI) in software development will be one of the fastest-rising technologies in the industry in the next years. Quality assurance will benefit from AI-based technologies by automating manual work and speeding up sprints within the SDLC [17,18]. Software quality assurance. One of the main goals of the US software business is to reduce software development costs while also enhancing the quality of the software. Though computerized economies are becoming more complicated, so is the underlying software required to sustain them. Software applications are now measured in terms of millions of lines of code rather than tens of thousands of lines of code, as was formerly the case. Concerns about software quality have grown as software complexity has increased and the typical market living standards of many software products have decreased. To conduct a thorough investigation, two industrial groups were chosen: automobile and aerospace equipment makers, as well as banking and finance players and associated digital communications equipment producers.

VI. CONCLUSION

This study looked at how artificial intelligence may be incorporated into the software testing process using test automation. Overall, the purpose of this study was to generate interest in artificial intelligence (AI) as a viable tool for use in the software testing automation industry. A Systematic Literature Review (SLR) was carried out to achieve the objective. In the near future, even "Continuous Testing" would be unable to keep up with the smaller delivery cycle durations, increased technological

complexity, and growing rates of change. We are rapidly nearing this point. For robots, the Internet of Things, and other cutting-edge technologies, the testing evolution must be carried out to ensure the efficiency necessary. While working on the IoT and practically driving "self-driving" automobiles, we need to learn how to work smarter, not harder, to ensure quality in an age where software processes an unthinkable number of data points in real-time. Although artificial intelligence (AI) continues to advance, it is clear that simulating the human brain is a difficult undertaking. It's important to remember that applications are used by people, and the technical improvements being developed consider that. This helps to guarantee a high-quality product.

REFERENCES

1. S. Amaricai and R. Constantinescu, "Designing a Software Test Automation Framework", *Informatica Economica*, vol. 18, no. 12014, pp. 152-161, 2014.
2. D. Banerjee and K. Yu, "3D Face Authentication Software Test Automation", *IEEE Access*, vol. 8, pp. 46546-46558, 2020.
3. L. Damm and L. Lundberg, "Results from introducing component-level test automation and Test-Driven Development", *Journal of Systems and Software*, vol. 79, no. 7, pp. 1001-1014, 2006.
4. C. Jordan, F. Maurer, S. Lowenberg and J. Provost, "Framework for Flexible, Adaptive Support of Test Management by Means of Software Agents", *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2754-2761, 2019.
5. J. Kasurinen, O. Taipale and K. Smolander, "Software Test Automation in Practice: Empirical Observations", *Advances in Software Engineering*, vol. 2010, pp. 1-18, 2010.
6. D. Kumar and K. Mishra, "The Impacts of Test Automation on Software's Cost, Quality and Time to Market", *Procedia Computer Science*, vol. 79, pp. 8-15, 2016.
7. C. Rankin, "The Software Testing Automation Framework", *IBM Systems Journal*, vol. 41, no. 1, pp. 126-139, 2002.
8. A. Bertolino, H. Foster, J. Jenny Li and H. Zhu, "Special section on automation of software test", *Journal of Systems and Software*, vol. 86, no. 8, p. 1977, 2013.
9. V. Garousi and F. Elberzhager, "Test Automation: Not Just for Test Execution", *IEEE Software*, vol. 34, no. 2, pp. 90-96, 2017.
10. B. Green, "Software test automation", *ACM SIGSOFT Software Engineering Notes*, vol. 25, no. 3, pp. 66-66, 2000.
11. J. Hollingum, "Reflex puts its software capabilities to the test", *Assembly Automation*, vol. 7, no. 1, pp. 21-23, 1987.
12. D. Hutton, "Software Test Automation: Effective Use of Test Execution Tools20005Mark Fewster and Dorothy Graham. Software Test Automation: Effective Use of Test Execution Tools. Reading, MA: Addison Wesley Longman 1999. 592 pp., ISBN: ISBN 0-201-33140-3 $39.95 (£25 approx.)", *Kybernetes*, vol. 29, no. 3, pp. 392-398, 2000.
13. D. O'Shea, F. Ortin and K. Geary, "A virtualized test automation framework: A DellEMC case study of test automation practice", *Software: Practice and Experience*, vol. 49, no. 2, pp. 329-337, 2018.
14. J. Park and J. Choi, "Test Framework Development for Software Reliability Test using Formal Method", *International Journal of Software Engineering and Its Applications*, vol. 10, no. 8, pp. 151-158, 2016.
15. M. Polo, P. Reales, M. Piattini and C. Ebert, "Test Automation", *IEEE Software*, vol. 30, no. 1, pp. 84-89, 2013.
16. K. Wiklund, S. Eldh, D. Sundmark and K. Lundqvist, "Impediments for software test automation: A systematic literature review", *Software Testing, Verification and Reliability*, vol. 27, no. 8, p. e1639, 2017.
17. S. Jan, A. Javed and M. Majeed, "Quality Category Matrix to Ensure the Quality of Software Product", *International Journal of Computer and Communication Engineering*, pp. 175-178, 2012.
18. B. Kitchenham, "Towards a constructive quality model. Part 1: Software quality modelling, measurement and prediction", *Software Engineering Journal*, vol. 2, no. 4, p. 105, 1987.