IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

IMPACT OF ARTIFICIAL INTELLIGENCE IN THE FIELD OF AUTOMATION

[1] Shelvin Sunil Philip, [2]Vaishnav Mohan, [3] H Muhammed Firoz, [4]Ivin Varghese George
[1] Student, [2] Student, [3] Student, [4] Student
[1]PG Department of Computer Applications and AI,
[1]Saintgits College of Applied Sciences, Kottayam, Kerala, India

Abstract: The use of machine learning algorithms, natural language processing, and other AI technologies to automate processes and tasks that were previously carried out by humans is a rapidly expanding field in artificial intelligence (AI). This technology is revolutionising how businesses run by boosting productivity, cutting expenses, and enhancing accuracy. Customers' questions are handled by chatbots powered by AI, and massive amounts of data are analysed using algorithms to help businesses make better decisions. AI can automate routine manual tasks like testing, assembly, and quality control in the manufacturing industry. AI can optimise logistics and supply chain operations by forecasting demand, controlling inventory levels, and enhancing delivery times. The software testing community is turning to AI to fill the gap since Aim can check the code for errors and problems without involving humans and much more quickly than humans. Car automation is also growing in the field of AI. It has the potential to revolutionize the automotive industry by improving safety, reducing traffic congestion, and increasing fuel efficiency. So, Artificial Intelligence plays an important role in automation.

IndexTerms: Software testing, API automation, Car Automation, Artificial intelligence, machine learning.

I. IMPACT OF ARTIFICIAL INTELLIGENCE IN SOFTWARE TEST AUTOMATION

INTRODUCTION

Artificial intelligence plays important role in our daily life and with surrounding applications and systems. Huge amounts of data are created every day from many different sources that need to be monitored properly. These sources are necessary to be observed, report results and take action. When a complex software application is built, time becomes a critical factor to release applications that must be fully tested. AI is useful in software testing because it can produce more accurate results faster.

Software testing has frequently sagged behind software development. For periods, it was the poor man of the software development process. But Artificial Intelligence is allowing it to catch up and indeed take a lead over the rest of the SDLC.

The technique of employing software tools to run tests automatically is known as test automation. Test automation in software testing is writing and running test scripts, which are often written in a programming language. Functional testing, regression testing, performance testing, and load testing are just a few of the testing processes that can be automated.

There are many benefits when using artificial intelligence in software testing. They are speed, reusability, consistency, accuracy, coverage, and scalability.

To implement test automation, testers use specialized testing tools that can help create and manage test scripts. These tools can be either open-source or commercial, and they can support different types of testing frameworks and programming languages.

Another important point is that test automation is not a complete replacement for manual testing. The manual is still necessary for certain types of testing, such as exploratory testing, usability testing, and user acceptance testing. However, test automation can greatly improve the efficiency and effectiveness of software testing, reducing the time and cost required for testing while increasing the quality and reliability of the software.

Artificial Intelligence can simulate user behavior and load patterns to identify potential performance issues. This helps to ensure that the software can handle expected usage and load.

Artificial Intelligence can identify potential security vulnerabilities and recommend improvements to prevent security breaches.

Overall Artificial Intelligence is helping to make software testing more efficient, effective, and accurate. It is enabling teams to deliver high-quality software with fewer defects, reducing costs and improving customer satisfaction.

LITERATURE REVIEW

Defect Prediction

Defect prediction in software testing is the use of machine learning and statistical techniques to predict the likelihood of a software defect occurring in a given code or module. The goal of defect prediction is to identify potential issues before they occur, allowing developers to take proactive measures to prevent defects from being introduced into the software.

There are several methods used for defect prediction in software testing, including:

- Static Code Analysis: This is a technique that analyses the source code of a software application to identify potential defects. Static code analysis tools use predefined rules to identify issues such as coding standards violations, syntax errors, and potential security vulnerabilities.
- Data Mining: This technique uses historical data to identify patterns and relationships between different code metrics and defects. Data mining tools use machine learning algorithms to analyze the data and identify potential risk areas.
- Expert Review: This involves having an experienced developer review the code to identify potential issues. This method is time-consuming and expensive, but it can be effective for identifying complex defects that may not be easily detected by automated tools.
- Model-Based Testing: This involves creating a model of the software application and using it to generate test cases. Model-based testing tools can identify potential defects by analyzing the model and identifying areas where defects are likely to occur.

Test Case Prioritization

Several factors can be used to prioritize test cases in software testing, including:

- Business impact: Test cases that have a higher impact on the business, such as those related to critical functionality or high-traffic areas of the application, should be prioritized.
- Risk assessment: Test cases that have a higher risk of uncovering defects, such as those related to new features or recently modified code, should be prioritized.
- Execution time: Test cases that have a shorter execution time should be prioritized to ensure that testing is completed within the available time frame.
- Dependency: Test cases that are dependent on other test cases should be prioritized to ensure that they are executed in the correct order.
- Frequency of use: Test cases that are related to functionality that is used frequently by users should be prioritized to ensure that the most critical areas of the application are thoroughly tested.

Fault Localization

Artificial intelligence (AI) fault localization is the act of locating the precise parts or regions of an AI system that are to blame for a certain flaw or malfunction. The process of fault localization in AI typically involves the following steps:

- Data collection: Data related to the fault, such as inputs and outputs of the system, logs, and other relevant information, are collected to provide insights into the fault.
- Analysis of the data: The collected data is analyzed using various techniques, such as statistical analysis, machine learning, or visualization, to identify patterns or correlations related to the fault.
- Localization of the fault: Based on the analysis, the specific components or areas of the AI system that are responsible for the fault are identified.
- Verification and correction: Once the fault is localized, it can be verified and corrected. This could involve adjusting the algorithm, improving the training data, or modifying the system configuration to improve performance.

Test Case Generation

Test case generation using artificial intelligence (AI) is a rapidly growing area of software testing. AI-powered testing tools can analyze software code, identify potential test scenarios, and automatically generate test cases.

Testim's AI-powered testing engine can also update the test cases automatically if any changes are made to the software application's UI. This helps to ensure that the test cases remain up-to-date and relevant, reducing the manual effort required to maintain the test cases.

Mabl's AI-powered testing engine can also detect changes in the software application's UI and update the test cases automatically. The tool can also prioritize the test cases based on the potential impact on the software application's functionality and performance.

Test Execution

Test execution using artificial intelligence (AI) is another area where AI can be used to improve the software testing process. AI-powered testing tools can execute test cases automatically, analyze the test results, and identify potential issues.

AI-powered testing tools can execute test cases automatically, analyze the test results, and identify potential issues. The tool can simulate user actions and input data into the software application, just like a human tester would. This approach can help organizations reduce the time and effort required to perform software testing while ensuring the software application is thoroughly tested.

The tool can also adapt the test cases automatically based on the analysis of the test results.

METHODOLOGIES USED

Case Studies

The primary means the case study was done by taking some sample case studies done by 7 different vendors and companies. The study explores the challenges or goals which had before implementing test automation.

In the following example, it was taken case studies of 7 different companies and vendors to explore the challenges or goals that they had before implementing test automation.

Table 1.1 shows the case study.

COMPANY	CHALLENGES/GOALS	RESULTS
Motionsoft	Automating smoke test cases to increase test coverage. Improve the slow testing process (2000 manual tests that took weeks to test).	Execution of 3600 automated tests daily.
Optimizely	Regression testing coverage is slow and insufficient.	4x faster test runs. 86% less time writing and debugging tests. 40% increase in new feature coverage.
GoFundMe	Test failures are high and test frequency is slow.	30x faster test runs. 98-99% reduction in test failures. 50% increase in developers writing tests.
Siemens Software	There are a lot of codes. glitchy and sluggish codes. Several false negatives.	49% reduction in test code. 38% increase in productivity. 375% increase in test execution speed. 18 minutes saved per execution. 290+ person hours saved per quarter
PlanGrid	Codes that are difficult to maintain and are slow. requires extensive UI setup.	2000+ tests run daily. 4-minute average run duration (without parallelization). 20+ custom commands for test efficiency.
Lightstep	Complex QA process. Manual test processing is time-consuming	8-12k tests run daily. 48x faster deployment validation speed.
Slido	Manual testing is very slow. Migration issues with coding	100% coverage of crucial user stories. 24x increase in test execution. 10x increase in weekly releases. 85% test coverage for all features.

On observing table 1.1, automated testing is employed by businesses across a variety of industries and can significantly improve a company's quality assurance procedure. Selenium is an open-source tool for test automation. In 50% of the case studies mentioned, we observed a common trait: the businesses used or experimented with selenium at first.

Companies, however, decided to use a different test automation provider after learning that Selenium was either too difficult to use or ineffective for their requirements. The main challenge found when observing the table is that bad code, slow testing, etc. If a business has a difficult-to-use or poor testing system, developers tend to avoid or limit testing because it increases the required effort without clearly demonstrating a value.

Challenges Faced by the testing community when performing the automated test

1) Bad code: The effort involved in manual testing can be decreased with test automation. According to reports, test automation replaced 50% or more of manual testing in 46% of the cases where it was used. Moreover, quality enhancement is cited as the primary strategic driver by 55% of businesses looking to automate their testing processes.
2) Slow testing: In the time of agile development and CI/CD, slow testing presents a serious challenge. Manual testing is thought to take up 35% of the testing cycle. Slow testing lengthens the development process and limits the feedback that can be provided for each design build.

II.IMPACT OF ARTIFICIAL INTELLIGENCE IN CAR AUTOMATION

INTRODUCTION

Artificial Intelligence (AI) is rapidly changing the automotive industry, particularly in the area of car automation. With the help of AI it is possible to create cars that are capable of driving themselves, which has significantly improved the safety, convenience, and efficiency of vehicles on the road.

AI in car automation involves the use of advanced algorithms and machine learning models to enable vehicles to sense, interpret, and respond to their surroundings. This involves using a range of sensors, such as cameras, radar, and lidar, to collect data about the environment, which is then analyzed and used to make decisions about how the car should behave.

LITERATURE REVIEW

Car automation using AI has gained significant attention in recent years due to the potential benefits it can offer, including improved safety, increased fuel efficiency, and reduced traffic congestion. This literature review provides an overview of the current state of research in the field, highlighting the various AI technologies used in car automation and the benefits and challenges associated with the implementation of AI-driven automated vehicles.

One of the key AI technologies used in car automation is machine learning. Machine learning algorithms enable vehicles to learn from their environment and make decisions based on data collected from sensors such as cameras, radar, and lidar. Researchers have used deep learning algorithms to improve object detection and recognition, which is critical for safe autonomous driving. LiDAR and computer vision have also been used to improve perception and object detection in challenging environments.

However, there are also challenges associated with the implementation of car automation using AI. One of the primary challenges is ensuring the safety and reliability of autonomous vehicles. This requires developing robust systems for testing, validation, and verification of autonomous vehicles. There are also legal and ethical issues surrounding the use of autonomous vehicles, such as liability and accountability in the event of an accident.

REGRESSION TESTING

Regression testing in car automation using AI is an essential process that involves testing previously validated features after making changes to the code or adding new features. This testing ensures that the changes or additions have not caused any unintended consequences and that the system still performs as intended. In the context of car automation, regression testing helps to ensure the safety and reliability of autonomous vehicles.

One of the main challenges of implementing regression testing in car automation using AI is the complexity of the system. Autonomous vehicles rely on a vast array of sensors, algorithms, and decision-making processes that interact in complex ways. As a result, creating an effective suite of tests requires a thorough understanding of the system and its components.

Another challenge is ensuring that the tests remain relevant as the system evolves. As new features are added or changes are made, the suite of tests must be updated to ensure that they still cover all the critical functionality of the system. Additionally, new test scenarios may need to be added to cover new driving situations or conditions that were not previously considered.

METHODOLOGIES USED

STEP 1: DATA COLLECTION

The first step in car automation using AI is data collection. Autonomous vehicles rely on data from various sensors, including cameras, lidar, radar, and GPS, to perceive their environment and make decisions. The data collected from these sensors must be comprehensive, accurate, and diverse to train machine learning models effectively. Data collection involves recording the sensor data from a variety of driving scenarios, including different weather conditions, lighting conditions, and road configurations.

STEP 2: DATA ANNOTATION

Once the data has been collected, it needs to be annotated to label objects and events in the data. This is a crucial step in preparing the data for machine learning models. Data annotation involves labeling objects such as pedestrians, vehicles, and traffic signs, as well as events such as lane changes and turns. Annotation can be done manually or through semi-automatic methods such as active learning or crowdsourcing.

STEP 3: MODEL DEVELOPMENT

After data annotation, machine learning models are developed to learn from the labeled data. Deep learning algorithms, such as convolutional neural networks (CNNs), are commonly used for object detection and recognition. Other models, such as recurrent neural networks (RNNs) and decision trees, can be used for decision-making and control. The models are trained on the annotated data to learn how to perceive the environment and make decisions.

STEP 4: SIMULATION AND TESTING

Simulation and testing are critical steps in car automation using AI. Simulations are used to test the machine learning models and decision-making algorithms in a virtual environment before real-world testing. Simulations allow developers to create complex driving scenarios and test the system's performance under different conditions. Once the models and algorithms have been tested in simulation, real-world testing can begin. Real-world testing involves testing the autonomous vehicle on public roads and monitoring its performance.

STEP 5: CONTINUOUS LEARNING AND IMPROVEMENT

Continuous learning and improvement are crucial for the ongoing development of autonomous vehicles. The system must continually learn from new data and adapt to changing driving conditions. Machine learning models must be retrained with new data to improve their accuracy and performance. Additionally, new features and algorithms must be developed to handle new driving scenarios and conditions.

STEP 6: VALIDATION AND CERTIFICATION

Validation and certification are the final steps in car automation using AI. Validation involves testing the system against established safety and performance standards. Certification involves obtaining regulatory approval for the autonomous vehicle to operate on public roads. These steps ensure that the autonomous vehicle is safe and reliable and that it meets all legal and regulatory requirements.

Car automation using AI has gained significant attention in recent years due to the potential benefits it can offer, including improved safety, increased fuel efficiency, and reduced traffic congestion. This literature review provides an overview of the current state of research in the field, highlighting the various AI technologies used in car automation and the benefits and challenges associated with the implementation of AI-driven automated vehicles.

One of the key AI technologies used in car automation is machine learning. Machine learning algorithms enable vehicles to learn from their environment and make decisions based on data collected from sensors such as cameras, radar, and lidar. Researchers have used deep learning algorithms to improve object detection and recognition, which is critical for safe autonomous driving. LiDAR and computer vision have also been used to improve perception and object detection in challenging environments.

Another area of research is developing decision-making algorithms that enable autonomous vehicles to make complex decisions in real-time. Reinforcement learning is a popular approach for developing these algorithms, where the vehicle learns from trial and error by receiving rewards for good decisions and penalties for bad decisions. Researchers have also developed algorithms for predicting the behavior of other road users and incorporating this information into decision-making.

There are numerous benefits of car automation using AI, including improved safety, reduced traffic congestion, and increased fuel efficiency. Automated vehicles can significantly reduce the number of accidents caused by human error, as well as improve the flow of traffic by minimizing human-caused delays. Furthermore, by optimizing driving patterns, AI-driven vehicles can improve fuel efficiency and reduce emissions.

III.CONCLUSIONS

In conclusion, the application of AI to automation has the potential to completely alter how companies run their operations. AI can boost productivity, cut costs, and improve accuracy by automating processes and tasks that were previously handled by humans. Algorithms can analyse a tonne of data to improve decision-making, while AI-powered chatbots can handle customer questions and complaints. Artificial intelligence (AI) can optimise operations in logistics and supply chain management by forecasting demand and controlling inventory levels, while it can automate repetitive and manual tasks in the manufacturing sector. AI can be used to automate financial and accounting tasks, increasing efficiency and lowering error rates. In the field of software automation, it can help in predicting the impact of changes made to the software, by analyzing the code and predicting the possible outcomes. Car automation using AI involves a complex set of steps, including data collection, annotation, model development, simulation and testing, continuous learning and improvement, and validation and certification.

IV.REFERENCES

[1] "Automation" (1999) *Laboratory Automation & Information Management*, 34(3), pp. 233–258. Available at: https://doi.org/10.1016/s1381-141x(99)80009-4.

[2] Kaplan, J. (2016) "Defining artificial intelligence," *Artificial Intelligence* [Preprint]. Available at: https://doi.org/10.1093/wentk/9780190602383.003.0001.

[3] Olszewska, J. (2020) "AI-T: Software Testing Ontology for AI-based systems," *Proceedings of the 12th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management* [Preprint]. Available at: https://doi.org/10.5220/0010147902910298.

[4] Rowles, C.D. and Leckie, C. (1988) "A design automation system using explicit models of Design," *Engineering Applications of Artificial Intelligence*, 1(4), pp. 258–268. Available at: https://doi.org/10.1016/0952-1976(88)90044-9.

[5] “Annual International Conference on Control, automation and robotics (car 2011)” (2011) *PsycEXTRA Dataset* [Preprint]. Available at: https://doi.org/10.1037/e602432011-001.

[6] Winston, P.H. (1999) *Artificial Intelligence*. Addison-Wesley.

[7] *Artificial Intelligence (AI) in software testing* (2023) *Tricentis*. Available at: https://www.tricentis.com/learn/artificial-intelligence-software-testing (Accessed: March 19, 2023).

[8] *What is an autonomous car? – how self-driving cars work* (no date) *Synopsys*. Available at: https://www.synopsys.com/automotive/what-is-autonomous-car.html (Accessed: March 19, 2023).

[9] *chatbot: Ai chat bot software for your website* (no date) *ChatBot*. Available at: https://www.chatbot.com/ (Accessed: March 19, 2023).