INTERNATIONAL JOURNAL OF COMPUTER SCIENCE
Scholorly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
Since 2012
www.ijcsjournal.com Volume 14, Issue 1, No 02 2026 ISSN: 2348-6600
REFERENCE ID: IJCS-582 PAGE NO: 048-057

IMPLEMENTATION OF SOFTWARE TESTING USING MACHINE LEARNING: A SYSTEMATIC MAPPING STUDY

S.Vivitha
Student,
Department of Computer Science,
Rajeswari College of Arts and Science for Women,
Bommayapalayam, Tamil Nadu, India.

Abstract

Software testing is an essential part of the life cycle of software development, verifying the quality. Dependability of software products. The introduction of machine learning (ML) technologies has led to Increasing a desire to use machine learning techniques to be improve and mechanize different areas of software testing. This systematic mapping project intends to offer a detailed an outline of the existing at the moment of research and application in the domain of software testing with machine intelligence. In the study thoroughly evaluates a wide range of literature, including academic publications and conferences. Proceedings and industry reports will be used to identify and categorize existing techniques. Machine learning-based approaches and technologies for software testing. A mapping study classifies the ML techniques used, such as classification, clustering, and anomaly detection. Analyses their application in several testing activities, such as test case creation, test Execution and fault forecasting.

Furthermore, the mapping study examines the problems, advantages and trends connected to the execution of machine learning in the software testing. It identifies significant research gaps Identifies areas for further investigation. By combining and arranging existing knowledge, the systematic mapping study is a useful resource for researchers, practitioners, and business. Professionals looking for insights into the changing landscape of software testing through Integration of machine learning technologies. This study's findings add to a deeper understanding of the current status of the industry and lay the groundwork for future breakthroughs in the junction of software testing and machine learning.

Introduction

Historically, traditional software was used to solve scientific computation and data handling problems. Along the last several decades improved technologies have produced a significant development in software industry. The contemporary era of technology raises the relevance of software. Software's are Capable of solving ordinary life challenges. Software engineers have come a

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE
Scholorly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
Since 2012
www.ijcsjournal.com Volume 14, Issue 1, No 02 2026 ISSN: 2348-6600
REFERENCE ID: IJCS-582 PAGE NO: 048-057

long way from the beginning. Towards technical advancement. The conventional software planet goes through development, Testing, deployment, and implementation of software systems, but the route of this planet was disrupted by the new paradigm for software development. The new paradigm, Data Driven Software, utilizes the Data was cleansed and curated to tackle unsolved problems. Factors such as unrealistic requirements, cost Quality concerns can be the cause of software failure, resulting in budget overruns. [5]

Decades ago, it was expected that such a machine would be constructed, capable of Think, understand, and make decisions. AI has added a new level to computer software development, making a decade-long dream a reality. Now, the computer has the intelligence to something, humans should not be instructed what and how to do. Nowadays, the solution is supplied to the computer creates software, and the computer selects the data and model for development. Computers are Capable of handling a huge volume of data in less time. Data volume is rapidly expanding, leading to Fault, failure, and expense overruns. The amount of data generated digitally at the beginning of 2022 was 463 exabytes per day by 2025, according to researchers. [3]

Software testing is the method of discovering faults and errors by executing the software in a controlled environment. This software testing procedure takes a long time, is expensive, and complicated. In this in today's technological age, automated software testing is everyone's first choice in the sector regarding software engineering. Software testing evaluates and validates the fulfillment of the criteria. The quality and capability of the generated program were traced back to the requirements given. The demand for automated software testing grows as program complexity increases Software testing activities delve focuses on studying the behaviour of software systems to error/bugs.

Testing activities involve diverse and high-priced, hence a hands-on approach has occurred implemented to avoid the problems associated with manual software testing, automated software testing is recommended.

The machine learning algorithm is used for automating the software testing in order to overcome difficulties or Challenges include effort, money, and time. Machine learning has become the most abundant technology to test.

Software or to complete tasks using supplied or learn facts. Unfortunately, automated tests using machine learning has also revealed some susceptibility to deceptive findings, which leads to inaccuracy or tragic failure. In this situation, a question arises: is the employment of machine learning, correct? Suitable for automated testing of program features.

Various works have been provided in the field of Machine Learning and Software Testing, but the There is no general guiding principle for using effective learning methods for software. Testing is the underlying cause of the question about the implementation of machine learning technologies.

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE
Scholorly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
Since 2012
www.ijcsjournal.com Volume 14, Issue 1, No 02 2026 ISSN: 2348-6600
REFERENCE ID: IJCS-582 PAGE NO: 048-057

For automated software testing. In this study, we present a research strategy to systematically to advance, review research work in the domains of software testing and machine learning. Toward selecting the best effective machine learning approach to test the software application employing automation. [2]. Fault identification and rectification grow increasingly expensive in the final stages of software development.

Even the price can exceed 50% of the development expense. The only way to overcome them overheads is to test the software as soon as feasible. Software testing is a critical component of the Software development process. Software testing and quality control operations have taken up a significant the whole budget for software development. [1]

Figure 1 shows the relative cost of fixing defects during the software development phases

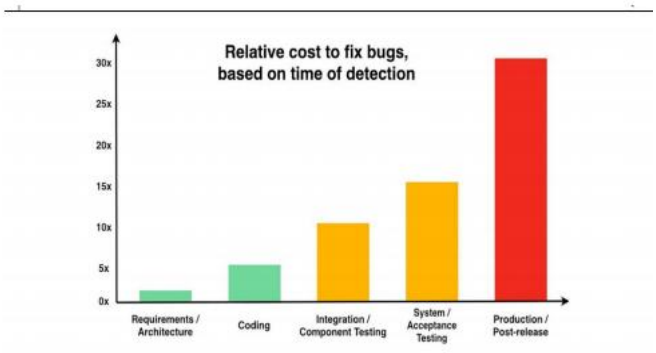


Figure 1: Relative cost to fix bugs in software development stages

Figure 1 depicts how the price of error elimination rises dramatically in the latter phases of software development. The primary goal is supplying customers with high-quality software is free of bugs.

Software development takes a long time, from traditional software to data-driven software. The complexity of software has expanded, and it is nearly difficult to deliver bug-free software. It can only extensive software testing can achieve this, which is expensive, time-consuming, and labour-intensive.

In some cases, software maintenance might account for up to two-thirds of the overall development cost cases. According to this statistic, automated testing procedures are expected to save money, effort, and time.

Software testing automation can increase software performance. Therefore, there is a requirement for developing automated software testing methodologies. Several attempts have previously been made to automate software testing.

Furthermore, the many testing techniques and approaches or automated testing, deploying an error Free software is similar to "a dream with open eyes". To deal with this problem, an investigation moving on to find a more effective method of software testing.

Artificial intelligence (AI) techniques have been effectively discarded to minimize software engineering activities. Machine learning (ML) is a sub-field of AI that integrates computer science and Statistics. It combined with AI have been utilized to a variety of software testing process.

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE
Scholorly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
Since 2012
www.ijcsjournal.com Volume 14, Issue 1, No 02 2026 ISSN: 2348-6600
REFERENCE ID: IJCS-582 PAGE NO: 048-057

All study demonstrates that machine learning techniques are capable of automating testing. Processes.

However, certain fundamental questions remain that must be addressed and investigated, such as:

- How effective are machine learning technologies at different stages of software testing?
- How does one assess the pros and shortcomings the studying approach the setting of the testing?
- How can we decide whether the given data inputs are possible to test the functional or not?
- What are the functional properties of the software?
- How does machine learning help uncover major software bugs?
- Can software testing using machine learning be as effective as traditional testing?

2. Objective

To discover the best appropriate machine learning method to test software products systematically. Review of studies in the areas of software testing and machine learning.

3. Literature Review

This section includes an exhaustive learning of multiple previous studies on approaches for features. Selection and classification of software fault prediction.

3.1 Software Fault Prediction with Feature Selection Approaches

Wang et al. discovered that the feature selection technique is critical to the model construction process. They include a non-linear component in the evaluation function to determine the most useful. Subset dimension is used to improve system performance. The selected characteristics were tested and validated using several machine learning methods, and the featured subset resulted in the significant increase in the classifier's accuracy. [2]

Reena and Rajan demonstrated that the invalidated version of a perfect subset based on the ideal approaches generate predictions about the under-development components of a computer system. The method of least spanning trees was used to decide whether qualities to exist. The outcome was examined following the modification of the algorithm to deal with different sets of data. [4]

Liu et al. consider that cost information is occasionally omitted throughout the categorization. They offer a novel to phase cost sensitive learning technique. The process of evaluating the proposed approaches used seven authentic NASA data sets. The findings prove after validating cost information by cost conscious feature selection and conventional cost blind feature selection, the TSCS approach outperformed the active single stage cost sensitive classifier. [19]

Xia et al. developed a method that used correlation analyses as part of the choosing relief features; three different classifiers were evaluated using a data set. These classifiers

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE
Scholorly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
Since 2012
www.ijcsjournal.com Volume 14, Issue 1, No 02 2026 ISSN: 2348-6600
REFERENCE ID: IJCS-582 PAGE NO: 048-057

were evaluated against two feature selection methodologies. The results of the ANOVA test indicate Implementing a "fusion method" based on"Relief F"and "Linear Correlation analysis"(Relief F-LC) may improve the ability to predict problems. [8]

Mandal and Ami demonstrate that the contribution of a collection of attributes is as significant as the ideal set in improving the performance of the error prediction model. They employed eight NASA datasets to compare this approach to two other working models, and they discovered that it resulted in a 54% increase in SDP, making the high promising of three. [6]

3.2 Classification Strategies for Software Failure Prediction

Dejaegar et al. examined and contrasted 15 Bayesian Networks (BN) and standardized the machine learning approach by incorporating them into the mainstream. They conduct a study to see whether the Markov is highlight selection guideline is effective. [10]
Bishnu and Bhattacherjee proposed a Quad Tree-based K Means methodology for predicting software module failures. They have created a method that is related to linking four trees for discovering faults in software by putting the clusters into the K-means algorithm. It demonstrated an increase in mistake detection rate in comparison to previous techniques. [7]

Catal et al.is developed an Eclipse-based coding fault prediction system for Java programs by Using Naive Bayes machine learning technique, we proved excellence in overcoming the issue of combining coding measurement with coding deficiency data by coders. Following that, a programmer created an Eclipse-based coding issue prediction model in an open-source the platform utilizes Naive Bayes. [11]

Malhotra and Jain created an error probability estimation model using Object-Oriented CK metrics and Quality Model for Object Oriented Design (QMOOD) measurements. Six distinct machine learning algorithms and a pragmatic methodology are utilized to create the prediction. To evaluate the model's correctness, a datasets from the under development open-source the source code of software. This confirmed that the random woodland and stowing approaches outperformed the other model. [17]

OkutanA and Yldz use Bayesian techniques to conclude the probabilistic causal relationship between coding metrics and error tendency. The authors utilized nine open-source data stores informative indexes to explore and determine that the best measurements are reply to the class, sequence of code and a poor quality coding. The study suggested that further extensive testing will be required for validation. [13]

Xing et al. employed a Support Vector Machine (SVM) is to categorize software components and predict Software quality is determined by its complexity metrics. The study shows that the SVM the prediction model outperforms previous active approaches for forecasting software quality. [15]

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE
Scholorly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
Since 2012
www.ijcsjournal.com Volume 14, Issue 1, No 02 2026 ISSN: 2348-6600
REFERENCE ID: IJCS-582 PAGE NO: 048-057

Kumar et al. used the minimum Squares Support Vector Machine (SVM) is a learning technique, together with the direct, polynomial, and dispersed foundation work in bit capabilities, to illustrate the building of an efficient fault prediction model through shape and analysis.

Coding metrics and the predictive power of a small number of standards. The researchers utilized them separate element selection to get the best fitting classification from a large collection of twenty.

Refine the studies based on criteria, collect authentic information, and narrow the research based on research questions.

- Data Selection
- Data Synthesis
- Data analysis

Research Results Investigate the major research source code metrics. The study found that the prediction model is useful for jobs but requires flawed classes. [16]

4. Mapping the Study Procedure

This segment defines the procedure that we proceeded to over the course of this organized mapping.This study was based on the guiding principle for directing secondary investigations suggested by Kitchen ham et al. and Petersen et al.

4.1 Research Questions

We focused the selection and classification of the literature to make it meaningful for the Researchers intend to employ ML for automated software testing.

Kitchenham et al. stated that the purpose of secondary investigations should be expressed by research questions. Our study is based on the following a research questions:

Q1: Can machine learning ensure software quality in the same way that traditional software testing does?

Q 2: What is the strength of ML when applied to software testing for research?

Q 3: What ML algorithms have been utilized to address issues raised during automated software testing?

Q 4: What do the researchers see and apply to automated software testing with machine learning?

Q 5: What problems do academics face when utilizing ML for automated software testing?

Methodology

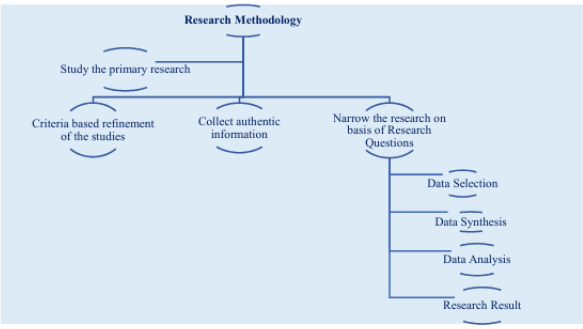
To obtain the answers to the research question, we researched literature from more than the last ten years. The investigations were refined based on a variety of parameters. The study of literature was focused on the basis for research questions. The study selection is based on those studies that contain the research results on ML methods of testing software, techniques of fault prediction, and ML algorithms. Which way to automated software testing. We gathered authentic

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE
Scholorly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
Since 2012
www.ijcsjournal.com Volume 14, Issue 1, No 02 2026 ISSN: 2348-6600
REFERENCE ID: IJCS-582 PAGE NO: 048-057

information from research published in reputable libraries like as IEEE, Springer, Elsevier, Wiley and ACM (conference proceedings, journals, and books), and tailored our search and study to our research concerns.

Following data collection, data extraction, synthesis, and analysis based on research the results were evaluated and formulated.

Figure 2 Illustrates the Whole Research Technique



Although the software paradigm for data-driven is quickly causing problems for typical software solutions, an organized the mapping investigation revealed extensive research into the excellent area of effective ML quality. Methodologies and procedures, there remains a space when compared to comprehensive Assurance of Quality operations. The research is entirely focused on increasing a software quality through the application of machine learning models and methods. In spite of the fact that this study identifies a research gap, it is possible that this systematic mapping analysis will reveal answers to research problems.

We modeled the following RQs and supplied solutions by analyzing the results of our organized mapping study:

Q1: Can machine learning ensure software quality in the same way that traditional software testing does?

According to our study, machine learning can ensure the quality of software testing by employing algorithms and producing new models are developed based on research findings.

Q 2: What is the strength of ML in the context of software testing for research?

According to our findings, ML programs have been used to solve the software testing problems. Since the 1970s (by Louridas and Ebert), although they have only recently gained attention from researchers. The interest was increased in the employing ML algorithms of software testing strengthens its performance.

Q 3: What ML algorithms are utilized to resolve concerns raised during automated software testing?

Various ways investigated in primary research to automate software testing utilizing machines learning algorithms. Our results reveal that in feature selection techniques TSCS algorithm Relief feature selection algorithm, the least spanning tree method, and the classification strategy for software fault The most commonly utilized prediction methods include K Means, QMOOD, and Naive Bayes.

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE
Scholorly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
Since 2012
www.ijcsjournal.com Volume 14, Issue 1, No 02 2026 ISSN: 2348-6600
REFERENCE ID: IJCS-582 PAGE NO: 048-057

Q 4: What observations are the researchers making and applying to automated software testing with machine learning?

We noticed that the community of researchers employs varied approaches, some are using some use upgraded or modified procedures, while others use hybrid approaches to resolve challenges.

Q 5: What problems do academics face when utilizing ML for automated software testing?

After analyzing our results, we felt that a tester must encounter the following challenges:

- **Data Availability and Quality:** Machine learning algorithms is to require a massive amounts of high-quality data is to train the model. It might be difficult to effectively train models for test automation when data is unavailable or of poor quality.
- **Complexity:** Interpreting the behavior of machine learning models for automation.
- Testing is tough since the ML techniques are sophisticated and difficult to comprehend.
- **Overfitting** occurs when a model is sophisticated or lacks enough data to train, and it performs. It works well on trained data but poorly on new data.
- **Machine learning** models must be maintained and monitored on a regular basis in order to retrain and update them for the system being tested.
- **Deployment concerns** include an unstable environment, malfunctioning code, and training-serving skew.
- **Integration:** Adding machine learning models to current test automation frameworks necessitates significant development effort.
- **Explain ability:** Some machine learning models may be too sophisticated to describe how they made their predictions.
- **Bias:** Because of insufficient data or prepossessing, the model may produce erroneous findings.

References

1. J. C. Westland, "The cost of errors in software development: Evidence from industry," Journal of Systems and Software, no. 62, no. 1, pp. 1-9, 2002.
2. Wang, J., & Zhang, C.: Software reliability prediction using a deep learning model based on RNN encoder/decoder. Reliab. Eng. Syst. Saf. 170, 73-82 (2018)
3. P. Ammann & J. Offutt, Introduction to Software Testing, 2nd Edition. Cambridge University Press. 2016.
4. Reena, P. and Rajan, B.: A new feature subset selection approach for software fault prediction. Int. J. Computing. Appl. 100(17), 39-43 (2014)
5. D. Zhang & J. J. Tsai, "Machine Learning and Software Engineering," Software Quality Journal, Vol. 11, no. 2, pp. 87-119, 2003.

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE
Scholorly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
Since 2012
www.ijcsjournal.com Volume 14, Issue 1, No 02 2026 ISSN: 2348-6600
REFERENCE ID: IJCS-582 PAGE NO: 048-057

6. Mandal, P., Ami, A.S.: Choosing the optimal features for software fault prediction. In the IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE), pp. 110-113. IEEE (2015)
7. Bishnu, P.S., Bhattacherjee, V.: Software defect prediction with quad tree-based k-means clustering algorithm. IEEE Transactions. Knowl. Data Engineering. 24(6), 1146-1150 (2012)
8. Xia, Y. et al.: A novel metric selection method for software fault prediction. In: International Conference on Progress in Informatics and Computing (PIC), pp. 433–436. IEEE (2014)
9. B. Lantz, Machine Learning with R, 2nd Edition. Packt Publishing. 2015.
10. Dejaeger, K., Verbraken, T., & Baesens, B.: Toward intelligible software fault prediction models with Bayesian network classifiers. IEEE Transactions. Software engineering. 39(2), 237-257 (2013)
11. Catal, C.: Software fault prediction: literature survey and current trends. Expert System. Appl. 38(4), 4626-4636 (2011)
12. K. Petersen, S. Vakkalanka & L. Kuzniarz's "Guidelines for Conducting Systematic Mapping Studies in Software Engineering: An Update," Information and Software Technology, volume. 64, pp. 1-18, 2015.
13. Okutan, A., Yıldız, O.T.: Using Bayesian networks to predict software defects. Empir. Software engineering. 19 (1), 154-181 (2014)
14. B. A. Kitchenham, D. Budgen and O. P. Brereton, "Using Mapping Studies as the Basis for Further Research--A Participant-Observer Case Study," Information and Software Technology, Vol. 53, no. 6, pp. 638-651, 2011.
15. Xing, F., Guo, P., and Lyu, M.R.: A Novel Method for Early Software Quality Prediction Based on Support vector machine. In: 16th IEEE International Symposium on Software Reliability Engineering (ISSRE 2005). p. 10. IEEE (2005)
16. Kumar, L., et al.: Effective fault prediction model created utilizing the least squares support vector. Machine (LSSVM). J. Syst. Software 137, 686–712 (2018)
17. R Malhotra and J Jain (2021): Predicting errors in object-oriented software using cost- IOP Conference: Sensitive classification. Series: Materials Science and Engineering 1022 (2021). 012112 IOP Publishing doi: 10.1088/1757-899X/1022/1/012112 https://en.wikipedia.org/wiki/Machine_learning
18. Altay Ataman 2023: Machine Learning for Test Automationhttps://research.aimultiple.com/machine-learning-test-automation/

INTERNATIONAL JOURNAL OF COMPUTER SCIENCE
Scholorly Peer Reviewed Research Journal - PRESS - OPEN ACCESS
ISSN: 2348-6600
Since 2012
www.ijcsjournal.com Volume 14, Issue 1, No 02 2026 ISSN: 2348-6600
REFERENCE ID: IJCS-582 PAGE NO: 048-057

19. Liu, Mingxia Miao, Linsong Zhang, and Daoqiang 2014/06/01 676 686 Two-Stage Cost-sensitive Learning for Software Defect Prediction, IEEE Transactions on Reliability 63, DO I-10.1109/TR.2014.2316951.