Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 2, 2026

A GEOMM FRAMEWORK FOR SOFTWARE TESTING TOOLS SELECTION

Hamza Zaheer [1], Muhammad Kaleem[2], Aneeqa Asghar [3], Asifa [4], Muhammad Azhar Mushtaq[*5], Humayun Salahuddin[6]

[1,2,3,4, *5] *Department of IT, Faculty of Computing & IT, University of Sargodha, Sargodha, 40100, Pakistan*
[6]*Department of Computer Science and IT NCBA&E, Multan, 66000 Punjab Pakistan*

[*5]azhar.mushtaq@uos.edu.pk

DOI: https://doi.org/10.5281/zenodo.21848173

Keywords

Article History
Received: 19 January 2026
Accepted: 24 February 2026
Published: 28 February 2026

Copyright @Author

Corresponding Author: *
Muhammad Azhar Mushtaq

Abstract

The process of software development involves many stages through which a software system is developed. In software development one of the most important parts of this life cycle process is software testing. The software systems need to be tested to ensure they are performing without any defects and bugs. There are large number of testing tools available for testing so now selecting the best testing tools is a complex task. This research aims to develop a framework for selection of software testing tools. The proposed framework GEOMM has two parts: (1) the first part is the criteria selection part of testing tools represented in the form of GEOMM taxonomy (2) the second part is the GEOMM selection methodology. GEOMM is based on geometric calculation for weightage criteria and minimum maximum technique is used for distance calculation. The GEOMM framework has been validated to ensure that the testing tool ranking is more reliable with high accuracy. This framework is providing accurate results and performance increase of 21% with a satisfactory response in decision making for software testing tools selection.

I. Introduction:

Software testing can be defined as a process to execute software system to identify the bugs, errors, and defects to avoid unexpected software systems behavior [1]. It ensures that the system meets the specified requirements when it is available to the end user. Therefore, software testing plays a very important and crucial part in the Software Development Life Cycle (SDLC) [2] because it helps to validate and verify software product functionalities. In SDLC the occurrence of errors are unavoidable [3]. There can be different modules within a software or a software can also be developed as a common one module. So, if a software is developed as a single module, then that module needs to be tested and if it's developed as different modules then each of them needs to be tested.

Software system is tested to ensure that it is providing good performance and achieving the functionality for which it is developed. The software testing needs to be carried out to ensure that software is meeting its performance, quality, and reliability in terms of providing the functionality. Software testing make sure that the modules are achieving their intended functionality to avoid future failures [4]. Software testing is important for quality assurance [5], early defects identification [6], defect-free software system [7], and is more than improving software systems [8]. The software testing phase should begin at the initial stage [9] when we develop even a single unit of a software system. Software testing must run from development to the maintenance of the software system. The software testing phase is very important in SDLC [10] because this phase consumes more time and cost as compared to other SDLC phases and is the main activity without which we cannot deliver software systems. Testing itself may not be a big problem but the main problem for testing any software system is deciding and choosing which testing tools is appropriate [11] and will support our testing criteria requirements. The previous framework using AHP and TOPSIS [12] provides the selection method for selecting software testing tools but the decision making for tool ranking need improvements and the framework has limitations that effect its accuracy and effectiveness. So, to overcome those drawbacks this research is about developing a software testing tools selection framework called the GEOMM framework. It will be important research within the field of software testing because this framework will contribute to the developers for selecting best appropriate testing tools with high accuracy. The software systems are becoming more complex [13] day by day and their functionalities are increasing. This framework will provide more structured and well-defined approach mechanism to select the best testing tools for software systems compared to the previous AHP and TOPSIS framework [12].

The GEOMM framework will reduce the cost for the testing process. Instead of randomly and repeatedly selecting different testing tools sometimes what happen is that the testing tools that are available are not free of cost but are paid ones [14]. The developers have to pay to use the functionality of those software testing tools but this framework will allow in advance to choose whether the testing tools to be considered are paid one or free of cost hence reducing the cost. With the GEOMM framework the test coverage of any software system will be increased because GEOMM framework has improved and well-defined taxonomy for the software testing tool criteria covering wide range of testing criteria that are considered when selecting a testing tool.

This paper remainder is structured in such a way that Section 2 discusses related work and the research gaps. Section 3 present the GEOMM framework for software testing tool selection. Section 4 present the performance evaluation of GEOMM framework. Finally, Section 5 include conclusion.

II. Related Work:

The choice of appropriate testing tool is very essential and selecting them is a very challenging task due to the variety they have. There are many types of software testing tools [15], [16], [17], [18] that are available to test the software systems. To conduct a software testing it involves test planning, designing the test, implementing the test, and importantly its execution. But to execute a testing of a software system then there are some attributes called the software quality attributes [19] that are important for defining software quality and for the selection of a software testing tools. These quality attribute should be considered to avoid their impact on taxonomy creation process. Some of the common quality attributes are security, maintainability, functionality, reliability, usability, portability, and efficiency. The number of criteria and quality attributes need to be organized in a well manner so that we can easily understand what features each testing tool support.

In Medical Image Analysis (MIA), the Convolution Neural Network (CNN) performance for classification and a federated learning strategy for data privacy protection is impressive. The CNN algorithm achieves an accuracy score of 0.90[20]. The taxonomy research results are not generated and the well detail of the testing tools is not present. A speed response comparison of the FOPID, KW-WOA-PID, SVR-PI, and FM-PI controller model simulation results indicates that the FM-PI speed controller guarantees better performance and displays an improvement in rising time and settling time, compared to other controllers[21]. The tools are classified according to the criteria they can support or deal with. The taxonomy is created to make selection of these testing tools easier but there is no selection method to choose testing tools. The

framework presented in [22] include a taxonomy that present the detail of the risks associated with different phase of the testing process. This taxonomy framework does not provide detail about which testing tools will be better and importantly no selection mechanism.
The previous framework with taxonomy for selecting the software testing tools using AHP and TOPSIS [12] has drawbacks and limitations. It does not handle the extreme values and the inconsistencies that occur. This framework for the testing tool selection will suffer if the criteria have different scales this will raise a big question mark on the testing tools recommendation that we get whether the recommendation is good enough or not. The outliner can affect the pairwise comparison matrix and decision making. If there are extreme values, it will dominate and results will be biased.
To deal with these challenges present in the previous framework [12] a new framework for selecting testing tools along with taxonomy called the GEOMM framework will be introduced in this research. The GEOMM framework has taxonomy which will be the improved version of the previous taxonomy [12] and different aspects are considered. This framework will be more robust to handle these issues and different approach will be utilized so that the framework can handle inconsistencies. The proposed framework will be able to handle complex and large numbers of criteria easily. The framework will make use of the scale approach for criteria so there will be a fair comparison when we scale out different criteria. With extreme values this framework will not be impacted. The GEOMM framework will provide more reliable and stable results with high accuracy of testing tools selection.

III. Proposed Framework:

This section will discuss about how GEOMM framework for software testing tool selection will actually work. As there are many testing tools available so to figure out which of the tool is best suited for the requirements then this framework will help us. This proposed framework has two parts shown in figure 1 and these two parts works together to achieve the functionality. The first part consists of the GEOMM taxonomy for the criteria of testing tools that the user can select, and the second part is where the GEOMM methodology is implemented. The two methods together provide the tool selection recommendations and make sure that the best suitable tool is recommended.

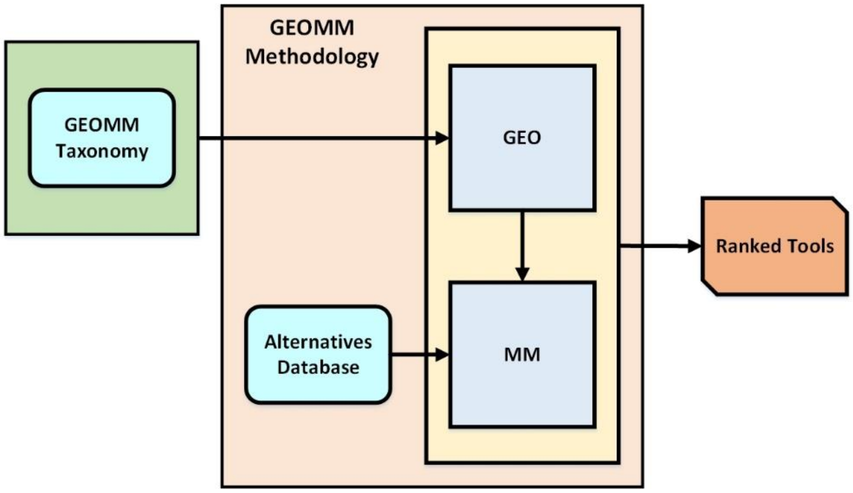

Figure 1. GEOMM framework approach

The proposed GEOMM framework mechanism has a proper structured approach. So, the tools that are recommended are not only rigid, but according to the specific task, and the specific need. Similarly, the efficiency of the tool selection will be improved because of the more accurate and efficient decision-making approach mechanism. The GEOMM framework will prioritize different testing tools on the basis of ranking. The GEOMM methodology will help us achieve improved performance. This framework will be helpful for many developers to automate their software testing tool selection task and eventually reducing the cost.

A) GEOMM Taxonomy:

The first and important part of this framework for selecting software testing tools is the well-defined taxonomy. GEOMM framework for software testing tool selection is going to use a taxonomy of testing tool criteria [12] that is further refined. The GEOMM taxonomy is actually the categorization or classification of the different criteria in such a way that it become more easier for the developers to choose their testing criteria. GEOMM taxonomy is refined and well organized in such a way that it overcome the issues related to classification because if such issues are present in the taxonomy they can affect our decision making. The GEOMM taxonomy is designed in more detail so that it covers the different criteria aspects which are important and to make sure that neither the criteria are overlapped nor they are misplaced. GEOMM taxonomy has three different scenarios that are to be reviewed to narrow down what actual criteria the developers are looking for the testing tools. Among these three scenarios the first scenario is the feasibility, the second scenario is examination, and the last scenario we have is the attribute scenario shown in figure 2. These are three different scenarios on the basis of which GEOMM framework taxonomy is organized. The feasibility scenario deals with tools functional, technical, and economical aspects. Examination scenario focuses on different testing aspects for the software system. Attribute scenario deals with quality standards attributes of an industry for testing the software system.

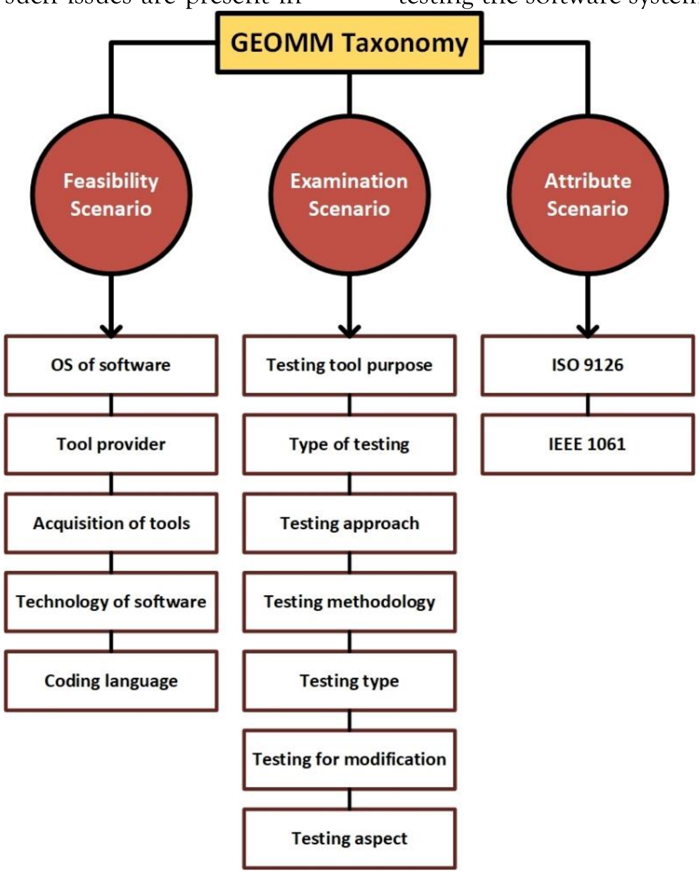

Figure 2. Three scenarios of GEOMM taxonomy

In GEOMM taxonomy there are three different taxonomy levels from level 1 to level 3 shown in table 1. Each of these levels is being further refined to make selection of the testing tools criteria easier. The taxonomy level 1 contains the three scenarios, taxonomy level 2 contains criteria factors that each of the three scenarios can support and the taxonomy level 3 contains selection criteria supported by taxonomy level 2 criteria factors. One should read the taxonomy from level 1 to level 3 to refine their criteria. So, the criteria selection must be done from the taxonomy level 3 which is the most refine level with more criteria clarity.

Table 1. GEOMM framework taxonomy.

Taxonomy Level 1	Taxonomy Level 2		Taxonomy Level 3
Feasibility Scenario	OS of software	OS	Android
			IOS
			Windows
			Linux/Unix
		IOT	Cloud testing
			Edge testing
			Fog testing
	Tool provider		Microsoft
			Parasoft
			IBM
			Neotys
			Katalon
			Micro Focus
	Acquisition of tools		Proprietary
			Open source
	Technology of software		Embedded system
			Web application
			Mobile application
			Object Oriented
			Database
			Open source
			Web service
			GUI
			Network protocol (TCP)
	Coding language		HTML
			PHP
			JAVA
			SQL
			Python
			C, C++, C#
Examination Scenario	Testing tool purpose		Test management
			Test design/ model-based testing
			Test data generator
			Test harness/test execution/ comparators
			Static analysis
			Debugging

		Configuration/ Change management
		Dynamic analysis/ performance testing/ monitoring
		Test evaluation/ coverage tools
		Security testing
	Type of testing	Static testing
		Dynamic testing
	Testing approach	Object oriented
		Service oriented architecture
		Aspect oriented
		V-Model
		Agile
		Test driven
	Testing methodology	Black box
		White box
		Gray box
	Testing type	Unit testing
		Integration testing
		System testing
		Acceptance testing
	Testing for modification	Regression testing
		Conformance testing
	Testing aspect	Functional
		Non-Functional
Attribute Scenario	ISO 9126	Maintainability
		Functionality
		Portability
	IEEE 1061	Reliability
		Efficiency
		Usability

IEER
Institute for Excellence in Education & Research

B) GEOMM Methodology:

The GEOMM methodology for testing tools selection is a combination of two improved Multi Criteria Decision Making (MCDM) methods. One of which is Analytic Hierarchy Process (AHP) and another one is Technique for Order Preference by Similarity to Ideal Solution (TOPSIS). The improved version of AHP with geometric mean is called GEO and the improved version of TOPSIS with minimum maximum normalization is called MM together the methodology is named GEOMM. The AHP and TOPSIS [23], [24] are one of the best and one of the popular methods through which we can carry out Multi Criteria Decision Making (MCDM).

The GEO can work with more complex scenarios and more complex hierarchies and with it the decisions are more highly subjective and more accurate. The MM can do more robust and more complex decision makings. So, to do more powerful decision making as compared to the standard AHP [25] and TOPSIS [26], [27] technique then these improved versions will be used. So, to get consistency in decision-making this methodology will do that. GEOMM methodology is designed in such a way that it will not only improve the accuracy but also the robustness in the decision making especially when the criteria have any specific characteristics and challenges.

ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 2, 2026

1. **GEO Technique:**

The GEO technique is more flexible and less sensitive to the inconsistencies that are available in the judgment of decision making. It can provide better performance instead of relying on the normal or standard AHP methods in which inconsistencies are encountered. This method will be less sensitive to the extreme values and the large inconsistencies. We can easily handle the inconsistencies in pairwise comparison matrix because of this approach. This approach will reduce the influence of outliers or extreme comparisons so that our results are more stable and the judgment is more accurate. This technique will also be effective if subjective bias is there to balance the judgment. The step-by-step process of GEO technique is shown in figure 3:

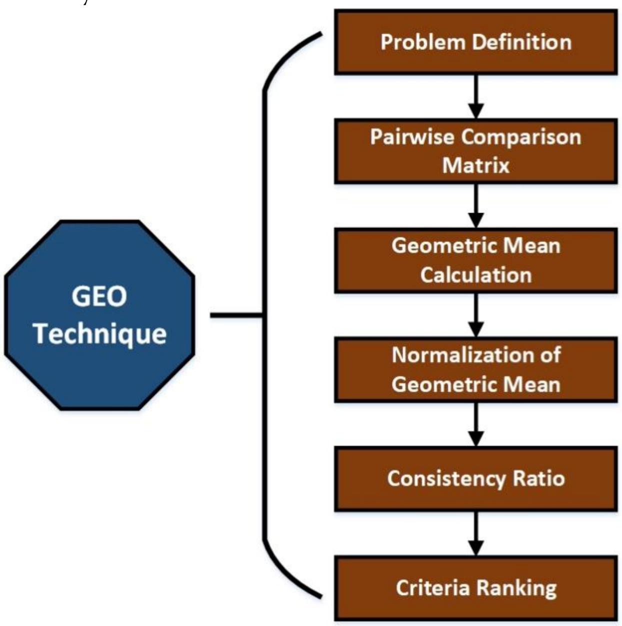

Figure 3. GEO technique process flow

First the problem is defined to identify goals, their criteria and their alternatives to be evaluated. Then structure is created for decision-making process so at the top is the goal for which we will work and then some criteria for this goal and sub-criteria and then alternatives. Next, pairwise comparison matrix is constructed for the criteria the values are assigned to each criterion. These values are not assigned randomly but assigned on the basis of the relative importance of each criterion with another criteria. These values actually tell which criteria is of more importance compared to other. The higher the number will tell which criterion has greater importance to us. The comparison matrix $\boldsymbol{A_{ij}}$ shows the comparison between the criteria $\boldsymbol{i}$ and $\boldsymbol{j}$. Once the pairwise comparison matrix is constructed then the geometric mean is calculated for each row of the pairwise comparison matrix. The geometric mean for a row GM_i is calculated as:

$$\boldsymbol{GM_i} = \left(\prod_{j=1}^{n} \boldsymbol{A_{ij}}\right)^{1/n}$$

Where $\boldsymbol{i}$ represent criteria and $\boldsymbol{A_{ij}}$ means comparison value in $\boldsymbol{i}$ rows and $\boldsymbol{j}$ column and $\boldsymbol{n}$ means the number of criteria we have. In fourth step geometric mean is normalized that is calculated for each row of the pairwise comparison matrix and to normalize it divide the calculated geometric mean of each row by total of geometric means of all in order to get each criteria relative weight. If $\boldsymbol{i}$ is the normalized weight for each of the criteria then:

$$\boldsymbol{W_i} = \frac{\boldsymbol{GM_i}}{\sum_{i=1}^{n} \boldsymbol{GM_i}}$$

The GEO technique will reduce the inconsistencies that affect the decision making but we will still calculate Consistency Ratio (CR) to make sure our decision making is more consistent and reliable. The CR is considered acceptable for reliable decision making if CR < 0.1 so the formula to measure the CR is as follow:

$$CR = \frac{CI}{RI}$$

Where:

$\boldsymbol{CR}$ Represent consistency ratio,

$\boldsymbol{CI}$ Is consistency index: $\boldsymbol{CI} = \frac{\lambda max - n}{n-1}$

$\boldsymbol{\lambda max}$ Is the comparison matrix maximum eigenvalue and $\boldsymbol{RI}$ Is random index depending on the size of the matrix and is pre-calculated. If CR value exceeds by more than 0.1, then it means that our consistency ratio is not acceptable and our pairwise matrix is also not stable, so it needs to be revised before moving to the next step. Finally, the criteria are ranked and each of them have their own weight from high to the low.

2. MM Technique:

The MM technique will enhance this framework performance together with GEO technique. Instead of different scaling values it will provide values of a common scale. This approach is more useful when the values of the different criteria are not consistent. So, MM technique will deal more better with scaling issues in our criteria data and will do better decision making. This technique will ensure that the values of such criteria will not dominate. With MM technique comparing alternatives becomes even more easier than standard TOPSIS technique. The step-by-step process of MM technique is shown in figure 4:

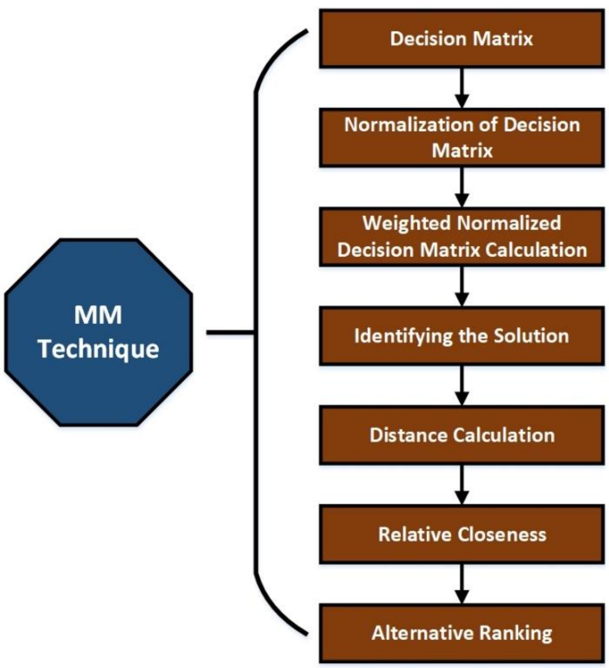

Figure 4. MM technique process flow

The first step in this approach is to construct a decision matrix this is the matrix in which each row is actually representing an alternative and each column is representing a criterion. This matrix is basically representing us how each alternative performs accordingly to the criteria. In the second step the decision matrix is normalized that is constructed based on the alternatives along with criterion through min-max normalization:

$$X'_{ij} = \frac{X_{ij} - X_{jmin}}{X_{jmax} - X_{jmin}}$$

Where:

$\boldsymbol{X_{ij}}$ Represent the matrix element showing performance of $\boldsymbol{i}$ alternative on $\boldsymbol{j}$ criteria,

$\boldsymbol{X'_{ij}}$ Is the normalized value and $\boldsymbol{X_{jmin}}$ represent minimum and $\boldsymbol{X_{jmax}}$ represent maximum value.

In the third step weight of the normalized decision matrix is calculated so for that we have to multiply our normalized values corresponding to their weights according to the criteria. So, the weighted normalized value is presented by V_{ij} and w_j represent the weight of criteria j:

$$V_{ij} = X'_{ij} \times w_j$$

The ideal positive or negative ideal solution is determined. The positive ideal solution is the best possible option and the negative ideal solution is the worst possible option:

$$V_j^+ = \max(V_{ij}), V_j^- = \min(V_{ij})$$

Next the distance to the positive ideal and negative ideal solution is calculated. For this Euclidean distance is calculated from each alternate option to ideal positive and negative solution:

$$D_i^+ = \sqrt{\sum_{j=1}^{m} (V_{ij}^+ - V_j^+)^2} \text{ , } D_i^- = \sqrt{\sum_{j=1}^{m} (V_{ij}^- - V_j^-)^2}$$

In the sixth step we calculate the relative closeness of an ideal solution where C_i represent the relative closeness:

$$C_i = \frac{D_i^-}{D_i^+ + D_i^-}$$

The alternatives are ranked based on relative closeness so the alternative with the highest closeness value is considered the best choice that we can have.

IV. Demonstrative Example Implementation for GEOMM Methodology:

This section will actually represent a demonstrative implementation of GEOMM methodology about how this technique will be implemented within this framework. First user will determine which criteria they are choosing from the proposed GEOMM taxonomy. From these criteria construction of a pairwise comparison matrix is done to determine how much importance each criterion has compared to the other criteria. Then weightage for each criterion is calculated using GEO technique.

Once the weight of the criteria is calculated then MM technique is implemented in which the decision matrix is constructed. MM technique will take two inputs for ranking the best alternatives. The first input will be the weight calculated through GEO technique and the second input is the decision matrix that is present in the form of a database of alternatives. Then MM technique will evaluate the alternatives using these two inputs and determines which one is the best solution for the decision making. So, the criteria that are chosen for this example implementation is shown in table 2.

IEER Institute for Excellence in Education & Research

Table 2. Showing the criteria we have chosen for the example implementation

No.	Criteria
1	Efficiency
2	Web Application
3	PHP
4	Static Testing
5	Open Source
6	Agile
7	White Box
8	Windows

The table 3 shows the pairwise comparison matrix of different chosen criteria for this example which provide us with information about which criterion has what relative importance to which criterion. The higher the number, the higher its relative importance.

Table 3. Showing the pairwise comparison matrix of chosen criteria

	Efficiency	Web App.	PHP	Static Testing	Open Source	Agile	White Box	Windows
Efficiency	1	3	4	4	2	3	1	3
Web App.	1/3	1	3	3	1	5	3	3
PHP	1/2	1/3	1	3	1/3	1	1	3
Static Testing	1/5	1/3	1/3	1	1/3	1	3	1
Open Source	1/4	1	3	3	1	4	2	1
Agile	1/4	1/3	1	1	1/3	1	2	1
White Box	1/2	1/3	1	1/3	1/2	1/2	1	1
Windows	1/3	2	1/3	2	1/2	1/4	1	1

When the GEO technique is applied, we get the following weightage for the chosen criteria depicted in figure 5. It can be seen that the Efficiency criteria have the highest weightage value of 0.25 and White Box criteria has the lowest weightage value of 0.06. The criteria are ranked from high to low weightage.

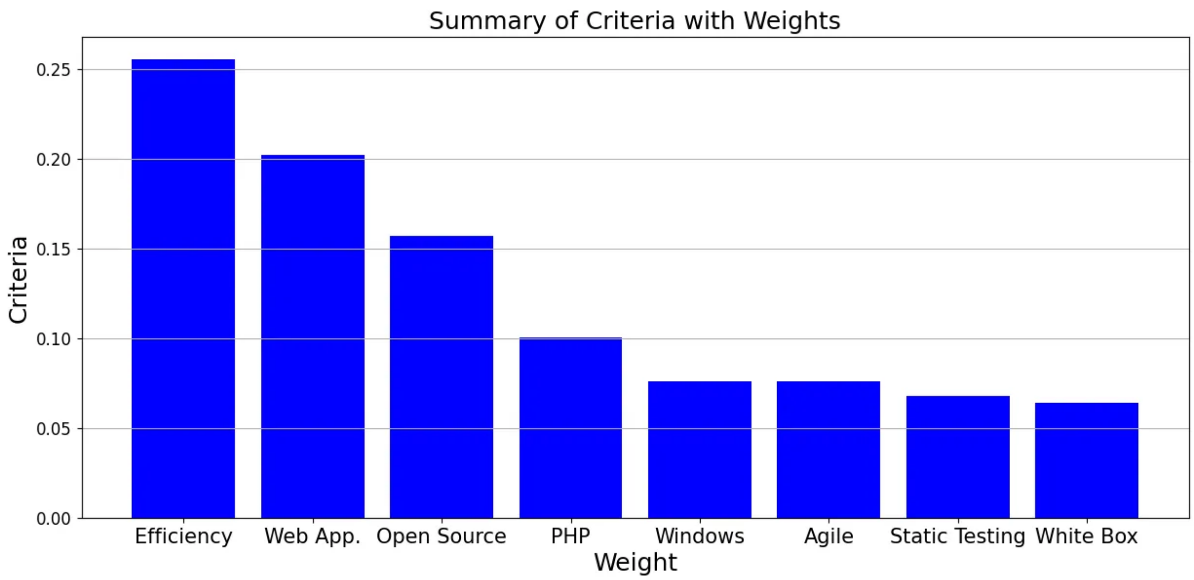

Figure 5. Criteria weightage

Next the decision matrix is constructed for the alternatives which act as the database for the alternatives selection shown in table 4. Each row representing the alternative and each column is representing the criteria that are chosen for this example implementation. Total 6 tools are used in this example implementation and for each tool the values are assigned for the different criteria.

Table 4. Decision matrix for alternatives

	Efficiency	Web App.	PHP	Static Testing	Open Source	Agile	White Box	Windows
Selenium	0.60	0.60	0.55	0.10	0.60	0.25	0.25	0.50
Coverity	0.45	0.20	0.35	0.60	0.00	0.35	0.55	0.50
Zephyr	0.45	0.25	0.15	0.10	0.00	0.30	0.25	0.50
Janova	0.45	0.55	0.30	0.10	0.00	0.30	0.20	0.50
Testrail	0.50	0.30	0.15	0.10	0.00	0.35	0.25	0.50
Appium	0.60	0.40	0.40	0.10	0.60	0.35	0.35	0.50

Next the relative closeness of these six tools is calculated to see which tool is performing best and which tool is performing worst according to the criterion we have chosen in this example. So finally, the alternatives are ranked from best to worst performance and shown in figure 6. It can be seen that the Selenium testing tool is ranked as the best alternative option according to the criteria that we have chosen with closeness coefficient of 0.76. Whereas Zephyr is performing worst and is ranked as the worst possible alternative for the chosen criteria with closeness coefficient of 0.11. So, for this example implementation Selenium is the best possible solution we have and Zephyr is the worst possible solution.

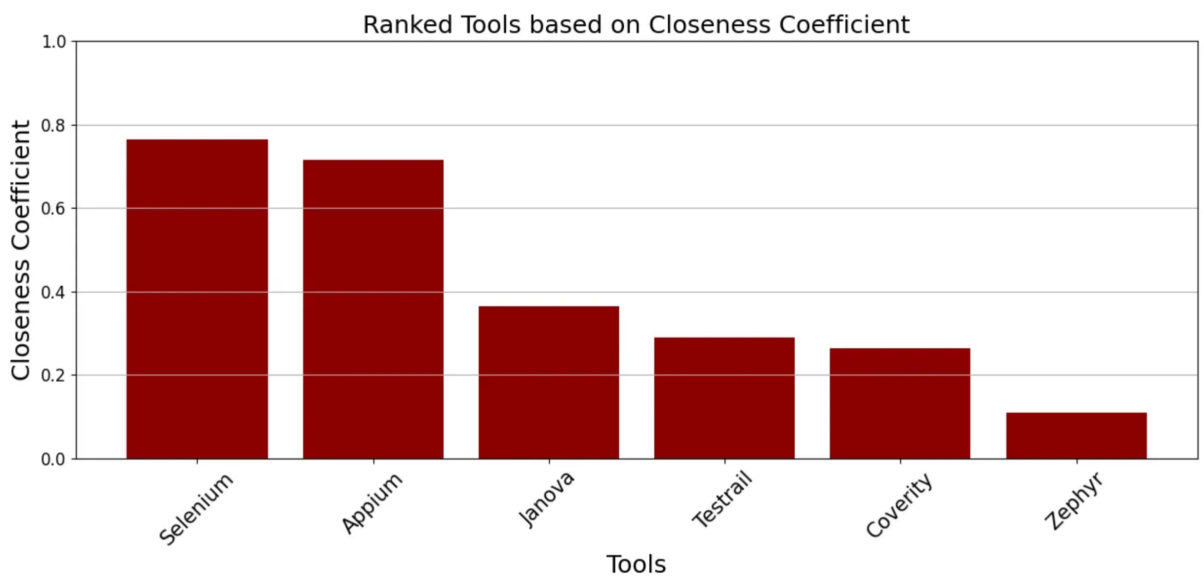

Figure 6. Alternatives ranking from best to worst

V. Framework Implementation and Evaluation:

This section discusses the validation, implementation and evaluation of the proposed GEOMM framework to check out how this framework will perform and if the framework is performing better, then how much performance improvement it provides.

A) Validation:

The GEOMM framework taxonomy which is proposed has been validated first by different academics in the field of software engineering. The GEOMM framework is also validated by different testers of a software systems who use different software tools to test their systems. We have involved them in this research to see whether this framework meets their performance requirements or not. We have received a very good response in this validation because many developers select those testing tools which this framework also make recommendation to them based on the criteria.

B) Implementation:

The GEOMM framework is implemented to check out whether this framework is performing better than the prior framework of software testing tool selection [12]. The GEOMM methodology will be implemented for testing tools selection. If GEOMM framework performs better it will be delivered as a web application. So that it will reach to software developers, developer community, and the industry market through which the framework will be enhanced further. The prior AHP and TOPSIS framework implementation criteria [12] is used for the evaluation of this framework. The same implementation requirements will be considered to make a fair comparison between the performance.

In that implementation 12 testing tools are considered along with their criteria they support [12]. These testing tools are popular tool in the software testing industry and most commonly these testing tools are used to test any software system. In order to widely evaluate GEOMM framework number of criteria are considered for each of these tools.

C) Framework Performance Evaluation:

This section discusses how GEOMM framework is performing for the software testing tools selection. The performance evaluation is an important factor to ensure that GEOMM framework along with taxonomy is providing good performance. So, to check this out different graphs and visualizations for GEOMM framework performance evaluation are depicted below.

In figure 7 we have a bar chart that shows the alternatives closeness comparison side by side both for the prior framework [12] and GEOMM framework. The closeness comparison of prior framework is represented with blue color and for GEOMM framework is represented with red color. On the x-axis the tools are displayed and on the y-axis closeness to ideal solution is displayed and for each tool two bars represent the two frameworks. It can be clearly seen from the figure 7 that our proposed framework called GEOMM framework has better closeness to the ideal solution providing better decision making for alternatives.

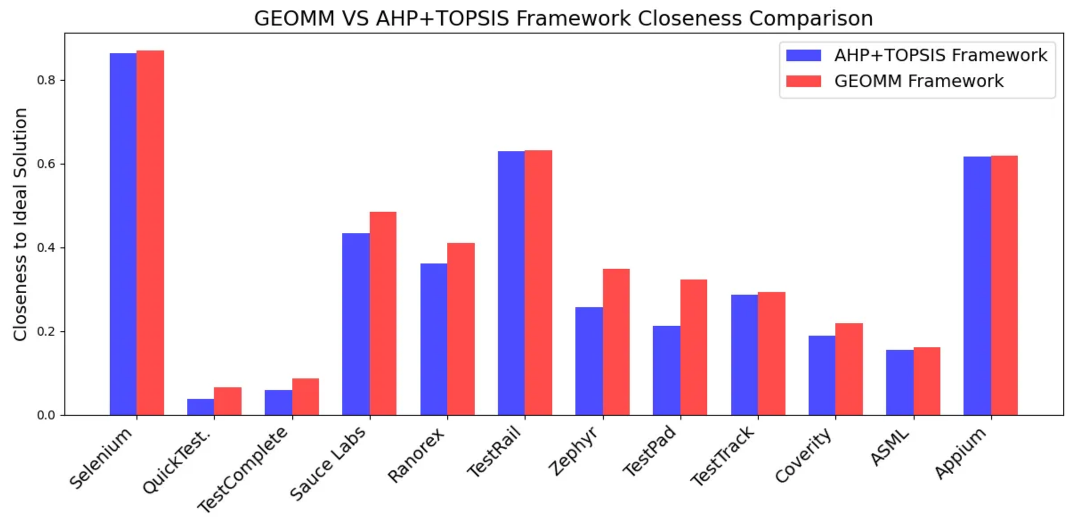

Figure 7. Shows alternatives closeness comparison

In order to represent the performance improvement that GEOMM framework is providing over prior framework the line graph for performance comparison is shown in figure 8. This figure shows that how GEOMM framework is performing better over prior framework compared to each alternative decision making. We can clearly see that GEOMM framework has better performance in term of decision making for the alternatives compared to the prior framework. The blue line flow is representing the decision-making performance of prior framework and the red line flow is representing the decision-making performance of GEOMM framework which is better than prior framework.

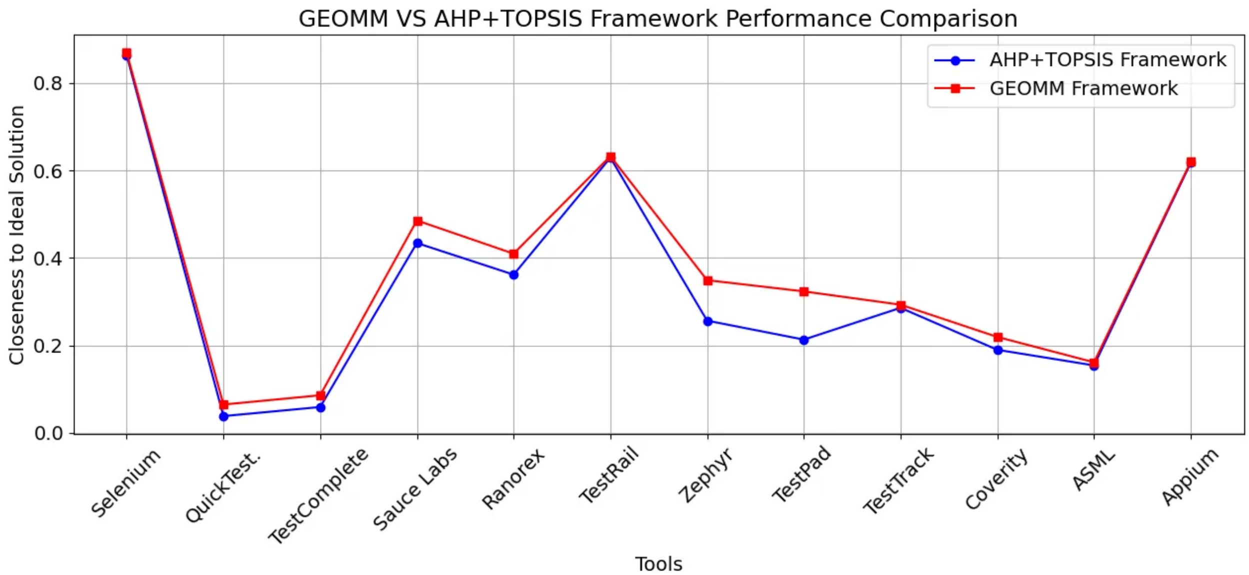

Figure 8. Shows performance improvement of GEOMM framework

To measure the performance achievement of GEOMM framework the speedup plot is shown in figure 9 for this purpose. The speedup plot shows how GEOMM framework has achieved its performance over prior framework. For each alternative the performance achieved by GEOMM framework is depicted in this figure where Quick Test Professional has gained best speedup for this framework compared to prior framework with TestPad as the second-best option. So, for each of the alternative from Selenium to Appium the speedup performance is presented. The x-axis represents the alternatives and y-axis show how much speedup the GEOMM framework each alternative has achieved over prior framework. The single green line shows the speedup value comparison of this framework over prior.

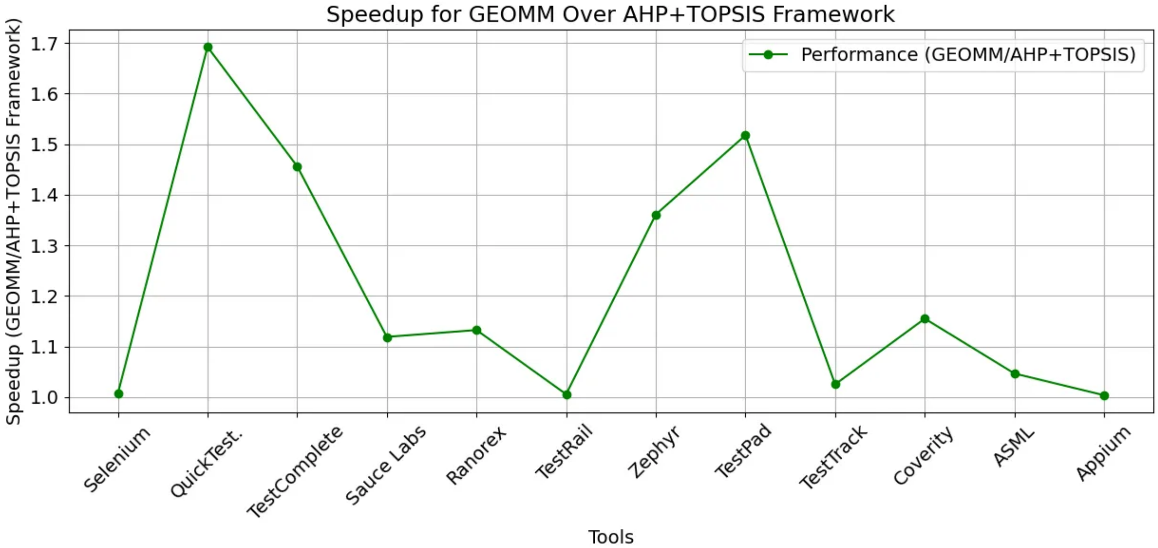

Figure 9. Shows performance speedup for GEOMM framework

As our proposed framework is performing better as expected so to know how much cumulative sum of closeness to ideal solution this framework is providing over prior framework a cumulative sum graph is shown in figure 10. The cumulative sum of prior framework is represented with the blue line and the cumulative sum of GEOMM framework is represented with the red line. The graph shows us how the cumulative sum of closeness to the ideal solution grows as the number of alternatives increases. The cumulative sum graph starts from Selenium for both frameworks. As the number of alternatives increases, we can see there is increase in the cumulative sum values of both frameworks. The proposed GEOMM framework onward has more cumulative sum value of closeness to ideal solution and it continues until the last alternative Appium. The final cumulative sum for closeness to ideal solution for GEOMM framework is 4.5. This shows that GEOMM framework is providing us more closeness to the ideal solution for better decision making for the alternative selection.

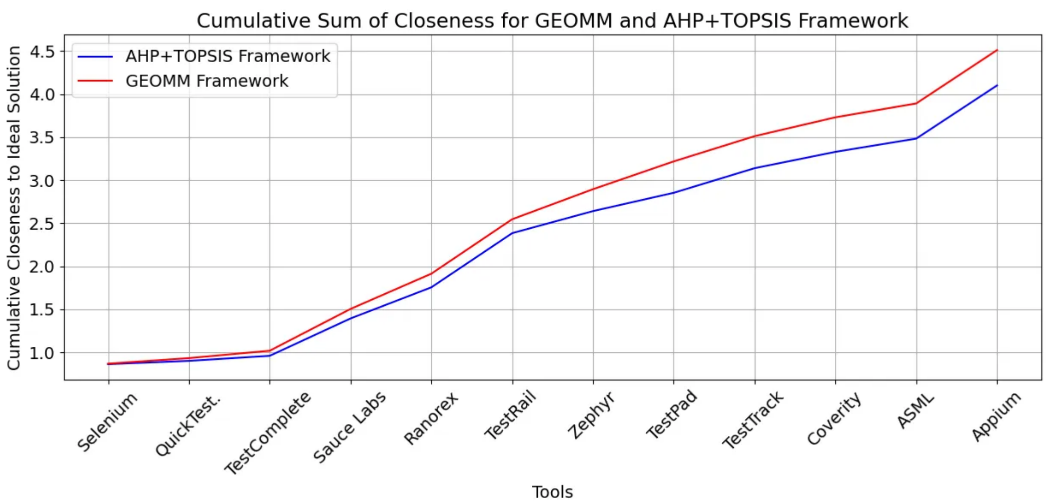

Figure 10. Cumulative sum of closeness comparison

The closeness to the ideal solution is also represented in figure 11 in the form of the boxplot comparison. The boxplot for prior framework and our proposed GEOMM framework is presented. These boxplots actually represent the performance metric in term of best ideal solution recommendation. We can see that the boxplot for the GEOMM framework is more toward the right side indicating higher closeness to the best ideal solution. The x-axis represents the closeness to the ideal solution value which increase toward the right side. So, our proposed framework is plotted more toward the right side indicating that it has better closeness to the ideal solution compared to the prior framework.

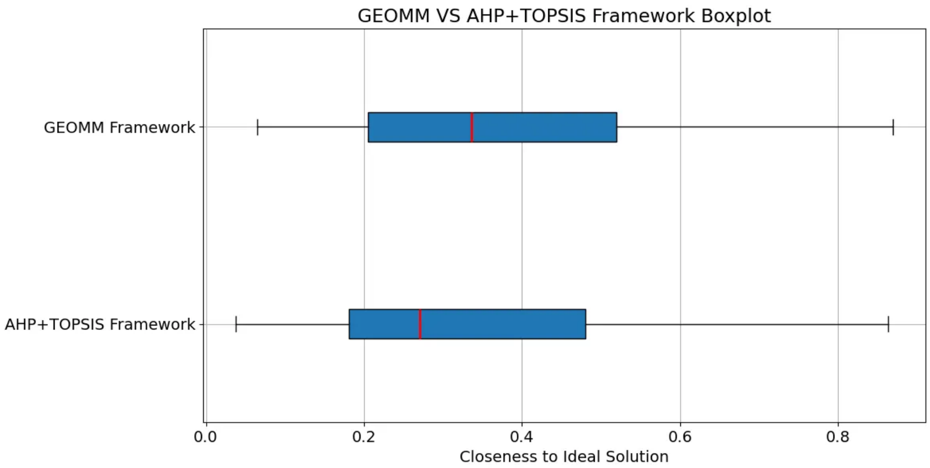

Figure 11. GEOMM and AHP+TOPSIS framework boxplot

So, from all these visualizations we can clearly see that our proposed framework called GEOMM framework has perform more better than the prior framework. The GEOMM framework not only provide us better decision making and more accuracy for the selection of the software testing tools but it also handles the different limitations and drawbacks.

The GEOMM framework not only provide us the percentage improvement for each alternative compare to the prior framework [12] but it also provides performance improvement of 21% for selecting the software testing tools based on the different criteria. Although the inconsistencies and limitations are reduced in the selection approach that we have implement but to still check whether our pairwise comparison matrix is stable for accurate decision making we measure the consistency ratio that is 0.09 which is acceptable.

VI. Conclusion:

In this research we have introduced a new software testing tool selection framework called GEOMM framework along with taxonomy of testing tool criteria. In this framework the first part is about GEOMM taxonomy and second part of this framework is about the implementation of the GEOMM methodology. The academics in the field of software engineering have reviewed GEOMM taxonomy and according to them this taxonomy is useful for developers. The main objective behind why GEOMM framework is introduced is that the previous framework has limitations and drawbacks. The GEOMM framework has been validated and is performing 21% better as compared to the previous framework. Its mean it has 21% of more accurate decision-making regarding software testing tools selection. In the future, this framework can be further improved.

REFERENCES:

J. Wang, Y. Huang, C. Chen, Z. Liu, S. Wang, and Q. Wang, "Software Testing with Large Language Models: Survey, Landscape, and Vision," Mar. 04, 2024, *arXiv*: arXiv:2307.07221. doi: 10.48550/arXiv.2307.07221.

V. Singh, V. Kumar, and V. B. Singh, "A hybrid novel fuzzy AHP-TOPSIS technique for selecting parameter-influencing testing in software development," *Decis. Anal. J.*, vol. 6, p. 100159, Mar. 2023, doi: 10.1016/j.dajour.2022.100159.

A. Abo-eleneen, A. Palliyali, and C. Catal, "The role of Reinforcement Learning in software testing," *Inf. Softw. Technol.*, vol. 164, p. 107325, Dec. 2023, doi: 10.1016/j.infsof.2023.107325.

S. Kumar, "Reviewing Software Testing Models and Optimization Techniques: An Analysis of Efficiency and Advancement Needs," *J. Comput. Mech. Manag.*, vol. 2, no. 1, pp. 32–46, Feb. 2023, doi: 10.57159/gadl.jcmm.2.1.23041.

H. Khorsheed F., J. Ibrahim N., and Q. Saihood, "Software Defect Prediction System Based on Well-Tuned Random Forest Technique," 2023, doi: 10.24412/2413-2527-2023-335-51-59.

R. Van Megen and D. B. Meyerhoff, "Costs and benefits of early defect detection: experiences from developing client server and host applications," *Softw. Qual. J.*, vol. 4, no. 4, pp. 247–256, Dec. 1995, doi: 10.1007/BF00402646.

M. Nashaat and J. Miller, "Refining software defect prediction through attentive neural models for code understanding," *J. Syst. Softw.*, vol. 220, p. 112266, Feb. 2025, doi: 10.1016/j.jss.2024.112266.

A. Ahmed and A. Abdullah, "Enhancing Software Supply Chain Resilience: Strategy for Mitigating Software Supply Chain Security Risks and Ensuring Security Continuity in Development Lifecycle," *Int. J. Soft Comput.*, vol. 15, no. 1/2, pp. 01–18, May 2024, doi: 10.5121/ijsc.2024.15201.

T. R. Devi, "Importance of Testing in Software Development Life Cycle," 2012.

"IJSRD - International Journal for Scientific Research & Development| Vol. 6, Issue 12, 2019 | ISSN (online): 2321-0613," vol. 6, no. 12.

M. Alphonce, "Challenges and Solutions in Software Testing Practices: A Systematic Review in Tanzanian Software Development Companies," 2023, *SSRN*. doi: 10.2139/ssrn.4607901.

A. J. Abdulwareth and A. A. Al-Shargabi, "Toward a Multi-Criteria Framework for Selecting Software Testing Tools," *IEEE Access*, vol. 9, pp. 158872–158891, 2021, doi: 10.1109/ACCESS.2021.3128071.

J. Droste, H. Deters, M. Obaidi, and K. Schneider, "Explanations in Everyday Software Systems: Towards a Taxonomy for Explainability Needs," Apr. 25, 2024, *arXiv*: arXiv:2404.16644. doi: 10.48550/arXiv.2404.16644.

A. Aburas, "Choosing the Right Automated Software Testing Tools," in *2024 IEEE 4th International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA)*, Tripoli, Libya: IEEE, May 2024, pp. 31–35. doi: 10.1109/MI-STA61267.2024.10599723.

S. K. Alferidah and S. Ahmed, "Automated Software Testing Tools," in *2020 International Conference on Computing and Information Technology (ICCIT-1441)*, Tabuk, Saudi Arabia: IEEE, Sep. 2020, pp. 1–4. doi: 10.1109/ICCIT-144147971.2020.9213735.

IEER Institute for Excellence in Education & Research

Department of Computer Science and Engineering, Amity University Uttar Pradesh Lucknow Campus, India and N. Srivastava, "Software and Performance Testing Tools," *J. Inform. Electr. Electron. Eng. JIEEE*, vol. 2, no. 1, pp. 1–12, Jan. 2021, doi: 10.54060/JIEEE/002.01.001.

H. V. Gamido and M. V. Gamido, "Comparative Review of the Features of Automated Software Testing Tools," *Int. J. Electr. Comput. Eng. IJECE*, vol. 9, no. 5, p. 4473, Oct. 2019, doi: 10.11591/ijece.v9i5.pp4473-4478.

M. A. Umar and C. Zhanfang, "A Study of Automated Software Testing: Automation Tools and Frameworks".

J. E. Gaffney, "Metrics in software quality assurance," in *Proceedings of the ACM '81 conference on - ACM 81*, Not Known: ACM Press, 1981, pp. 126–130. doi: 10.1145/800175.809854.

Ahmad, G., Saleem, M., Malik, J. A., Bukhari, W. A., Kashif, M. I., Salahuddin, H., ... & Rehman, A. U. (2024). Mobile application for skin disease classification using CNN with user privacy. Journal of Computing & Biomedical Informatics.

Salahuddin, H., Imdad, K., Chaudhry, M. U., Iqbal, M. M., Bolshev, V., Hussain, A., ... & Jasiński, M. (2022). Electric vehicle transient speed control based on vector control FM-PI speed controller for induction motor. Applied Sciences, 12(17), 8694.

M. Felderer and I. Schieferdecker, "A taxonomy of risk-based testing," Dec. 24, 2019, *arXiv*: arXiv:1912.11519. doi: 10.48550/arXiv.1912.11519.

A. Toloie-Eshlaghy and M. Homayonfar, "MCDM Methodologies and Applications: A Literature Review from 1999 to 2009".

A. Singh, "Major MCDM Techniques and their application-A Review," *IOSR J. Eng.*, vol. 4, no. 5, pp. 15–25, May 2014, doi: 10.9790/3021-04521525.

W. Ho and X. Ma, "The state-of-the-art integrations and applications of the analytic hierarchy process," *Eur. J. Oper. Res.*, vol. 267, no. 2, pp. 399–414, Jun. 2018, doi: 10.1016/j.ejor.2017.09.007.

"Technical Approach of TOPSIS in Decision Making," *Int. J. Recent Trends Eng. Res.*, vol. 3, no. 8, pp. 58–64, Aug. 2017, doi: 10.23883/IJRTER.2017.3388.WPYUJ.

M. Amudha, M. Ramachandran, V. Saravanan, P. Anusuya, and R. Gayathri, "A Study on TOPSIS MCDM Techniques and Its Application," 2021.