

Novel artificial intelligence-based ensemble learning for optimized software quality

Sangeetha Govinda¹, Agnes Nalini Vincent², Merwa Ramesh Babu³

¹Department of Computer Science, Christ University, Bangalore, India

²Faculty of Information Technology, AMITY Institute of Higher Education, Quatre Bornes, Mauritius

³Department of Computer Science, Bharathiar University, Coimbatore, India

Article Info

Article history:

Received Mar 19, 2024

Revised Feb 26, 2025

Accepted Mar 15, 2025

Keywords:

Accuracy

Artificial intelligence

Defect density

Software engineering

Software quality

ABSTRACT

Artificial intelligence (AI) contributes towards improving software engineering quality; however, existing AI models are witnessed to deploy learning-based approaches without addressing various complexities associated with datasets. A literature review showcases an unequilibrium between addressing the accuracy and computational burden. Therefore, the proposed manuscript presents a novel AI-based ensemble learning model that is capable of performing an effective prediction of software quality. The presented scheme adopts correlation-based and multicollinearity-based attributes to select essential feature selection. At the same time, the scheme also introduces a hybrid learning approach integrated with a bio-inspired algorithm for constructing the ensemble learning scheme. The quantified outcome of the proposed study showcases 65% minimized defect density, 94% minimized mean time to failure, 62% minimized processing time of the algorithm, and 43% enhanced predictive accuracy.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Sangeetha Govinda

Department of Computer Science, Christ University

Dharmaram College, Hosur Main Road, Bhavani Nagar, Post, Bengaluru, Karnataka-560029, India

Email: sangeetha.g@christuniversity.in

1. INTRODUCTION

Software quality is one of the essential agendas that assist in ensuring the success of an overall business, better maintainability, and practical system reliability. It positively influences user satisfaction in software engineering [1]. When software is designed in adherence to higher quality standards, it can offer a positive user experience that results in higher retention of users and higher satisfaction [2]. The better design quality of software always ensures higher reliability by reducing unexpected failures, reducing downtime and ensuring better continuity in business processes [3]. A well-structured software code can offer optimal cost reduction with more straightforward maintainability [4]. Apart from this, a well-designed software product is also less susceptible to security vulnerabilities. Thereby it can protect sensitive data and resist lethal threats [5]. However, accomplishing a better software quality standard is often encountered with various challenges. With increasing competition towards yielding the optimal form of product design, the software design system is transforming into a more complex problem where it is quite challenging to ensure optimal quality across all its interactions and components [6]. Changing requirements is another challenge that leads to a more significant problem if not identified on proper product development time [7]. A limited set of time for development is another issue that leads to overlooked software defects that potentially affect design quality [8]. Another significant challenge is associated with resource limitations that involve skilled personnel, budget, and time [9]. It was also noted that when software demands integration with a different platform or system, compatibility issues surface that potentially affect reliability and performance simultaneously [10].

Finally, extensive demand for validation and testing tools and methodologies is sometimes resource-intensive and time-consuming [11].

One effective way to address all the challenges mentioned above is to adopt a preemptive approach where artificial intelligence (AI) significantly enhances software quality. With automated testing, generating test cases, executing them, and performing deeper analysis is possible [12]. The test outcomes can be used for training using a machine learning algorithm, thereby assisting in pattern identification and prediction of potential failures and enhancing testing coverage [13]. Another contribution of AI is to analyze the code quality that can identify potential errors, followed by recommending enhancements that can assist the developers in developing cleaner and maintainable code [14]. AI algorithms can analyze code changes, user feedback, and system behaviour to identify and prioritize bugs. By automatically triaging and assigning bugs, AI streamlines the bug resolution process, reducing the time to detect and fix defects. AI-driven predictive analytics can anticipate software failures by analyzing historical data, system logs, and performance metrics. This proactive approach enables organizations to address issues before they impact users, improving system reliability and uptime. AI can automate parts of the software development process by generating code snippets, templates, or even entire modules based on high-level specifications or design patterns. This accelerates development and reduces the likelihood of errors introduced during manual coding. AI algorithms can optimize software performance by analyzing usage patterns, resource consumption, and system configurations. By dynamically adjusting parameters and configurations, AI systems can maximize efficiency, scalability, and responsiveness. However, there are challenges in effective AI implementation towards software quality improvement [15], [16]. i) AI models trained on biased or incomplete datasets may produce biased predictions, leading to unfair outcomes. Ensuring fairness and mitigating bias in software quality prediction models requires careful data collection, preprocessing, and model training techniques, ii) extracting relevant features from software artefacts for input into AI models requires domain expertise and careful consideration of feature selection techniques. Inadequate feature representation can lead to suboptimal performance and predictive accuracy, and iii) AI models trained on specific datasets or contexts may struggle to generalize to new or unseen scenarios. Ensuring the robustness and generalizability of software quality prediction models across different projects, domains, and environments remains a significant challenge.

The related work in this perspective of AI-based methods has been reviewed to include varied approaches and techniques addressing enhancing software quality. The work conducted by Cheng *et al.* [17] used machine learning to investigate the supportability of tools targeting increasing reliability in validation techniques. Saklamaeva and Pavlič [18] have also investigated AI-based approaches, focusing on software development in agile methodologies. The study showcases the better scope of AI-based methods, providing its inherent issues can be addressed. Kokol [19] have presented a work where the significance of research on software quality is increasing with more advancement of data mining and fault prediction using machine learning. AI methodology has also been investigated concerning pharmaceutical research design, as reported in the work of González *et al.* [20]. The study by Siebert *et al.* [21] discussed a framework for software quality models using machine learning. The work presented by Stocco *et al.* [22] discussed the impact of machine learning and deep learning that can facilitate automated testing programs to address software security threats. Cho *et al.* [23] have developed a unique maturity framework using AI to increase the degree of reliability of software processes where statistical analysis is carried out considering multiple real-time software projects. The discussion presented by Boukhelif *et al.* [24] has disclosed that natural language processing and neural networks are frequently adopted approaches for predictive assessment in software testing. Overall, the authors concluded the beneficial scope of using AI in software testing. The discussion reported by Barenkamp *et al.* [25] has reported a similar aspect of the enhanced scope of AI towards multiple operations in software development, from discovering the pattern to increasing the computational speed. The prime limitation of review is adoption of sophisticated AI-approaches focusing mainly on local perspective of software issues without much consideration of global issues. Another significant issue is related to low computational efficiency being recorded.

The proposed system, therefore, contributes to a novel form of simplified predictive AI-based model towards optimization of the degree of software quality. The novel value added to the proposed study is as follows: i) introduces a simplified scheme to identify the issues followed by improving the large dataset associated with software quality assessment, ii) a novel and simplified empirical scheme is presented towards leveraging the preliminary suitability and usage of complex dataset towards next-level of analytical operation, iii) a simple correlated-based selection method of an essential feature has been presented towards simplifying the complex relationship among the variables in the dataset, and iv) a novel ensemble based AI approach has been used that uses both supervised and unsupervised learning methodologies to carry out predictive analysis of software quality. The following section discusses the research methodology implemented towards accomplishing the above-stated study contribution.

2. METHOD

The proposed system's core purpose is to harness AI's potential to increase the quality score of software development. For this purpose, the proposed system develops a unique dataset motivated by the existing standard dataset [26] that is frequently used for investigating software quality concerning the score of defects. The prime agenda of the proposed analytical study model is to reduce the cost of testing the software quality along with the retention of an optimal score of accuracy while deploying AI. From the perspective of AI, the existing literature finds that existing AI has mainly used learning-based approaches for predicting the score of software quality; however, various degrees of fluctuations and inconsistencies are associated with it. Hence, the proposed scheme implements different variants of learning approaches where only selective attributes and categorization operations are carried out to accomplish optimal accuracy scores. The process flow of the proposed study model is shown in Figure 1.

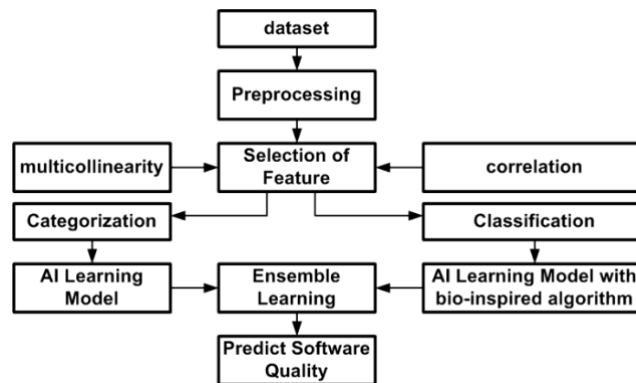


Figure 1. Process flow of proposed study

According to Figure 1, the proposed scheme investigates different forms of learning-based approaches in AI on publicly available datasets with the sole intention of optimizing the accuracy of the dataset in contrast to existing studies. The proposed system uses a clustering approach to group the class labels and then subject the extracted attributes to classification approaches. The scheme also uses a nature-inspired algorithm to optimize the internal operation of learning-based methodologies. Standard performance parameters associated with accuracies are adopted to testify to the study model's effectiveness. Following is the sequence of operations being carried out in the proposed model towards accomplishing the study objectives:

- Preprocessing data: the primary task in the proposed study is to transform the dataset before subjecting it to the learning operation to accomplish a standard format. It was noted that there is a significant gap between the columnar values within the dataset concerning varied standard metrics. This significant difference leads to abnormally higher statistical scores regarding standard deviation. This is sorted out by using a standard scaling mechanism, which can make the dataset much more standardized in contrast to what it was in its original form. The empirical expression of such scale α is represented as (1).

$$\alpha = \frac{A_1}{A_2} \quad (1)$$

In (1), the computation of scale α for standardization is represented as the A_1 and A_2 variables. The variable A_1 is a representation of the difference between observation σ and the mean value of samples of training data μ , i.e., $A_1 = (\sigma - \mu)$, while the second variable A_2 represents standard deviation associated with samples of training γ , i.e., $A_2 = \gamma$.

- Selection of feature: the prime purpose of this module is to minimize the cardinality of features associated with performing training and validation operations in a predictive model. The proposed scheme uses a multicollinearity and correlation approach for determining the significance of undertaken features once the dataset is ready to be processed. The proposed scheme considers that if one feature is directly proportional to another, then it states two features to possess a positive correlation score. However, if one feature is inversely proportional to another, it states two features to possess a negative correlation score.

On the other hand, if one feature value doesn't affect the feature value of another, the proposed scheme considers it to have a zero-correlation score. To simplify the analysis, the proposed scheme considers only the initial m number of n attributes ($n \ll m$) where such selected m attributes are either negatively or zero correlated without the presence of any class labels. This selection process of features assists in significantly controlling the overfitting issues and minimizing the operation cost involved in optimizing the performance of the proposed study model. The proposed scheme adopts the hybrid learning approach in AI where unsupervised and supervised learning approaches are used for categorizing and classification, respectively towards optimizing the software quality prediction. Following is further information about its implementation.

- Categorization: the proposed study model uses the k-means clustering (KMC) approach to study class labels to choose the cardinality of clusters optimally. The empirical form of the k score of clusters is as (2):

$$\delta = \sum_{i=1}^n \tau^2 \quad (2)$$

The above empirical expression (2) represents δ , i.e., the sum of squares presents in one cluster out of k clusters represented by τ , i.e., the distance between the data point and respective centroid in the k number of clusters.

- Classification: the proposed scheme uses a supervised learning method for the data characterized by the class label as the output. Further, the scheme classifies the data into 70% and 30% of the training and testing data, respectively. The data with class labels are considered for training while data without any class labels are used for testing. Further, the scheme uses multiple classifiers to conduct classification analysis in the form of ensemble classifiers. The scheme uses random forest (RF), naïve Bayes (NB), and support vector machine (SVM), where the RF and SVM are considered as candidate predictive models. In contrast, NB is considered a baseline predictive model for performing classification.

The prime justification behind adopting the RF approach is that it acts as an integrated learning method that integrates varied forms of classifiers for optimizing the predictive outcome. Multiple decision trees can be deployed to the data subset followed by extracting its mean value to arrive at the final value of predictive performance. The scheme uses a 950 tree structure along with 50 arbitrary states for implementing this classifier. The scheme uses the NB algorithm known for its categorization capabilities and classification. This approach computes all the likelihoods and then estimates probabilities that don't favour the likelihoods. The approach works suitably in the presence of no connection among the essential features of the dataset. The scheme considers 40 as a state of arbitrariness. Finally, the proposed scheme uses SVM, another dominant form of supervised learning approach in AI. Considering a restricted dataset size, SVM offers better performance and reduced processing time. The considered data is classified using a boundary of decision by SVM where each involved class are classified. The system accomplishes an optimal hyperplane in the presence of the highest degree of margins generated from all the classes. A similar value of 40 is considered a state of arbitrariness in SVM implementation.

It is to be noted that the proposed AI method uses an ensemble form of predictive approach and not an integrated form of predictive approach. It will mean that the proposed scheme has considered one learning model NB as the baseline predictive model while other models RF and SVM act as candidate predictive models. The proposed scheme implements this AI approach using a stacking classification-based methodology. At the same time, a value of 40 is also maintained to implement this ensemble predictive model with respect to its value of state of its arbitrariness. The standard performance parameters associated with the accuracy-based attributes are considered for the assessment. The following section elaborates on the outcome after implementing this scheme of predicting software quality.

3. RESULTS

This section discusses the outcome of the proposed study illustrated in the prior section. The implementation of the proposed study has been carried out considering the standard CM1 dataset that consists of 22 properties with 499 modules with 449 defect free instances, 49 defective instances, and written in C language. A closer look into this dataset shows approximately a 10% defect rate. From the perspective of metrics, this dataset consists of Halstead and McCabe metrics, which are numerical data and method-level attributes. This dataset is subjected to preprocessing, selecting essential features, and performing an ensemble AI-based learning approach with respect to categorization and classification. The proposed system model is scripted using Python, considering a normal windows machine. The proposed system is

benchmarking by comparing it with an existing standalone approach of learning algorithms, viz. NB, SVM, RF, KMC. It is also compared with standalone bio-inspired approaches viz: particle swarm optimization (PSO) and ant colony optimization (ACO). The assessment concerns defect density, mean time-to-failure (MTTF), accuracy, and algorithm processing time.

3.1. Accomplished outcome

The primary performance metric evaluated in the outcome assessment process is defect density, which is computed as total number of defects in each code line. It can also be analyzed from the functional points. A practical design of software will always anticipate for lower number of defect densities. To offer clear and understandable inference, the defect density score is transformed to probability values for better quantification of outcomes. The simulated outcome of the proposed system with an existing AI-based other standalone learning approaches and bio-inspired approaches are shown in Figure 2.

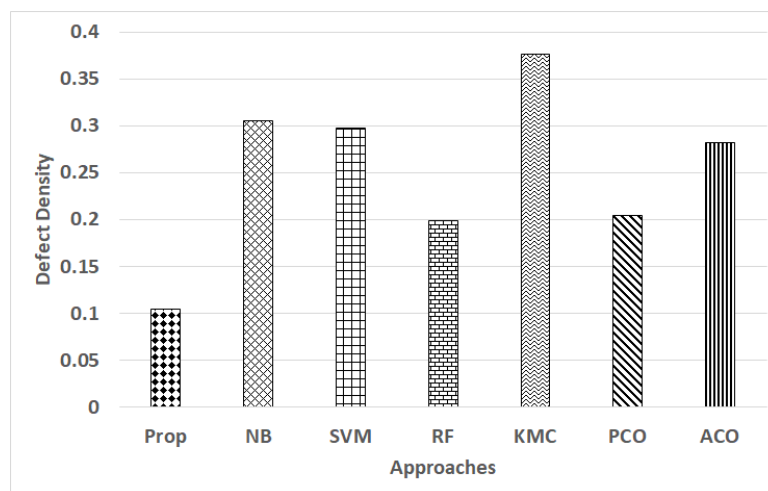


Figure 2. Comparative analysis of defect density

The outcome shown in Figure 2 showcases that the proposed system prop offers approximately 65% of the reduced score of defect density in contrast to the existing system. Some of the interesting findings can be withdrawn from this outcome. A closer look into the outcome shows that the NB approach, which acts as a baseline model in the proposed system, performs much better than the standalone NB approach. A similar trend is also seen for conventional standalone SVM and RF approaches, also used in ensemble form in the proposed system. It can also be seen that KMC usage as a standalone approach offers more defects in contrast to its integrated usage in the proposed scheme. This outcome showcases that the ensemble approach of AI used in the proposed system offers better reduction of defect density performance in contrast to conventional AI-based standalone approaches. Apart from this, it is noted that using bio-inspired approaches integrated into the proposed system has reduced defect density scores compared to standalone PSO and ACO algorithms. The second line of analysis shown in Figure 3 is associated with evaluating MTTF, computed as mean duration between the software failures represented in the probability score. Further, Figures 3 to 5 shows that the proposed system offers approximately 94% reduced MTTF, 43% increased accuracy, and 62% reduced algorithm processing time in contrast to existing AI-based approaches for enhancing software quality prediction.

3.2. Discussion of results

The overall result score showcases that the proposed system offers a consistent pattern of outcomes compared to existing AI-based approaches. The prime reason behind these outcome patterns and trends shown in graphical outcomes can be justified as follows: unlike any conventional studies with AI-based software engineering solutions, the proposed system doesn't consider its input dataset as it is subjected to learning approaches. Instead, the raw dataset undergoes a series of operations, eliminating its inconsistency and enhancing the quality, thereby offering higher data purity. Hence, it offers a significantly low computational burden when AI-based methodologies are applied. The proposed scheme presents a novel ensemble approach of dual forms after performing the categorization operation using KMC. The first

ensemble approach was to consider RF, NB, and SVM, while the second was to integrate all three learning approaches with the bio-inspired approach of PSO. This reduces the computational load and increases the accuracy, which can be directly stated as the core reason for software quality improvement.

However, the conventional AI-based approaches in their standalone form were witnessed with sub-optimal performance scores. When the NB approach is used in the proposed design implementation, it effectively offers effectiveness towards better processing of categorical data and higher dimensional data. However, when used as a standalone form, it showed under-performance issues in all the evaluation metrics, especially in the presence of a higher number of correlated features. Further, standalone NB was proven not to offer much assistance towards complex relationships within the data. The second algorithm of SVM, when used with the proposed system, showcased its capability to process high-dimensional data, and there are not many issues towards overfitting. However, the standalone usage of SVM on the CM1 dataset was witnessed with higher processing time. A similar trend was also witnessed for the RF algorithm, which lacked interpretability and was encountered with an intermittent slow training process.

Further, the RF algorithm doesn't perform well in an imbalanced dataset. It is to be noted that the KMC algorithm was significantly assistive in the proposed scheme towards categorization with more straightforward implementation efficiency for larger datasets, too. The standalone version of KMC was noted with higher sensitivity towards initial clusters and demands the acquisition of predefined clusters, which may not be suitable for the real-time environment of software design assessment. When used with the proposed ensemble approach with AI methods, a bio-inspired approach like PSO was witnessed with more straightforward implementation with better consistency. However, the standalone version of PSO and ACO is witnessed to offer limited performance in high-dimensional spaces and slower convergence speeds, respectively. This results in higher defect density, higher MTTF, lower accuracy, and increased algorithm processing time. Hence, the overall outcome suggests the proposed ensemble AI-based approach to offer optimal software design quality.

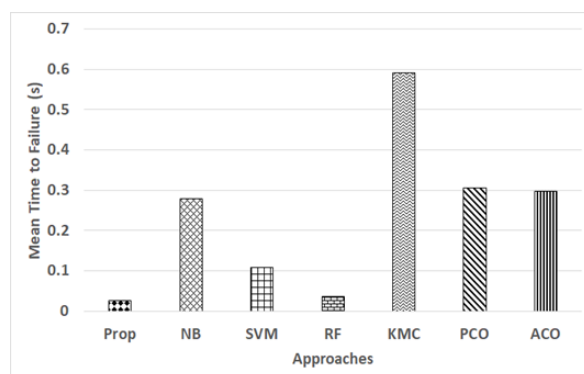


Figure 3. Comparative analysis of mean time to failure

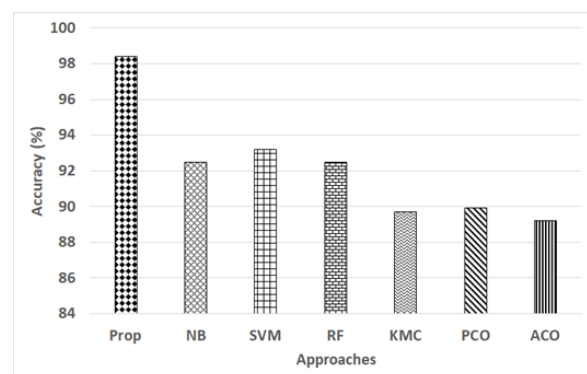


Figure 4. Comparative analysis of accuracy

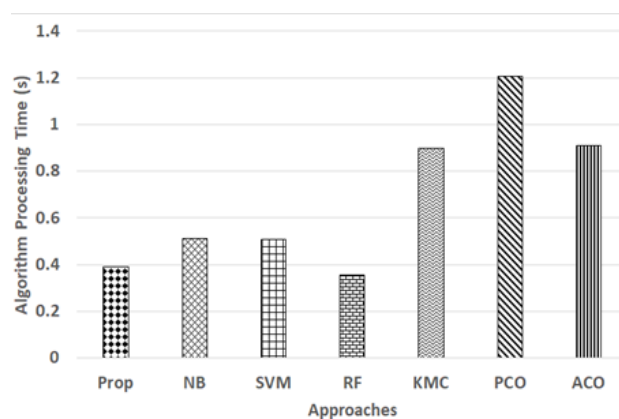


Figure 5. Comparative analysis of algorithm processing time

4. CONCLUSION

The proposed system has presented a novel ensemble-based AI method towards leveraging the predictive performance associated with software quality. The proposed study has deployed learning-based approaches that assist towards feature selection while the software quality prediction is carried out by clustering. The proposed study model contributes towards the following novel features: i) the proposed scheme uses a standard publicly available dataset for software quality that is subjected to empirically designed scalar attributes towards performing the preprocessing operation, unlike the existing approach that doesn't emphasize much on this step, ii) the proposed scheme also introduced a simplified selection of feature considering correlation and multicollinearity attribute that is successfully used for establishing relationship among complex attributes of dataset, iii) the proposed scheme uses unsupervised learning approach in AI for performing categorization while supervised approach is used towards carrying out predictive performance, and iv) proposed study model offers approximately 65% of minimized defect density, 94% of minimized MTTF, 62% of minimized processing time of algorithm, and 43% of enhanced predictive accuracy. However, the prime limitation of the current work is that it doesn't address the problem of improving software quality where there is a considerable lesser amount of inputs or necessary ground truth information. This limitation can be addressed in future direction of work where generative AI modelling can be carried out towards generating possible cases of synthesise data.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Sangeetha Govinda	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
Agnes Nalini Vincent	✓		✓	✓			✓			✓	✓		✓	✓
Merwa Ramesh Babu	✓				✓		✓	✓	✓	✓	✓			

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review & Editing

Vi : Visualization

Su : Supervision

P : Project administration

Fu : Funding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author, [SG], upon reasonable request.




REFERENCES

- [1] A. Alami and O. Krancher, "How scrum adds value to achieving software quality?," *Empirical Software Engineering*, vol. 27, no. 7, Dec. 2022, doi: 10.1007/s10664-022-10208-4.
- [2] P. Karhapää *et al.*, "Strategies to manage quality requirements in agile software development: a multiple case study," *Empirical Software Engineering*, vol. 26, no. 2, Mar. 2021, doi: 10.1007/s10664-020-09903-x.
- [3] L. Chazette, W. Brunotte, and T. Speith, "Explainable software systems: from requirements analysis to system evaluation," *Requirements Engineering*, vol. 27, no. 4, pp. 457–487, Dec. 2022, doi: 10.1007/s00766-022-00393-5.
- [4] L. Lavazza, S. Morasca, and M. Gatto, "An empirical study on software understandability and its dependence on code characteristics," *Empirical Software Engineering*, vol. 28, no. 6, Nov. 2023, doi: 10.1007/s10664-023-10396-7.
- [5] M. Aydos, Ç. Aldan, E. Coşkun, and A. Soydan, "Security testing of web applications: a systematic mapping of the literature," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 9, pp. 6775–6792, Oct. 2022, doi: 10.1016/j.jksuci.2021.09.018.
- [6] P. Orviz Fernández, M. David, D. C. Duma, E. Ronchieri, J. Gomes, and D. Salomoni, "Software quality assurance in INDIGO-datacloud project: a converging evolution of software engineering practices to support european research e-infrastructures," *Journal of Grid Computing*, vol. 18, no. 1, pp. 81–98, Mar. 2020, doi: 10.1007/s10723-020-09509-z.




- [7] M. A. Akbar, A. A. Khan, S. Mahmood, and A. Mishra, "SRCMIMM: the software requirements change management and implementation maturity model in the domain of global software development industry," *Information Technology and Management*, vol. 24, no. 3, pp. 195–219, Sep. 2023, doi: 10.1007/s10799-022-00364-w.
- [8] L. Neelu and D. Kavitha, "Estimation of software quality parameters for hybrid agile process model," *SN Applied Sciences*, vol. 3, no. 3, Mar. 2021, doi: 10.1007/s42452-021-04305-0.
- [9] E. Ronchieri and M. Canaparo, "Assessing the impact of software quality models in healthcare software systems," *Health Systems*, vol. 12, no. 1, pp. 85–97, Jan. 2023, doi: 10.1080/20476965.2022.2162445.
- [10] S. Wai, "Software quality and backward compatibility in the video game industry," *Journal of Industrial and Business Economics*, vol. 49, no. 3, pp. 545–570, Sep. 2022, doi: 10.1007/s40812-022-00224-2.
- [11] M. Saadatmand *et al.*, "SmartDelta project: automated quality assurance and optimization across product versions and variants," *Microprocessors and Microsystems*, vol. 103, Nov. 2023, doi: 10.1016/j.micpro.2023.104967.
- [12] V. Lenarduzzi, F. Lomio, S. Moreschini, D. Taibi, and D. A. Tamburri, "Software quality for AI: where we are now?," *Software Quality: Future Perspectives on Software Engineering Quality*, Springer, Cham, vol. 404, pp. 43–53, Jan. 2021, doi: 10.1007/978-3-030-65854-0_4.
- [13] F. Alaswad and E. Poovammal, "Software quality prediction using machine learning," *Materials Today: Proceedings*, vol. 62, pp. 4714–4720, 2022, doi: 10.1016/j.matpr.2022.03.165.
- [14] A. Khan, R. R. Mekuria, and R. Isaev, "Applying machine learning analysis for software quality test," in *2023 International Conference on Code Quality (ICQ)*, pp. 1–15, Apr. 2023, doi: 10.1109/ICQ57276.2023.10114664.
- [15] F. A. Batareseh, L. Freeman, and C.-H. Huang, "A survey on artificial intelligence assurance," *Journal of Big Data*, vol. 8, no. 1, Dec. 2021, doi: 10.1186/s40537-021-00445-7.
- [16] B. Gezici and A. K. Tarhan, "Systematic literature review on software quality for AI-based software," *Empirical Software Engineering*, vol. 27, no. 3, May 2022, doi: 10.1007/s10664-021-10105-2.
- [17] K. S. Cheng, P.-C. Huang, T.-H. Ahn, and M. Song, "Tool support for improving software quality in machine learning programs," *Information*, vol. 14, no. 1, Jan. 2023, doi: 10.3390/info14010053.
- [18] V. Saklamaeva and L. Pavlič, "The potential of AI-driven assistants in scaled agile software development," *Applied Sciences*, vol. 14, no. 1, Dec. 2023, doi: 10.3390/app14010319.
- [19] P. Kokol, "Software quality: how much does it matter?," *Electronics*, vol. 11, no. 16, Aug. 2022, doi: 10.3390/electronics11162485.
- [20] A. B. -González *et al.*, "The role of AI in drug discovery: challenges, opportunities, and strategies," *Pharmaceuticals*, vol. 16, no. 6, Jun. 2023, doi: 10.3390/ph16060891.
- [21] J. Siebert *et al.*, "Construction of a quality model for machine learning systems," *Software Quality Journal*, vol. 30, no. 2, pp. 307–335, Jun. 2022, doi: 10.1007/s11219-021-09557-y.
- [22] A. Stocco *et al.*, "Software testing in the machine learning era: special issue of the empirical software engineering (EMSE) journal," *Empirical Software Engineering*, vol. 28, no. 3, May 2023, doi: 10.1007/s10664-023-10326-7.
- [23] S. Cho, I. Kim, J. Kim, H. Woo, and W. Shin, "A maturity model for trustworthy AI software development," *Applied Sciences*, vol. 13, no. 8, Apr. 2023, doi: 10.3390/app13084771.
- [24] M. Boukhelif, M. Hanine, and N. Kharmoum, "A decade of intelligent software testing research: a bibliometric analysis," *Electronics*, vol. 12, no. 9, May 2023, doi: 10.3390/electronics12092109.
- [25] M. Barenkamp, J. Rebstadt, and O. Thomas, "Applications of AI in classical software engineering," *AI Perspectives*, vol. 2, no. 1, Dec. 2020, doi: 10.1186/s42467-020-00005-4.
- [26] A. Ali, N. Khan, M. Abu-Tair, J. Noppen, S. McClean, and I. McChesney, "Discriminating features-based cost-sensitive approach for software defect prediction," *Automated Software Engineering*, vol. 28, no. 11, pp. 1–18, Jul. 2021, doi: 10.1007/s10515-021-00289-8.

BIOGRAPHIES OF AUTHORS






Dr. Sangeetha Govinda    an esteemed faculty member in the Department of Computer Science at Christ (Deemed to be University), Central Campus, Bangalore, India, holds a Ph.D. degree from Bharathiar University, Coimbatore. With an extensive career spanning over 19 years, she has made significant contributions to teaching, research, and administration, shaping educational methodologies across undergraduate and postgraduate levels. Her scholarly endeavors are underscored by the publication of 6 national and 12 international research papers in prestigious journals indexed in IEEE, WoS, and Scopus. Beyond academia, she actively serves as a board of examination (BOE) member for esteemed institutions such as Bengaluru City University, Bangalore University, and Mount Carmel College. Additionally, she contributes her expertise as a member of the review committee for ASTES journal. Her diverse research interests encompass data mining, IoT, software engineering, cryptography, computer networks, R basics, and IT for Business. Her dedication to academic excellence extends beyond research, as evidenced by her numerous invited talks, guest lectures, international and national conference participation, and organization of workshops, seminars, and faculty development programs (FDPs). Recognized for her outstanding contributions, she was honored as a Microsoft research fellow in 2014. Furthermore, she has received acclaim for her innovative work, including an Australian Patent (No. 2021103341) granted for eight years from June 15, 2021, on August 4, 2021, for her groundbreaking project titled "Artificial intelligence based automatic detection of infection rate of COVID-19." Her remarkable journey exemplifies her unwavering dedication to advancing education and research in the field of computer science, leaving an indelible mark on both academia and society. She can be contacted at email: sangeetha.g@christuniversity.in.



Agnes Nalini Vincent    is the Dean of Faculty of Information Technology and Head of Teaching and Learning at AMITY Institute of Higher Education (AIHE), Mauritius. She holds a master's in engineering degree from Anna University, Chennai, India and is currently pursuing her Ph.D. in the field of artificial intelligence. With over 17+ years of experience in teaching, research and administration, she has been instrumental in framing the pedagogies from undergraduate to post graduate studies. She has been the pioneer to develop curriculum in big data analytics, internet of things and data sciences and launched them as courses in Mauritius. She has published national and international research papers in journals indexed in IEEE, and Scopus. As a faculty, she has been offering a wealth of talent in the development and implementation of educational technology tools and applications in the classroom. Her area of interests includes artificial intelligence, data mining, big data analytics, internet of things, computer networks, and business data analytics. She has deeply invested in achieving her tenure through administrative service contributions and an accomplishment-oriented approach to teaching. She has given more than 10+ invited talks, 12+ guest lectures, has conducted 1 International conference, has organized 5+ national and international workshops. Her contribution to quality standards formulation and to development of credit system in the capacity of head of teaching and learning at AIHE is remarkable. Her sterling track record of academic excellence and visionary leadership brings a wealth of knowledge, experience, and insight to the forefront of the ICT field. Through her unique and people-centric approach towards administration and management, she has always fostered a supportive and collaborative environment that enables each member of the team to reach their full potential. She can be contacted at email: vanalini@mauritius.amity.edu.



Merwa Ramesh Babu    is the research scholar in Bharathiar University, Coimbatore, Tamil Nadu, India. He holds a master's in computer applications degree from Bangalore University, Bangalore, India and is currently pursuing his Ph.D. in the field of internet of things (IoT). With over 13+ years of experience in teaching, research and administration, he has been instrumental in framing the pedagogies from undergraduate to post graduate studies. He has been the pioneer to develop curriculum in object-oriented programming using java, internet of things and data sciences and launched them as courses in various institutions. He has published national and international research papers in journals indexed in IEEE, and Scopus. As a research scholar, he has been offering a wealth of talent in the development and implementation of educational technology tools and applications in the research field. His area of interests includes artificial intelligence, data mining, big data analytics, internet of things, mobile application development, and oop's using java. He has attended more than 10+ invited talks, 12+ guest lectures, 1 international conference, 2+ national and international workshops. He can be contacted at email: merwaramesh@gmail.com.