Journal of Information & Optimization Sciences
ISSN 0252-2667 (Print), ISSN 2169-0103 (Online)
Vol. 47 (2026), No. 7, pp. 2789–2806
DOI : 10.47974/JIOS-2366

AI-driven optimization model for software requirement prioritization

Deepali Shahane [@]
Department of Computer Science and Applications
School of Computer Science & Engineering
Dr. Vishwanath Karad MIT World Peace University
Pune 411038
Maharashtra
India

Babasaheb Dnyandeo Patil [†]
Department of Computer Applications
Institute of Management & Rural Development Administration
Bharati Vidyapeeth (Deemed to be University)
Sangli 416416
Maharashtra
India

Prashant Kharat [§]
Department of Information Technology
Walchand College of Engineering
Shivaji University
Sangli 416415
Maharashtra
India

[@] *ORCID-ID:* **0000-0002-3099-9009**
[†] *ORCID-ID:* **0000-0002-6339-9595**
[§] *ORCID-ID:* **0000-0002-7111-2291**

[@] *E-mail:* **shahanedeepali@gmail.com**
[†] *E-mail:* **babasaheb.d.patil@bharatividyapeeth.edu**
[§] *E-mail:* **prashant.kharat@walchandsangli.ac.in**

Manisha Shinde-Pawar *
Department of Computer Applications
Kasegaon Education Society's Rajarambapu Institute of Technology
Sakharale 415414
Maharashtra
India

Vikas V. Patil ‡
Department of Management Studies
Yashwantrao Mohite Institute of Management
Bharati Vidyapeeth (Deemed to be University)
Karad 415539
Maharashtra
India

Abstract

Manual Software requirements engineering has encountered significant challenges related to time overhead, high human effort, susceptibility to errors, and limited scalability for dynamic change handling, ranking and prioritization of non-functional requirements and requirements change requests. This study proposes an adaptive, dynamic, scalable, and AI-driven optimization model for software requirements scaling and prioritization using advanced machine learning and hybrid AI-based prioritization to improve accuracy and efficiency in requirement decision-making.

The proposed approach employs hybrid AI-driven framework integrating machine learning models for requirement classification and prediction, natural language processing for text processing, optimization-based scoring for ranking and prioritization, and domain-aware AI models. Software Requirements Prioritization and Change Management Model (SRPCMM) embed AI-driven analysis across requirement engineering phases, enabling optimization-based scoring and dynamic re-prioritization, Natural Language Processing (NLP) -based ambiguity reduction, and efficient change management. A novel risk-adjusted weighted priority scoring mechanism supports realistic and integrated criteria-driven evaluation. The experimental evaluation on an industrial software dataset has shown a 25–30% improvement in prioritization accuracy. The proposed model addresses key limitations

* *ORCID-ID:* **0000-0002-8276-9782**
‡ *ORCID-ID:* **0000-0003-2964-3838**

* *E-mail:* **manisha.pawar@ritindia.edu; (Corresponding Author)**
mjs.imrda@gmail.com
‡ *E-mail:* **vikas.patil@bharatividyapeeth.edu**

of traditional requirement engineering by enabling integrated, automated, and intelligent decision-making processes.

Subject Classification: *68N30, 68T05, 90C27, 90C59, 90B50.*

Keywords: *AI requirement engineering, Machine learning, Optimization model, Requirement prioritization, Software engineering.*

1. Introduction

Software engineering is a disciplined, systematic, process-oriented approach for the development of high-quality software. It includes software requirements engineering as a critical technique to discover, describe, analyze, classify, validate and model requirements. This process provides an opportunity to describe software features, functions, processes, and constraints. Software configurations manage and control the requirement change requests. Discovering requirements is a challenging process for complex and new technologies. Requirement Elicitation and prioritization is a time-consuming, expensive and risk-prone method that can slow down the overall development process [1]. Predicting future requirements to determine whether they become obsolete or require redesign is also challenging. The application of machine learning can improve the accuracy of data analysis [2]. Automated decision-making enhances software project management by ensuring that requirements remain aligned with organizational and business objectives.

2. Review of literature

Traditional Approach

Traditional requirement engineering methods employ structured techniques such as the Analytic Hierarchy Process (AHP), Must have, should have, could have, Will not have (MoSCoW), and decision matrix approaches to prioritize requirements based on predefined criteria. Requirement ratings are evaluated based on criteria such as importance, urgency, and feasibility. The MoSCoW technique performs manual qualitative input classification into four categories—must have, should have, could have, and won't have. Though these techniques are effective for small to medium-sized projects, conventional approaches often face scalability challenges as the number of requirements [3]. Exponential growth in requirements limits the practical applicability of these approaches in complex projects. Additionally, Manual ratings often

introduce bias and inconsistency, making it difficult to systematically resolve conflicting priorities. These approaches also fail to handle dynamic changes and slow down responsiveness and the overall requirement management process. Despite their critical importance to software quality, nonfunctional requirements are often treated separately or inadequately. Therefore, traditional manual approaches and structured decision frameworks face challenges related to scalability, automation, and complex interdependencies. This has motivated the development of AI-driven and hybrid techniques to overcome these limitations [4, 5].

Agile requirement engineering introduces a systematic three-dimensional approach to mitigate Software Requirements Specification (SRS) challenges without relying on Artificial Intelligence or Machine Learning techniques. The Analytic Hierarchy Process (AHP) originally introduced a significant and reliable requirement prioritization method but with limited scalability [6]. AHP is found to be more reliable, dependable, accurate and significant than numerical assignment techniques for large projects' requirements prioritization [7]. Hudaib et al. [1] underscore the significance of key aspects of traditional project success stakeholder views and non-functional requirements in prioritization. Stakeholder coordination and limited effectiveness persist as challenges in complex prioritization [8]. Fuzzy logic-based software requirement analysis and change management improves classification, scaling and prioritization [9, 10]. The QuARS (Quality Analyzer for Requirements Specifications) tool–based Quality Model is presented for refined, unambiguous assessment of SRS semantic quality through lexical and syntactical analysis.

AI (Artificial Intelligence)-ML (Machine Learning) Approach

AI has significantly transformed software engineering by enabling process automation, improving accuracy, and enhancing productivity [2]. According to Babar et al. [4] integration of neural network, AHP and value-based techniques can reduce professional partiality and boost accuracy. Integration of Human judgement with AI techniques provides significant and accurate results [11] The study suggests that systemized project management transitioning to an automated AI-based project management process can be a holistic approach. The study, Studies by Avesani et al. [5], Perini, Susi, and Avesani [12], and Richardson et al. [13] reported that early requirement management, automated elicitation, risk identification, and handling uncertainty vague and incomplete

requirement challenges can be addressed through modern AI-ML driven TITLE NAME 5 solutions. AI ML integrated software requirement analysis, planning, monitoring, control, scaling, prioritization techniques are found to be effective and enhanced solutions for large, complex and multifaceted project requirements.

Hybrid Approach

Integration of modern soft computing techniques like fuzzy logic, AI ML reforms traditional requirement analysis, prioritization and scaling techniques to address uncertainty, ambiguity, and incompleteness [3]. Earlier studies by Avesani et al. [11] demonstrated the effectiveness of hybrid requirement prioritization models. Akbar et al. [6] and Arshad et al. [14] suggested potential of AI integration to maturity model for improvement in insights and requirements prioritization. With reduced risk, efforts, cost and time constraints, hybrid approach provides significance in terms of accuracy, prediction, flexible, robust, and reliable solutions [15, 16, 17, 18]. Shabbir et al. [19] suggest that future research should extend the real-time adaptive framework to account for human, organizational, and domain-specific risk factors. Integration of these contextual dimensions can lead in comprehensive, dynamic, realistic, and practical software requirement classification and prioritization models.

3. Research gap

1. Traditional prioritization strategies need thorough analysis and investigation due to scalability and stakeholder preferences issues. For large project requirement prioritization, ranking an analysis becomes difficult and complicated. Even small-scale requirement ranking and prioritization found tedious tasks need to reanalyze, re-examine and reform prioritization techniques. Poor stakeholder communication, slow manual process and tasks persist as a major challenge in requirement elicitation and further requirement engineering.
2. It also traced the gap in experimentation rather than telecommunication. Inclusion of non-functional requirements prioritization strategies for improved decision quality is also a potential integration opportunity for research.
3. There is also a gap identified for integration of system for requirement elicitation methods, dynamic requirement management, and

classification of ambiguous, vague, uncertain and/or incomplete requirements.

4. Experimental research is mainly required to endorse consequences and assurance that the proposed AI-driven requirement prioritization research model can be applied to various real-world organizational sizes.

4. Objectives

- To classify software requirements or needs using machine learning models.
- To prioritize and rank software requirements using optimization-based scoring.
- To validate critical software requirements based on AI-driven classification and prediction.

5. Limitations of the Study

The data for study was extracted in Comma-Separated Values (CSV) dataset format from secondary sources.

The requirement analysis, elicitation, classification, prioritization, and ranking are the only aspects for the study.

There may be differences in parameters and the resultant requirement classification priority ranks.

6. Research Methodology

The experimental research with a create-and-design strategy focuses on designing an AI-driven optimization model for software project requirement prioritization and ML based requirement classification. Dynamic real-time adjustment to priority ranks and re-prioritization learns from its own incremental data and changing needs are met in a timely manner. Mathematical optimization using AI and ML reduces manual bias chances and provides risk-adjusted data driven reliable intelligent decisions. As model considers stakeholder's priority and emergency input dynamically, it can assess the impact of changing needs and can be quickly validated and absorbed into the final essential, desirable, and deferred set of requirements for implementation. Table 1 illustrates key challenges in software requirements prioritization, ML-

driven solutions, and their alignment with the objectives of the proposed model.

Table 1

ML-Driven Requirement Prioritization Model

	Challenges in Conventional Prioritization	**Suitability**	**Solution aims to**
Scalability	Inefficient for large requirement sets.	ML processes complex and large requirement datasets quickly without human intervention.	Build a scalable Software Requirements Prioritization and Change Management Model framework to classify,
Dynamic Changes	Market, technical, or stakeholder changes; limited adaptability	ML models can learn dynamically changed instances.	Apply dynamic, adaptive and real-time adjustments to priority ranks and re-prioritize in the SRPCMM
Personal Bias and Subjectivity	Manual judgment of personal bias and subjectivity.	algorithms of Data-driven AI/ML models reduce bias	Supervised ML and risk-adjusted scoring.
Stakeholder and User Alignment	Contradictory and Incomplete priorities make accord difficult.	Optimization can aggregate and balance multi-stakeholder input quantitatively.	Improve AI-Driven decision-making support in SRPCMM
Non-Functional & Vague Requirements	Difficult to quantify; often deprioritized or deferred in spite of significance.	Classifies requirements accurately.	AI for context-aware analysis, classification, and priority scoring and ambiguity reduction.
Manual Effort	Requires manual, time-consuming workshops, reviews, and expert evaluation sessions.	Automates requirement classification and prioritization.	Automation improving traceability

Source: Compiled by Authors

This ML-driven classification, prioritization, and management approach effectively addresses scalability challenges in large and complex projects. This model updates runtime prioritization rankings, supports dynamic changes, and effectively handles evolving or changing requirements. Using ML-based algorithms with risk-adjusted scores reduces inconsistency and bias in requirement decision-making. It can resolve prioritization conflicts by aggregating inputs from multiple stakeholders. ML-based algorithms can achieve data-driven consensus, which aligns with stakeholder needs more effectively. Trained input requirements and responses are used to classify requirements into essential, desirable, and deferred or optional categories. This permits filtering non-functional and unambiguous requirements from unstructured text data. The AI-driven optimization model aids requirement prioritization automation and provides resource, cost, time, and manual effort optimization.

Data Source

Requirement Dataset Size- 977 requirements in .csv format

Source**:** https://www.kaggle.com/datasets/iamvaibhav100/software-requirements-dataset

This study includes industrial software requirements dataset (977 requirements)

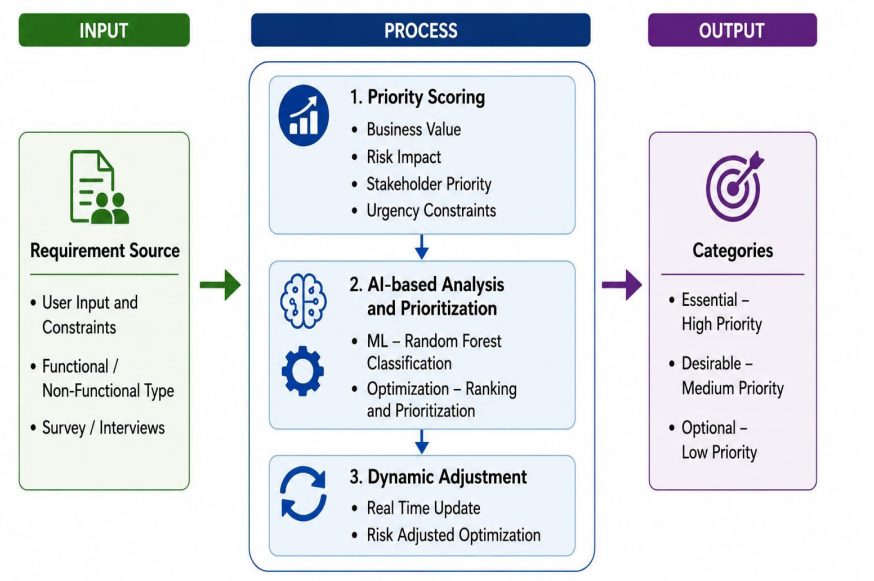
Scope of Research Work

Traditional techniques in requirement engineering and prioritization are inefficient to classify, prioritize and validate requirements in context of human expertise context. The challenges persist and grow with the size and complexity of the system. Changing or evolving needs and expectations of stakeholders need to be traced and satisfied quickly. As the process initiates with requirement elicitation right from scratch, there is an opportunity to enhance elicitation to optimize carry forward of deferred expectations. Systemized efforts and stepwise AI-driven requirement elicitation, planning, analysis and validation results in successful and qualitative project completion. The ML-based classification and decision Optimization model for requirement prioritization considers the multi-criteria input. Challenges in requirement analysis and management are addressed by AI's engagement in software development phases. Software tools for the implementation of SRPCMM are built, and the SRPCMM

model uniquely applies dynamic reprioritization by integrating NLP and optimization techniques.

Software Requirements Prioritization and Change Management Model

This integrated approach reduces ambiguity and enhances requirement classification and prioritization. It focusses on criteria evaluation and rule-based optimization model, Priority Score is computed for weighted criteria value estimation integrating importance of core functionality, time sensitivity, risk exposure and human judgement value. It estimates requirement priority score for given dataset based on weighted aggregation of criteria filtered data values and stakeholder input provided. So, the model addresses scalability to any size of real-world requirement set. The proposed SRPCMM depicted in Figure 1 demonstrates how criteria input requirements like source, user input, time input, function input, hardware software requirement and operational environment can



Source: Author's own compilation

Figure 1

ML Framework for Software Requirements Prioritization and Change Management Model. (SRPCMM)

be captured and pre-processed for further ML-based classification, prioritization and implementation. Machine learning models are more precise and scalable than traditional rule-based approaches in classifying software requirements into Essential, desirable, and deferred categories.

Predictive machine learning algorithms can efficiently evaluate different criteria as shown in Table 2 to rank software requirements, increasing the precision of decision-making, and lowering stakeholder effort. By combining the categorization and prediction results from machine learning models, a recommendation system may accurately pinpoint and indicate essential software needs, increasing stakeholder satisfaction and project efficiency.

Algorithm

1. Start
2. Requirement Collection Elicitation
3. Requirement Pre-processing using Natural Language Processing
4. ML based Classification
5. Optimization based Priority weighted Score Estimation
6. Ranking, Prioritization and Recommendation.
7. Stop

Rule Specification Criteria used

If any requirement directly is part of core functionality or is required to be mandatory, then business criticality will be assigned a high score of '5'. If requirement enhances efficiency, performance or user experience but is not mandatory then business criticality score assigned to medium range is 4 and third lowest business criticality score '3' is assigned to optional or may-or-may-not-be-included requirement.

Table 2

Criteria Used for Rule Specification

Sr. No.	Criteria	Context	Need of Requirement	Class	Score
1	Business Criticality	core system functionality, revenue generation, or regulatory compliance	Critical functionality and mandatory	High	5
			enhances system efficiency or user experience but not mandatory	Medium	4
			May be included or Optional	Low	3
2	Risk Impact	Potential of unexpected consequences	Security breaches. Legal or system failure	High	5
			affects performance or reliability without critical damage	Medium	4
			minimal operational or legal impact	Low	3
3	Urgency	Reflects the time sensitivity based on deadlines, dependencies, or release cycles	needed immediately	High	5
			Needed but not immediately	Medium	4
			Can be postponed	Low	3
4	Stakeholder Priority	importance demanded by stakeholders	Strong Demand by Authority	High	5
			Demand by few stakeholders but not all	Medium	4
			Limited interest	Low	3

Source: Compiled by Authors

Similarly, risk impact for critical negative consequences like system failure, security breaches or system failure specifies high score'5', the risk only affects performance or reliability without higher damage to system, security or doesn't create legal issue is assigned to medium score '4'. Low or minimal or negligible risk impact is assigned to score '3'. Time sensitivity is measured and dependent based on deadlines, so if requirement needed to complete immediately are urgent so assigned to High Class with Urgency Score '5', followed by needed but not immediately with medium class score 4 and can be postponed assigned to no urgency that is Low Class Urgency Score is '3'. Stakeholder Priority is Human component set [20] i.e. if any requirement is strongly demanded by any authority or expert user then assigned to class High with Stakeholder Priority score '5', demanded by few but not by all or authority will be assigned to medium class with score '4', and if less importance of requirement recognized by limited interest will be Low with Score '3'.

Priority Class Scoring logic and Formula

The formula for score logic is derived using each criterion' weight score into assigned critical value. The prioritization score was calculated using a **weighted formula**:

Weighted Prioritization Class Score Formula

$$PSi = WBC \times BCi + WSP \times SPi + WRI \times RIi + WU \times Ui$$

Weighted Prioritization Class Score *= (Weight for Business Criticality ×Business Criticality Value) + (Weight for Stakeholder Priority × Stakeholder Priority Value) – (Weight for Risk Impact × Risk Impact Value) + (Weight for Urgency × Urgency Value)*

This formula reflects a practical approach where **business criticality value, Risk Impact, time urgency** and **stakeholder priority** value are given more weight, while **risk** negatively influences the score. Feature complexity has a smaller positive influence, acknowledging its relevance but not overemphasizing it. Weights are expert defined to include human expert inclusion.

ML based Classification Rule for Decision Prioritization

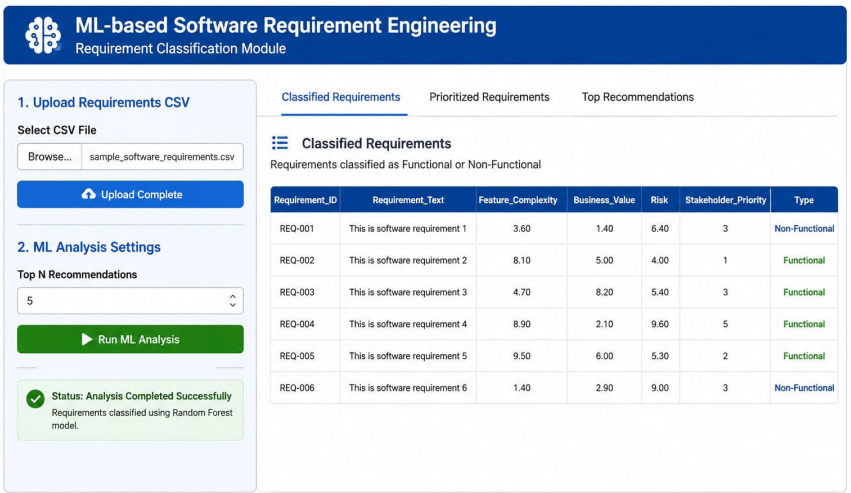
$$Priority\ Score\ Classification = \begin{cases} Essentital\ Requirement,\ \ High\ PS_i \\ Desirable\ Requirement,\ Medium\ PS_i \\ Optional\ Requirement,\ Low\ PS_i \end{cases}$$

So, the given formula will verify the estimated threshold and compare as per priority classification rule to determine the fitness of membership of class.

7. Results

Requirements Classification mode

The classification model represented in Figure 2 successfully labelled each requirement as either **Functional** or **Non-Functional** based on the input features.



Source: Compiled by Authors

Figure 2

Requirement Classification

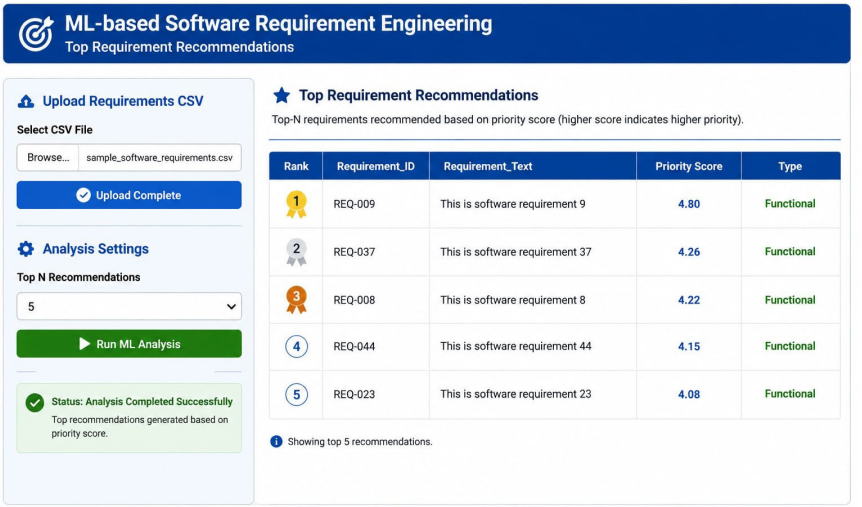
Out of six requirements:

- 4 requirements were classified as Functional
- 2 requirements were classified as Non-Functional

Top Recommendations

The system depicted in Figure 3 effectively filtered and displayed the Top 5 recommended requirements, guiding development planning and resource allocation. High-priority items received immediate attention, while low-complex requirements received lower prioritization. The risk

penalization logic demonstrated effectiveness, with high-risk items deprioritized.



Source: Compiled by Authors

Figure 3

Top Requirement Recommendations

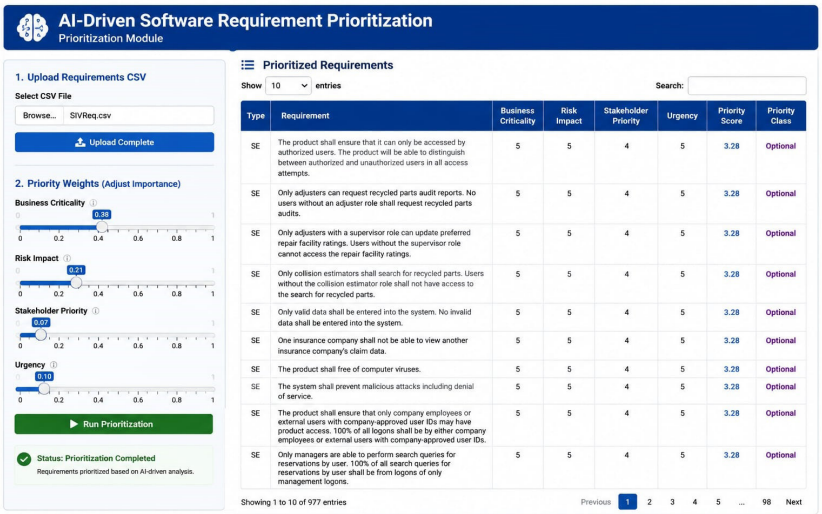
8. Analysis

This classification enables project stakeholders to understand the nature of each requirement better, facilitating teams in assigning responsibility and determining the technical versus operational effort required.

The model aligns with real-world software engineering expectations, and non-functional requirements like REQ-001 and REQ-006 were identified and managed separately. The model provides clarity across classification, prioritization, and actionable top N selection, ensuring a clear understanding of requirements.

9. Discussions

AI driven software requirements optimization model in Figure 4 uses criteria-based ML rule system as specified is used to automate software requirement classification in functional and non-functional requirements and prioritization process filtering top N requirements based on priority



Source: Compiled by Authors

Figure 4

AI Driven Software Requirements prioritization and Classification

scoring ranks. This automation reduces manual efforts, mistakes, time and cost required for traditional approaches. Poorly specified or ill-defined requirements are always difficult to classify and prioritize using traditional techniques.

10. Conclusion

This research article introduces a new ML based optimization model named Software Requirements Project and Change Management Model (SRPCMM). SRPCMM solves the issues and challenges of uncertainty, non-scalability, and non-adaptability. The ML based classification and Optimization based- Priority Score results in an enormous solution for the classification, prioritization, and recommending of software requirements. Dynamic Adjustment and Real Time Update for Risk adjusted Optimization responsive [21] to both dynamic and stakeholder-centric requirements. The experimental results classify requirement in Essential (High Priority), Desirable (Medium Priority), and Optional (Low Priority) classes, Automated requirement tracing reduces manual intervention and gives accurate and quick results. The capability of the model to manage non-

functional, vague, and changing requirements is a clear demonstration of its utility in real-life scenarios. Moreover, SRPCMM ensures stakeholders' contentment through the provision of a user-friendly interface where decision-making is open and feedback is quick, thereby making it easy for Top-N recommendation and risk-adjusted prioritization. This work has provided solutions to the significant gaps that traditional techniques, such as AHP, MoSCoW, and value-based models, have left unaddressed, particularly in the case of large, evolving software projects. It demonstrates that AI-driven requirement engineering can reduce manual mistakes and simultaneously improve the levels of automation, traceability, green testing and adaptability [22].

The future research work should focus on the integration of fuzzy logic, expert systems, and reinforcement learning for real-time and large projects requirement prioritization. Besides this, scalability, robustness, and class prioritization for cross domain project applicability can be investigated.

References

[1] A. Hudaib, R. Masadeh, M. H. Qasem, and A. Alzaqebah, "Requirements prioritization techniques comparison," *Mod. Appl. Sci.*, vol. 12, no. 2, pp. 62–73 (2018), doi: 10.5539/mas.v12n2p62.

[2] J. S. N. Yeow, M. E. Rana, and N. A. A. Majid, "An Automated Model of Software Requirement Engineering Using GPT-3.5," in *Proc. 2024 ASU Int. Conf. Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS)*, pp. 1746–1755 (Jan. 2024), doi: 10.1109/ICETSIS61505.2024.10459458.

[3] F. A. Bukhsh, Z. A. Bukhsh, and M. Daneva, "A systematic literature review on requirement prioritization techniques and their empirical evaluation," *Computer Standards & Interfaces,* vol. 69, Art. no. 103389 (2020), doi: 10.1016/j.csi.2019.103389.

[4] M. I. Babar, M. Ghazali, D. N. Jawawi, S. M. Shamsuddin, and N. Ibrahim, "PHandler: an expert system for a scalable software requirements prioritization process," *Knowl.-Based Syst.*, vol. 84, pp. 179–202 (2015), doi: 10.1016/j.knosys.2015.04.010.

[5] P. Avesani, C. Bazzanella, A. Perini, and A. Susi, "Facing scalability issues in requirements prioritization with machine learning techniques," in *Proc. 14th IEEE Int. Requirements Engineering Conf.*, pp. 297–305 (2005), doi: 10.1109/RE.2005.53.

[6] 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," *Inf. Technol. Manag.*, vol. 24, no. 3, pp. 195–219 (2023), doi: 10.1007/s10799-022-00364-w.

[7] J. A. Khan, I. U. Rehman, Y. H. Khan, I. J. Khan, and S. Rashid, "Comparison of requirement prioritization techniques to find best prioritization technique," *Int. J. Mod. Educ. Comput. Sci.*, vol. 7, no. 11, pp. 53–60 (2015), doi: 10.5815/ijmecs.2015.11.06.

[8] P. Achimugu, A. Selamat, R. Ibrahim, and M. N. R. Mahrin, "A systematic literature review of software requirements prioritization research," *Inf. Softw. Technol.*, vol. 56, no. 6, pp. 568–585 (2014), doi: 10.1016/j.infsof.2014.02.001.

[9] M. Shinde-Pawar and D. Sahasrabuddhe, "Software requirement scaling using fuzzy logic," *International Journal of Application or Innovation in Engineering & Management*, vol. 3, no. 3, pp. 564–568 (Mar. 2014).

[10] U. Samal, S. Kushwaha, G. Nain, S. Singh, S. Usmani, and A. Kumar, "Necessity of fuzzy logic: Trends in software reliability assessment," in *Reliability Assessment and Optimization of Complex Systems.* Amsterdam, The Netherlands: Elsevier (2025), doi: 10.1016/B978-0-443-29112-8.00009-8.

[11] P. Avesani, C. Bazzanella, A. Perini, and A. Susi, "Supporting the requirements prioritization process: A machine learning approach," in *Proc. 16th Int. Conf. Software Engineering and Knowledge Engineering (SEKE)*, Banff, AB, Canada, pp. 306–311 (Jun. 2004).

[12] A. Perini, A. Susi, and P. Avesani, "A machine learning approach to software requirements prioritization," *IEEE Trans. Softw. Eng.*, vol. 39, no. 4, pp. 445–461 (2013), doi: 10.1109/TSE.2012.52.

[13] N. Richardson, C. Bazzanella, A. R. Onteddu, R. R. Kundavaram, and R. R. Talla, "AI-driven optimization techniques for evolving software architecture in complex systems," *ABC J. Adv. Res.*, vol. 12, no. 2, pp. 71–84 (2023), doi: 10.18034/abcjar.v12i2.783.

[14] H. Arshad, S. Shaheen, J. A. Khan, M. S. Anwar, K. Aurangzeb, and M. Alhussein, "A novel hybrid requirements prioritization approach based on critical software project factors," *Cognit. Technol. Work*, vol. 25, no. 2, pp. 305–324 (2023), doi: 10.1007/s10111-023-00729-3.

[15] J. Lunze, Ed, *Handbook of Hybrid Systems Control: Theory,* Tools, *Cambridge University Press (2011),* doi: 10.1017/CBO9780511807930.

[16] L. A. Palinkas, S. J. Mendon, and A. B. Hamilton, "Innovations in mixed methods evaluations," *Annu. Rev. Public Health*, vol. 40, no. 1, pp. 423–442 (2019), doi: 10.1146/annurev-publhealth-040218-044215.

[17] N. Ryan, D. Vieira, J. Gyamfi, T. Ojo, D. Shelley, O. Ogedegbe, J. Iwelunmor, and E. Peprah, "Development of the ASSESS tool: A comprehenSive tool to Support rEporting and critical appraiSal of qualitative, quantitative, and mixed methods implementation reSearch outcomes," *Implementation Science Communications*, vol. 3, no. 1, Art. no. 34 (Mar. 2022), doi: 10.1186/s43058-021-00236-4.

[18] J. F. Molina-Azorin, "Mixed methods research in strategic management: Impact and applications," *Organizational Research Methods,* vol. 15, no. 1, pp. 33–56 (Jan. 2012), doi: 10.1177/1094428110393023.

[19] M. Shabbir, R. K. Yadav, M. W. Khan, and H. Singh, "Empirical analysis of computationally intelligent technique for software risk prediction," *J. Cybersecurity Inf. Manag.*, vol. 17, no. 2, pp. 200–209 (2026), doi: 10.54216/JCIM.170214.

[20] M. Nazim, M. Arif, C. W. Mohammad, and M. Sadiq, "Generating large dataset for software requirements prioritization and selection under fuzzy environment," *J. Inf. Optim. Sci.*, vol. 44 no. 2, pp. 285–299 (2023), doi: 10.47974/JIOS-1290.

[21] V. Ramisetty and S. S. Saheb, "Leveraging AI for sustainable risk assessment in the insurance industry," *J. Inf. Optim. Sci.*, vol. 46, no. 8, pp. 2499–2510 (2025), doi: 10.47974/JIOS-2067.

[22] M. K. Singhal and C. Gunawat, "AI-driven green testing: Optimizing efficiency and sustainability in software testing," *J. Inf. Optim. Sci.*, vol. 46, no. 4-B, pp. 1347–1356 (2025), doi: 10.47974/JIOS-2001.

Received December, 2025