AI-Driven Prioritization Techniques of Requirements in Agile Methodologies: A Systematic Literature Review

Aya M. Radwan, Manal A. Abdel-Fattah, Wael Mohamed
Information System Department, Faculty of Computers and Artificial Intelligence, Cairo, Egypt

***Abstract*—Software requirements are the foundation of a successful software development project, outlining the customer's expectations for the software's functionality. Conventional techniques of requirement prioritization present several challenges, such as scalability, customer satisfaction, efficiency, and dependency management. These challenges make the process difficult to manage effectively. Prioritizing requirements by setting criteria in order of importance is essential to addressing these issues and ensuring the efficient use of resources, especially as software becomes more complex. Artificial intelligence (AI) offers promising solutions to these challenges through algorithms like Machine Learning, Fuzzy Logic, Optimization, and Natural Language Processing. Despite the availability of reviews on conventional prioritization techniques, there is a notable gap in comprehensive reviews of AI-based methods. This paper offers a systematic literature review (SLR) of AI-driven requirements prioritization techniques within Agile methodologies, covering 32 papers published between 2010 and 2024. We conducted a parametric analysis of these techniques, identifying key parameters related to both the prioritization process and specific AI methods. Our findings clarify the application domains of various AI-based techniques, offering crucial insights for researchers, requirement analysts, and stakeholders to choose the most effective prioritization methods. These insights consider dependencies and emphasize the importance of collaboration between stakeholders and the development team for optimal results.**

***Keywords*—Requirement analysis; requirement prioritization; agile; fuzzy logic; machine learning; optimization**

I. Introduction

Requirement Engineering (RE) is a crucial component of software engineering. It involves identifying and understanding client needs, the contexts in which the system will be developed, modeling, analyzing, prioritizing, and documenting stakeholder requirements. RE ensures that these documented requirements align with agreed-upon specifications and manages the evolution of changing requirements [1], [2], [3].

Requirement Engineering (RE) faces numerous challenges, particularly in the areas of human communication and collaboration, and understanding and clarifying requirements. Effective communication and collaboration between project teams and customers are critical, yet often fraught with difficulties, including conflicts among stakeholders and the need for active involvement from all parties. Understanding and clarifying requirements present additional challenges, as ensuring high-quality requirements and well-defined user stories can be complex and time-consuming [4], [5].

Agile Software Development (ASD) introduces specific challenges for requirements prioritization (RP) techniques, such as stakeholder conflicts, changes in priority lists leading to rework, and factors influencing requirement selection during the RP process. Among the most significant challenges are managing and coordinating distributed teams, prioritizing requirements, maintaining proper documentation, adapting to changing and over-scoping requirements, and effectively organizing processes while monitoring progress and incorporating feedback. Addressing these challenges requires robust strategies and tools to enhance communication, clarify requirements, and streamline prioritization and management processes [6].

A key aspect of Requirement Engineering (RE) is requirements prioritization (RP). As the name implies, RP involves identifying the most critical requirements for implementing a successful system. It is an iterative process involving complex decision-making activities that support the development of a high-quality system within defined constraints. RP ensures the correct ordering of requirements based on stakeholder perceptions, by rearranging them according to various criteria such as importance, cost, penalty, and risk. Active stakeholder involvement is crucial for achieving accurate prioritization results [7].

While addressing the prioritization challenge, these techniques have been applied without considering the hierarchical dependencies among requirements, such as stakeholder needs and their derived requirements. Derived requirements are the detailed requirements extracted from stakeholder needs, often in the form of use cases or non-functional requirements [6] [8].

Most existing requirement prioritization techniques lack scalability, dependability, continuous prioritization, rank updating, feedback handling, and comprehensive implementation of methods or algorithms. In Agile development, where requirements change rapidly, a continuous requirement prioritization process is essential [8] [9].

The objective of this paper is to systematically review and analyze AI-driven techniques, such as Fuzzy Logic, Machine Learning, NLP, and Optimization, for requirements prioritization in Agile methodologies. It aims to compare these techniques based on their strengths, weaknesses, and

effectiveness in addressing challenges like scalability, stakeholder collaboration, and requirement dependencies. Additionally, the paper seeks to evaluate the impact of AI on improving the prioritization process and provide recommendations for future research.

This section presents the idea of the requirement prioritization in Agile methodology and discusses the common challenges. In the second section the background of the used techniques to prioritize the requirements in Agile methodology is presented. In the third section the research method is described, and the requirements prioritization analysis is illustrated at the taxonomy, the research questions are defined and research strategy. The fourth section provides the surveyed techniques and reviews the 32 papers that were sorted in subsection by the type of technique. In section five the Evaluation criteria are defined by clearing the strengths, weakness and limitations of the research. Finally, a comprehensive of the evaluation criteria is presented in the tables for each technique.

II. Background

Requirement prioritization in Agile methodology can be improved using advanced techniques such as Fuzzy Logic, Machine Learning, Unsupervised Learning, Optimization, and Natural Language Processing (NLP). In this section, an illustration of the four techniques is defined.

Fuzzy Logic has the unique ability to process numeric input and linguistic information simultaneously. It applies a nonlinear transformation to the input feature vector, resulting in a single numeric output, thus transforming numeric values into other numeric values [10]. Defining requirements into precise numerical values is a significant challenge. Fuzzy Logic allows for the accurate definition of concepts such as low cost, high quality, and high progress, making it a powerful tool for managing uncertainty. It does this by using human language and allowing for interpretation of assertions that are not entirely precise or incorrect [11].

Machine Learning (ML) provides a range of algorithms and strategies to imitate the way of human think by acquiring knowledge from data, it can be used to address software engineering challenges using supervised, unsupervised, and reinforcement learning approaches. Machine Learning facilitates software development by utilizing predictive models to make informed decisions on algorithms and features. The process of applying Machine Learning requires understanding the problem at hand, collecting relevant data, performing preprocessing tasks, and conducting comprehensive evaluations [12]. Software requirements engineering, and Machine Learning are major areas of research, where Machine Learning is applied in many software engineering procedures. Machine Learning approaches, such as text feature extraction and algorithms, are employed to categorize and prioritize software needs, utilizing the large amount of data and domain expertise gathered throughout the development process [12]. Natural Language Processing is interconnected to Machine Learning (ML) as NLP heavily relies on Machine Learning approaches, utilizing algorithms and models derived from ML. The objective of Natural Language Processing (NLP) or computational linguistics is to devise algorithms and methodologies that construct computational models with the ability to analyze natural languages. These models are specifically built to carry out important functions, such as enabling the exchange of information between humans and machines, improving communication among individuals, or simply analyzing and interpreting text or speech [13].

Optimization is a strategy to select the most efficient solution from a set of options, considering constraints and intended results. By considering aspects such as minimizing costs or maximizing efficiency. Optimization is a commonly employed strategy in diverse fields such as mathematics, computer science, engineering, economics, and operations research. It improves decision-making processes and successfully solves complex challenges [14].

In Agile methodologies, various AI techniques offer unique solutions to key challenges in requirements prioritization. Fuzzy Logic stands out in managing uncertainty and imprecise requirements by allowing for flexible decision-making under ambiguous conditions, thus improving the overall accuracy of decisions [11]. Machine Learning (ML) techniques are particularly effective in handling large datasets and automating the prioritization process, which addresses scalability concerns and significantly reduces manual effort [12]. Natural Language Processing (NLP) plays a crucial role in automating the interpretation of user stories and feedback, enhancing communication and resolving issues related to unclear or incomplete requirements. Finally, Optimization algorithms, such as AHP and PSO, provide a structured approach to balancing conflicting priorities, making them highly effective in resolving stakeholder conflicts and managing competing requirements efficiently [15][16]. By applying these techniques, the prioritization of requirements becomes more efficient, accurate, and adaptable to the dynamic nature of Agile software development.

III. Research Method

A. Planning the Review (Preparation Stage)

This Systematic Literature Review aims to investigate, analyze, and summarize Agile software requirements prioritization techniques, including Fuzzy Logic, Machine Learning, Natural Language Processing, and Optimization within the framework of parametric benchmarks. The review aims to define the key strengths and weaknesses and define how artificial intelligence techniques can affect requirement prioritization in Agile methodology, also explore the limitations of applying these techniques in the Agile prioritization process. Additionally, the study seeks to understand the significance of these techniques in the software development process, the role of stakeholders, and the challenges or limitations in Agile practices, while providing future directions for further research. The main objective of this Systematic Literature Review (SLR) is to identify the various techniques employed to prioritize requirements in Agile methodology.

The key of the SLR is illustrated through three phases at the taxonomy depicted in Fig. 1, to illustrate the core ideas of the paper. First, defining with requirements prioritization techniques as Fuzzy Logic, Machine Learning, Natural Language Processing and Optimization. Second, comparative

analysis presented in a table to show the strength, weakness and limitations in requirements prioritization for the 32 papers. Third, the evaluation criteria to determine the effectiveness of each requirements prioritization technique.

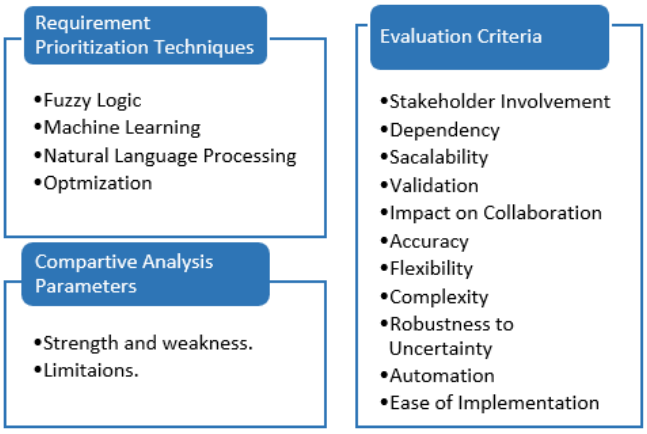

Fig. 1. Requirements prioritization taxonomy.

B. Research Question

The following research questions have been defined:

RQ1: What requirement prioritization techniques used in Agile Software Development in the research literature?

RQ2: How are the proposed techniques evaluated?

RQ3: What evaluation criteria can be used in requirement prioritization in Agile Software Development?

RQ4: What are the limitations in the current Agile Software Development technique?

RQ5: Does dependency play a role in prioritizing requirements in Agile methodology?

RQ6: Is collaboration with stakeholders considered when prioritizing requirements in Agile methodology?

RQ7: Is scalability considered when prioritizing requirements in Agile methodology?

C. Search Strategy

A comprehensive search strategy is developed to collect all research articles that are relevant to the domain of our study from a variety of online resources. The most renowned online databases and combined key terms was selected to produce search strings. Subsequently, the search was implemented to identify all relevant articles. 900 prospective studies were identified during the initial search phase. 250 studies were determined to be relevant to requirement engineering in general after the titles and abstracts were reviewed. The selection criteria focused on 32 papers specifically applying AI techniques, such as Machine Learning, Fuzzy Logic, NLP, and Optimization, to requirements prioritization in Agile software development. Studies published between 2010 and 2024 were included to ensure the review reflected the most current research. Papers were chosen based on their application of AI techniques within Agile methodologies like Scrum, Kanban, XP, and DevOps. After a thorough quality assessment, these 32 papers were selected to address the research questions, with no additional recent studies identified.

IV. Surveyed Techniques

In this section, all the surveyed techniques were reviewed. Software organizations are addressing the limitations of conventional techniques by implementing AI-based methods for software requirements prioritization in Agile. These techniques are divided into four categories: Optimization-based, Fuzzy Logic-based, Machine Learning-based, and Natural Language Processing-based ones. A detailed review is provided of each technique within these groups individually.

A. Fuzzy Logic

Borhan in [15] introduces an innovative approach to prioritizing user stories by collaborating with stakeholders in the Agile-Scrum prioritization process. The approach implements Fuzzy Logic operations to prioritize stakeholders according to predetermined criteria, thereby facilitating a more comprehensive comprehension of their contributions and concerns. This methodology is applied to an ATM system's requirements and their related user stories. A group of software experts who are experienced in Agile methodologies give feedback within their organizations for this approach. The findings suggest that including stakeholder analysis has a positive influence on the prioritization process, demonstrating the effectiveness of this method in achieving a balance between functional and non-functional user stories. By carefully considering stakeholder perspectives, this approach enhances the accuracy and efficacy of user story prioritization in Agile-Scrum projects. It guarantees a more equitable assessment of the system's requirements and the diverse demands of users.

Abusaeed in [16] suggest a quantitative framework that effectively prioritizes the identified cost factors based on the following four categories: people, initiatives, processes, and products. They conduct a systematic literature review and empirical study within the ASD context to identify and validate the cost overhead factors. Similarly, the validated factors are classified and prioritized in the present study by employing a multi-criterion decision-making Fuzzy-Analytic Hierarchy Process technique. This method effectively mitigates the subjectivity and ambiguity of the identified factors. The implementation results provide a prioritized list of cost overhead factors that would be beneficial to Agile practitioners in the context of Agile Software Development.

Ottoli in [17] introduces a novel approach to requirement prioritization that uses expert opinions presented by fuzzy linguistic labels on multiple decision criteria. Fuzzy linguistic labels are used to allocate weights to each expert and criterion, and these opinions are aggregated using a majority-guided linguistic IOWA (Induced Ordered Weighted Averaging) operator. The method contrasts requirements by comparing the aggregated expert opinions and their weighted importance. The algorithm demands users to submit opinions. However, additional empirical validation with actual users is required. Allowing selective expert opinions, incorporating consensus metrics, and investigating other (IOWA) Induced Ordered Weighted Averaging operators and T-norms are all potential future enhancements.

B. Machine Learning

Anand in [18] used the apriori algorithm to prioritize requirements. They addressed the limitations of the traditional methods used such as MoSCoW, Validated Learning, Walking Skeleton, and Business Value which frequently fail to effectively resolve stakeholder conflicts. The apriori algorithm is implemented to identify the most frequently requested requirements, to reduce conflicts between stakeholders by prioritizing requirements based on stakeholder input by continually performing join and prune functions to identify frequent items within a database of transactions.

Hudaib in [19] employs Self-organizing Maps (SOMs), a type of Unsupervised Neural Network as a method to prioritize requirements within Agile methodologies. This method is designed to classify and prioritize requirements automatically by applying patterns that are derived from historical data and stakeholder feedback. The SOM technique prioritizes requirements by grouping similar requirements based on their historical significance and characteristics. This enables project teams to identify which requirements should be addressed first. They provide a visualization to understand the priority of requirements.

Varsha in [20] improved the decision-making process of the requirements by grouping stakeholders according to their interests and perspectives. They aimed to improve the accuracy and efficiency of requirements prioritization in complex project environments. When requirements values are gathered from many individual stakeholders, it is necessary to organize stakeholders into groups to implement requirements prioritization. They involve the use of a hierarchical clustering analysis technique to create such groups from stakeholder ratings. The group weights were determined using AHP.

Belsis in [21] introduces "PBURC", a Patterns-Based, Unsupervised Requirements Clustering Framework. It utilizes Machine Learning to effectively manage data inconsistencies and validate requirements, ensuring that requirements are clustered properly using K-means. This method defines the optimal number of sprints by collaborating with stakeholders. The framework seeks to simplify the development process by prioritizing client requirements and protecting business value, despite the complex nature of distributed development environments.

Achimugu in [22] focuses on addressing the problem of prioritizing many software requirements using the k-means clustering technique. K-means is utilized to classify requirements based on the weights of their attributes, provided by project stakeholders. The effectiveness of this method was validated with the RALIC dataset, revealing that different stakeholder weights influence the clustering outcome. The approach was further refined by employing a synthetic method with scrambled centroids, proving effective in prioritizing requirements. The results indicated that this technique enhances the scalability and reliability of requirement prioritization, successfully addressing previous limitations such as rank reversals and disparity in weighted rankings.

Kumar in [23] designed an approach to enhance Agile methodology by clustering user stories using the k-means algorithm and cosine similarity. The process includes preprocessing steps like tokenization, stop-word removal, stemming, and lemmatization, followed by clustering based on similarity measures, and validating cluster cohesion with silhouette coefficient values. Experimental results show that cluster quality is improved as the number of clusters (k) increases and this reflects effectively reducing the time required for requirement implementation.

C. Natural Language Processing

Sachdeva in [24] aims to propose an approach that effectively balances both business value and process flow in the prioritization process. This approach helps the Product Owner in deciding which requirements to prioritize and how to organize them in the Product Backlog. This approach models requirements iteratively through user stories to align with the rapidly changing business needs and with the continuous change of requirements. The Product Owner (PO) typically prioritizes based on Business Value without considering dependencies of user stories. They used UML Activity Diagrams to visualize process flows, then they converted user requirements into these diagrams using NLP. This visualization helps prioritize stories based on their process flow.

Shafiq in [25] introduces the NLP4IP approach, a recommendation system to prioritize issues such as stories, bugs, or tasks. It is a semi-automatic approach that uses Natural Language Processing (NLP) to address prioritization challenges to create a recommendation model. The rank of newly added or modified issues is dynamically predicted by this model. The JIRA issue tracking software was employed to evaluate the approach across 19 projects from 6 repositories, resulting in a total of 29,698 issues. Furthermore, they implemented a JIRA plug-in that illustrates the predictions generated by the new Machine Learning model.

Sami in [26] introduced a tool that integrates OpenAI, Flask, and React to automatically generate and rank user stories based on core requirements to improve the project management workflows. The tool allows users to input the requirements then it's used to generate and prioritize user stories and epics. The tool converts responses into a user-friendly JSON format and supports CSV downloads then the tool integrates with task management platforms such as JIRA and Trello.

Sharma in [27], a proposed approach that employs Natural Language Processing (NLP) algorithms to categorize user experiences that are similar and organize them into project releases. The goal of this approach is to enhance the release planning process for intricate and substantial software initiatives that can be simplified. The method entailed the initial development of a word corpus for each project release, followed by the conversion of user stories into vector representations using Java utilities. Lastly, the RV coefficient NLP algorithm is employed to organize them into distinct software releases. Furthermore, these algorithms can be integrated into commercial tools such as JIRA and Rally to facilitate enhanced release planning in Agile environments.

Kifetew in [28] proposes ReFeed approach. It aims to provide a more accurate and automated way to prioritize requirements based on user feedback. The approach employs

Natural Language Processing (NLP) to prioritize requirements. The approach extracts and propagates quantifiable properties from related user feedback. They used domain knowledge to bridge the vocabulary gap between users and developers. This approach bridges the gap between end-users' words and developers' words to formulate requirements.

D. Optimization

Asghar in [15] presents a methodology that integrates traditional prioritization factors with contemporary metrics and techniques, such as AHP and MOSCOW, as well as ISO standards. The objective of this method is to enhance the quality of both the process and the product by offering a more comprehensive and consistent framework for prioritizing requirements. Additionally, the methodology aims to improve the quality of requirements that are selected and prioritized during SCRUM sessions. The proposed model was validated through the use of comprehensive simulations with the iThink software.

Kumar in [16] identifies eight critical attributes—such as leadership support, human resources, and information technology—through literature review and expert validation. To rank these attributes effectively, the study employs an integrated technique called the Analytic Hierarchy Process (AHP) Technique for Order Preference by Similarity to Ideal Solution (TOPSIS). This combined approach allows for a structured evaluation of the attributes based on their relative importance and their distance from an ideal solution.

Muhammad in [17] applied Multi-Criteria Decision Making (MCDM) techniques to rank non-functional requirements (NFRs). The study integrates Fuzzy Logic to manage the imprecision in NFR evaluations and employs pairwise comparison to establish the relative importance of various NFRs. The effectiveness of the proposed model was validated through a case study involving Agile projects, which demonstrated its capability to improve the prioritization process by accurately addressing and ranking NFRs.

Agrawal in [18] proposes an Agile-based Risk Rank (AR-Rank) method to prioritize risk factors in Agile methodology. This method is applied by using Particle Swarm Optimization (PSO) for iterative Optimization. The method provides precedence ranking of risks to minimize their impact and ensure timely delivery of risk-free software.

The AR-Rank approach is validated against other methods and tested on ten real-life projects.

Prakash in [19] proposed ARP–GWO method, this method uses grey wolf Optimization and the k-means clustering algorithm. They aim to prioritize risks in Agile Software Development and to enhance quality, reduce costs, and improve delivery times. The Experimental demonstrates five industrial projects that demonstrate ARP–GWO's effectiveness and high satisfaction among developers and users.

Chaves in [20] presents a Mult-objective swarm intelligence presents a multi-objective swarm intelligence metaheuristic (MOABC) to solve The Next Release Problem (NRP). They present superior performance compared to other methods on real-world datasets. They are seeking to improve the Optimization process by incorporating hybrid approaches, larger datasets, and additional constraints.

Brezočnik in [21] aim to enhance iteration planning in Agile Software Development. By introducing STAPSO, a novel algorithm. This algorithm combines Scrum task allocation and Particle Swarm Optimization. STAPSO is tested on a real-world dataset. It showed promising results in task allocation and applies to various estimation techniques.

Brezočnik in [22] provides an overview of swarm intelligence algorithms. The paper systematically classifies swarm intelligence algorithms based on Agile Software Development tasks like the next release problem, risk, and cost estimation, and discusses their promising results.

Sagrado in [23] address multi-objective Optimization problems in the Next Release Problem (NRP). They proposed approach includes the Ant Colony System (ACS) to multi-objective Optimization, where ants build solutions probabilistically, considering pheromone levels and heuristic information. And Comparative Analysis compared with Greedy Randomized Adaptive Search Procedure (GRASP) and Non-dominated Sorting Genetic Algorithm (NSGA-II) to validate its effectiveness in generating high-quality solutions for requirement prioritization.

Somohano in [24] aims to reduce computational complexity and enhance the precision of requirements prioritization. They present an approach to enhancing the Analytic Hierarchy Process (AHP) by integrating evolutionary computing techniques. Additionally, the research suggests integration of multiple criteria and developing a software tool to assist project managers in the prioritization process more efficiently.

E. Other Approaches

AbdElazim in [9] introduces a comprehensive framework for prioritizing requirements in Agile Software Development, focusing on continuous and scalable prioritization. It effectively manages rapidly changing requirements and their dependencies by integrating early into the Agile process with epics and user stories. This fully integrated approach ensures that the framework can handle requirement changes at any stage of the development cycle, making the Agile development process more adaptable and responsive to evolving project needs but no tool was provided.

Govil in [39] presents a comparative analysis of various requirement prioritization techniques such as AHP, Pair wise comparison, MoSCoW, planning poker, ping pong balls, bubble sort, and others detailing their strengths and weaknesses. The paper highlights discrete parameters that influence the success of software projects and discusses the difficulty of prioritizing requirements in Agile methodologies, where changes can occur late in development. The goal is to provide product owners with insights to select the most suitable prioritization technique based on the project's specific needs and constraints.

Kamal in [40] explores the factors that contribute to the success of Agile Requirements Change Management (ARCM) in the context of Global Software Development (GSD). It follows a two-step approach: first, identifying success factors through a Systematic Mapping Study (SMS) and validating

them with industry practitioners via a questionnaire survey. And the second step is to prioritize these factors using the Analytical Hierarchy Process (AHP). This study shows twenty-one critical success factors for ARCM in GSD, with top priorities being resource allocation at overseas sites, communication, coordination, control (3Cs), process improvement expertise, a geographically distributed change control board (CCB), and continuous top management support. The findings aim to assist practitioners in effectively implementing ARCM activities in GSD settings. The research further needs to develop a comprehensive readiness model for ARCM. This model should identify negative impact factors and collect best practices through multiple case studies.

Kifetew in [41] discusses the use of automated decision-making techniques to help engineers in the process of selecting and prioritizing requirements. Effective involvement of the development team and stakeholders is crucial in the decision-making process enabled by these tools. The paper used Analytic Hierarchy Process (AHP) and Genetic Algorithms to introduce a tool-supported to perform collaborative requirements prioritization process. The tool enables an iterative prioritization process, therefore enabling stakeholders to actively participate in decision-making throughout the process.

Perkusich in [42] presents an approach that leverages intelligent software engineering techniques to enhance requirement prioritization in Agile environments. The approach utilizes a combination of automated tools and Machine Learning algorithms to analyze user feedback, historical project data, and other relevant metrics to prioritize requirements dynamically. This method allows for continuous re-evaluation of priorities based on new data and evolving project needs.

AL-Ta'ani in [43] proposes a conceptual framework for continuous requirements prioritization in Agile development, addressing the challenge of selecting key user requirements for implementation across iterations. The framework is informed by a thorough review of related literature and content analysis of the data. It delineates the critical factors impacting the requirements prioritization process and their effect on the final product, categorizing them into three primary dimensions: environment, process, and product. The environment includes stakeholders' characteristics, constraints relevant to the project, and Requirement Nature; the process outlines the particular processes involved in prioritization, and the Product describes the outcomes. The study highlights the importance of systematic prioritization to prevent costly development errors and potential project failures, suggesting areas for future research to refine prioritization methods and techniques.

Alkandari in [44] discusses three models for requirements prioritization in Agile development. Model 1 focuses on estimating business value using work breakdown structure and knowledge factors but lacks clarity on selection criteria for iteration. Model 2 consists of initial project backlog, prioritized project backlog, sprint backlog, and implemented requirements, emphasizing client-driven prioritization. Model 3 is an improved version of Model 2 based on case studies, incorporating business value, negative value, and risk for more accurate prioritization. The models were evaluated based on factors like cost, importance, risk, and dependencies to propose a comprehensive prioritization approach.

Saeed in [45] conducted a case study across five organizations. They used the grounded theory to assign numerical values to qualitative data, resulting in a more efficient and effective prioritization process. The study shows that organizations often deviate from traditional steps, using unique methods to handle requirements prioritization, focusing on Business Value, Risk Factors, and Priority Criteria. The proposed mode allows for enhancement by integrating other techniques. Agile development's evolving nature means prioritization methods will keep changing. To improve this process, the study suggests more case studies to propose better models.

Many researchers have noted gaps in the literature regarding AI-based techniques for requirements prioritization in Agile methodologies. The current literature identifies several key issues, including a lack of real-world empirical validation of these techniques. Scalability problems still exist, particularly when handling multiple requirements or large-scale projects. Furthermore, managing requirement dependencies is insufficient in current methodologies. Stakeholder collaboration is rarely fully integrated into prioritization models. This study addresses these gaps by evaluating the effectiveness of AI techniques in real-world Agile settings, focusing on scalability, dependency management, and stakeholder collaboration, while providing recommendations for future research and practical implementation.

V. Evaluation Criteria

Prioritization of requirements is the systematic procedure of arranging requirements according to specific inputs, processing techniques, and intended outputs. For this goal, the following primary categories of AI-based methodologies—Fuzzy Logic, Optimization algorithms, and Machine Learning—have been utilized. It is essential to have evaluation criteria that assess each type separately. This is necessary because these approaches differ significantly in their characteristics and operational methods. Therefore, in addition to general criteria for evaluating prioritization techniques, unique standards have been developed to evaluate Machine Learning approaches, Optimization algorithms, and Fuzzy Logic are developed. All the surveyed strategies are evaluated using ten metrics that are derived from Fuzzy Logic, Optimization, and Machine Learning, including Stakeholder Involvement, Dependency, Scalability, Validation, Impact on Collaboration, Accuracy, Flexibility, Complexity, Robustness to Uncertainty, Automation, Ease of Implementation. These criteria are evaluated using different values: 'Yes', 'No', and 'Moderate'.

VI. Results Analysis and Comparison

A. Demographics of the SLR

Table I presents a comprehensive summary of 32 research studies categorised by year, with a detailed analysis of the publication types, specifically journals. The analysis of 32 research studies from 2011 to 2024 reveals that 21 were published in journals and 11 in conferences. The publishers include Springer with 9 papers, IEEE with 10, and Elsevier with 6. The studies cover a range of topics: 3 on Fuzzy Logic, 6 on

Machine Learning, 5 on NLP, 10 on Optimization techniques, and 8 on other approaches like frameworks and gamification. Springer leads in publishing both journals and conferences, while IEEE has a strong presence in both areas, especially in Machine Learning and Natural Language Processing. Elsevier's contributions are consistent across various topics, all in journals.

In Fig. 2, the frequency of different approaches that used in requirements prioritization in Agile methodology. In Fig. 3, The year chart provides an insightful look into the trend of publications focused on using AI to prioritize requirements in Agile methodology from 2011 to 2024 that's illustrated in Fig. 3.

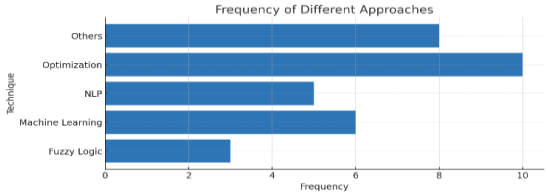

Fig. 2. Frequency of different approaches.

TABLE I. OVERVIEW OF SELECTED STUDIES (PUBLICATION TYPE JOURNAL)

No	Reference	Year	Technique Used	Related Category	Type	Publisher
15	Borhan, et al. [15]	2024	Fuzzy Logic Operations	Fuzzy Logic	Journal	JOAASR
16	Abusaeed,et al.[16]	2023	Fuzzy AHP (Analytic Hierarchy Process)	Fuzzy Logic	Journal	Elsevier
17	Rottoli, et al.[17]	2021	Fuzzy Linguistic Labels	Fuzzy Logic	Conference	CEUR-WS
18	Anand, et al. [18]	2017	Apriori Technique	Machine Learning	Journal	Elsevier
19	Hudaib, et al. [19]	2019	Self-Organizing Maps	Machine Learning	Journal	IEEE
20	Veerappa, et al. [20]	2011	Clustering	Machine Learning	Conference	Springer
21	Belsis, et al. [21]	2014	Unsupervised Requirements Clustering	Machine Learning	Journal	Springer
22	Achimugu, et al. [22]	2014	Clustering	Machine Learning	Conference	Springer
23	Kumar, et al. [23]	2022	K-Means Algorithm	Machine Learning	Conference	IEEE
24	Sachdeva, et al. [24]	2018	User Requirements Prioritization	Natural Language Processing (NLP)	Conference	IEEE
25	Shafiq,et al. [25]	2021	NLP-based Recommendation Approach	Natural Language Processing (NLP)	Journal	IEEE
26	Sami, et al. [26]	2024	Large Language Models	Natural Language Processing (NLP)	Conference	Springer
27	Sharma, et al. [27]	2019	NLP Algorithm	Natural Language Processing (NLP)	Journal	IEEE
28	Kifetew, et al. [28]	2021	User-Feedback Driven Prioritization	Natural Language Processing (NLP)	Journals	Elsevier
29	Asghar, et al. [29]	2016	Requirements Elicitation & Prioritization MoSCoW, Interviews, Workshops	Optimization	Journal	IJACSA
30	Kumar, et al. [30]	2020	AHP and TOPSIS	Optimization	Journal	Emerald
31	Muhammad, et al.[31]	2023	Multi-Criteria Decision Making Analysis	Optimization	Journal	IEEE
32	Agrawal, et al. [32]	2016	Risk Prioritization and Optimization	Optimization	Journal	IEEE
33	Prakash, et al. [33]	2021	Grey Wolf Optimizer (GWO)	Optimization	Journal	Springer
34	Chaves, et al. [34]	2015	Multiobjective Swarm Intelligence Evolutionary Algorithm	Optimization	Journal	Elsevier
35	Brezočnik, et al. [35]	2018	Particle Swarm Optimization	Optimization	Conference	Springer
36	Brezočnik, et al. [36]	2020	Swarm Intelligence Algorithms	Optimization	Conference	Springer
37	Del Sagrado, et al. [37]	2015	Ant Colony Optimization	Optimization	Conference	Springer
38	Somohano [38]	2021	Evolutionary Computing for AHP	Optimization	Conference	Springer
39	AbdElazim, et al. [9]	2020	Framework-Based Approach	Other Approaches	Journal	IOP
40	Govil, et al. [39]	2021	Information Extraction Techniques	Other Approaches	Journal	IEEE
41	Kamal, et al. [40]	2020	Prioritization Techniques	Other Approaches	Journal	IEEE
42	Kifetew, et al.[41]	2017	Gamification, Collaborative Techniques	Other Approaches	Conference	IEEE
43	Perkusich, et al.[42]	2020	Intelligent Software Engineering	Other Approaches	journal	Elsevier
44	AL-Ta'ani, et al. [43]	2013	Conceptual Framework	Other Approaches	Journal	Elsevier
45	Alkandari, et al. [44]	2017	Enhancement Techniques	Other Approaches	Journal	JSW
46	Saeed, et al. [45]	2023	Requirements Prioritization Techniques	Other Approaches	Journal	Technical Journal of UET

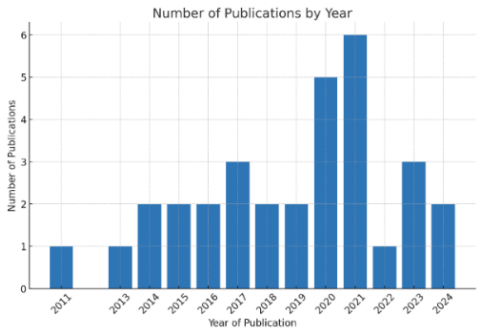

Fig. 3. Number of publications per year.

B. Strengths and Weaknesses of the Surveyed Techniques

The evaluation of the 32 papers reveals a diverse range of strengths, weaknesses, and limitations in applying various methodologies to Agile Software Development is presented in Table II. Many papers showcase innovative approaches, such as the combination of Fuzzy Logic with AHP for enhanced decision-making, the use of NLP for automating prioritization, and the application of advanced algorithms for task allocation and Optimization. These strengths are often balanced by significant weaknesses, including complexity in implementation, the need for expert knowledge, and high computational or data requirements. Moreover, limitations such as scalability, data dependency, and subjective judgment frequently emerge, indicating that while these methods provide valuable solutions, they may not be universally applicable without careful consideration of context and resources. Overall, the analysis underscores the need for a tailored approach when applying these techniques to Agile environments, ensuring that their strengths are fully leveraged while mitigating their inherent weaknesses and limitation.

TABLE II. STRENGTH, WEAKNESS, AND LIMITATIONS OF REQUIREMENTS PRIORITIZATIONS TECHNIQUES

Reference	Strengths	Weaknesses	Limitations
Borhan, et al. [15]	Improved decision making under uncertainty	Complexity in fuzzy rule definitions	Limited scalability
Abusaeed,et al.[16]	Combines advantages of AHP and fuzzy logic	Requires expert knowledge	Complexity in AHP hierarchy formulation
Rottoli, et al.[17]	Handles linguistic uncertainty effectively	Potential bias	Subjective interpretation of linguistic labels
Anand, et al. [18]	Resolves conflicts through data mining	Complexity in implementation	Data dependency
Hudaib, et al. [19]	Visualizes data patterns effectively	Requires expertise in SOM	Complexity
Veerappa, et al. [20]	Groups similar stakeholders effectively	May not capture all stakeholder nuances	Data dependency
Belsis, et al. [21]	Unsupervised approach reduces bias	Requires large dataset	Complexity
Achimugu, et al. [22]	Efficiently handles large scale data	May overlook individual nuances	Scalability
Kumar, et al. [23]	Simple and fast clustering method	Sensitive to initial conditions	Scalability
Sachdeva, et al. [24]	Focuses on user requirements	May overlook technical requirements	Subjective judgement
Shafiq,et al. [25]	Automates prioritization using NLP	Requires large dataset	Data dependency
Sami, et al. [26]	Leverages advanced language models	Computationally intensive	Data dependency
Sharma, et al. [27]	Automates release planning using NLP	Requires expertise in NLP	Data dependency
Kifetew, et al. [28]	Incorporates user feedback in prioritization	Potential bias in feedback	Data dependency
Asghar, et al. [29]	Enhances quality by thorough elicitation	Time-consuming	High dependency on stakeholder input
Kumar, et al. [30]	Effective prioritization by combining AHP and TOPSIS	Data-intensive	Complexity in combining methods
Muhammad, et al.[31]	Comprehensive evaluation of multiple criteria	Resource-intensive	High complexity
Agrawal, et al. [32]	Focuses on risk management and optimization	Complexity in risk assessment	High complexity
Prakash, et al. [33]	Efficient optimization algorithm	Requires parameter tuning	Complexity
Chaves, et al. [34]	Handles multiple objectives simultaneously	Computationally intensive	High complexity
Brezočnik, et al. [35]	Efficient task allocation algorithm	Requires parameter tuning	Complexity
Brezočnik, et al. [36]	Effective in solving complex problems	Requires expertise in swarm intelligence	Complexity
Del Sagrado, et al. [37]	Effective multi-objective optimization	Computationally intensive	High complexity
Somohano [38]	Enhances AHP with evolutionary computing	Complexity in implementation	Requires expert knowledge
AbdElazim, et al. [9]	Provides structured approach for prioritization	May not fit specific project needs	Subjective judgement
Govil, et al. [39]	Automates data extraction process	Requires comprehensive datasets	Data extraction accuracy
Kamal, et al. [40]	Focuses on global software development challenges	Difficult to generalize	High variability in global context
Kifetew, et al.[41]	Engages stakeholders through gamification	Potential for bias in game dynamics	Engagement dependency
Perkusich, et al.[42]	Leverages AI for software engineering	Requires extensive training data	Data dependency
AL-Ta'ani, et al. [43]	Provides a structured approach	May not fit specific project needs	Subjective judgement
Alkandari, et al. [44]	Improves existing processes	Requires adaptation to specific projects	Subjective judgement
Saeed, et al. [45]	Enhances quality of agile practices	Requires adaptation to specific contexts	Subjective judgement

C. Comparison of AI-based Requirements Prioritization Techniques

A comprehensive study is presented among Fuzzy Logic based, Machine Learning based, Optimization and Natural Language Processing-based techniques. The effectiveness of AI-driven prioritization in Agile methodologies is influenced by key parameters such as scalability, dependency management, stakeholder collaboration, automation, and accuracy. AI techniques like machine learning and optimization improve scalability by handling large datasets and managing dependencies effectively. NLP enhances stakeholder collaboration by automating feedback interpretation, while automation in AI reduces manual effort and allows dynamic prioritization. Fuzzy logic supports decision-making under uncertainty but may struggle with scalability. Overall, these parameters ensure that AI techniques provide adaptable, efficient, and accurate solutions to the challenges of requirements prioritization in Agile development. This comprehensive is defined based on evaluation criteria with answer of 'Y' for Yes, 'N' for No, and 'M' for 'Moderate'.

Regarding the Fuzzy Logic-based technique, evaluating the implementation of Fuzzy Logic in Agile Software Development requires examining both its theoretical advantages and practical challenges. Fuzzy Logic offers a flexible and nuanced approach to handling uncertainty and imprecision in decision-making, particularly in complex environments like Agile projects where requirements are often ambiguous and evolving. The evaluation, as shown in Table III, reveals that while these Fuzzy Logic-based approaches are strong in handling uncertainty and, in some cases, fostering stakeholder collaboration, they generally struggle with scalability and managing dependencies. The high complexity and limited usability in some methods could restrict their practical implementation in real-world Agile projects. This highlights a need for more balanced solutions that are both theoretically robust and practically applicable.

TABLE III. FUZZY LOGIC TECHNIQUE APPLICATIONS

Reference	Stakeholder Collaboration	Dependency	Scalability	Complexity and Usability	Flexibility	Robustness to Uncertainty
Borhan, et al. [15]	Y	N	N	N	Y	Y
Abusaeed,et al.[16]	N	N	N	N	N	Y
Rottoli, et al.[17]	Y	N	N	Y	Y	Y

Regarding Machine Learning technique, the evaluated research offers innovative clustering and prioritization techniques for Agile Software Development most to techniques aims to categorize related requirements into clusters. The cluster are pushed to be the next sprints based on their ranking. Some evaluation criteria as scalability, accuracy, and stakeholder management shows good indicator as it's shown in Table IV. However, they generally face challenges in practical implementation due to complexity, lack of empirical validation, that would show limited effectiveness in handling stakeholder conflicts. While these approaches are valuable, their applicability may be constrained by the need for specialized knowledge and tools.

TABLE IV. MACHINE LEARNING TECHNIQUE APPLICATIONS

Reference	Stakeholder Collaboration	Dependency	Scalability	Validation	Impact on Collaboration	Accuracy	Flexibility
Anand, et al. [18]	Y	N	N	N	Y	Y	N
Hudaib, et al. [19]	Y	Y	Y	N	N	Y	Y
Veerappa, et al. [20]	Y	Y	N	N	Y	Y	Y
Belsis, et al. [21]	Y	N	Y	N	Y	Y	Y
Achimugu, et al. [22]	Y	N	Y	N	Y	Y	Y
Kumar, et al. [23]	N	N	Y	Y	Y	Y	Y

Regarding Natural Language Processing technique, most of the paper used NLP techniques reduce direct stakeholder involvement, which is crucial for aligning requirements with business needs. In Table IV, we can realize many of the papers moderate to high scalability, ease of implementation, suggesting a potential gap in practical applicability. The validation factor generally moderate to high. Most of the papers didn't consider dependency directly (Table V).

TABLE V. NATURAL LANGUAGE PROCESSING TECHNIQUE APPLICATION

Reference	Stakeholder Collaboration	Dependency	Scalability	Automation	Validation
Sachdeva, et al. [24]	M	M	Y	Y	M
Shafiq,et al. [25]	Y	Y	M	M	Y
Sami, et al. [26]	M	M	Y	Y	Y
Sharma, et al. [27]	Y	M	Y	Y	Y
Kifetew, et al. [28]	M	M	Y	Y	M

Regarding Optimization technique, the evaluation of these papers indicates a week-long focus on addressing requirements dependencies and facilitating scalability across most approaches. As shown in Table VI, traditional decision-making techniques, such as AHP and TOPSIS, are well-integrated with collaboration with stakeholders. And less collaboration with swarm intelligence or evolutionary algorithms. Continuous improvement is a common feature across all methods, reflecting the iterative nature of Agile methodologies. Overall, while most approaches are not considered dependencies directly and scalability, there is variability in how well they facilitate stakeholder collaboration, which underscores the potential of improvement in algorithm-based methods.

Regarding other frameworks and techniques, as it's shown in Table VII, the collection of papers on requirements prioritization in Agile Software Development offers a range of theoretical frameworks and innovative tools, most approaches address well result in collaboration, dependencies, and continuous improvement effectively. But there is challenges in such terms as ease of implementation and scalability for certain innovative methods like DMGame. This suggests that while these approaches offer significant potential, they may require additional refinement or adaptation to be fully effective across different Agile contexts

TABLE VI. OPTIMIZATION TECHNIQUE APPLICATION

Reference	Stakeholder Collaboration	Dependency	Scalability	Ease of Implementation	Validation
Asghar, et al. [29]	N	N	N	Y	Y
Kumar, et al. [30]	Y	N	N	Y	Y
Muhammad, et al.[31]	Y	Y	Y	Y	Y
Agrawal, et al. [32]	Y	N	N	Y	Y
Prakash, et al. [33]	N	N	N	Y	Y
Chaves, et al. [34]	N	N	N	N	Y
Brezočnik, et al. [35]	N	Y	N	Y	Y
Brezočnik, et al. [36]	N	N	N	Y	Y
Del Sagrado, et al. [37]	N	N	N	Y	Y
Somohano, et al, [38]	Y	Y	Y	Y	Y

TABLE VII. OTHER TECHNIQUES APPLICATION

Reference	Stakeholder Collaboration	Dependency	Scalability	Ease of Implementation	Validation	Flexibility in Handling Change
AbdElazim, et al. [9]	Y	Y	Y	Y	Y	Y
Govil, et al. [39]	Y	Y	Y	Y	Y	Y
Kamal, et al. [40]	Y	Y	Y	Y	Y	Y
Kifetew, et al.[41]	Y	N	N	N	Y	Y
Perkusich, et al.[42]	N	Y	Y	N	Y	Y
AL-Ta'ani, et al. [43]	Y	Y	Y	Y	Y	Y
Alkandari, et al. [44]	Y	Y	Y	Y	Y	Y
Saeed, et al. [45]	Y	Y	Y	Y	Y	Y

Results show that Machine Learning techniques, such as clustering algorithms, effectively address the scalability challenges in requirement prioritization for large Agile projects by grouping similar requirements, reducing manual effort, and improving accuracy. Fuzzy Logic also enhances stakeholder collaboration, resolving conflicts and improving decision-making. These findings suggest that AI techniques not only offer technical advantages but also enhance team communication and responsiveness in Agile workflows, though real-world validation is still needed.

VII. LIMITATIONS

Despite the promising results of AI-based techniques for requirements prioritization, this study has several limitations. A key limitation is the lack of empirical validation, which affects the practical applicability of many AI methods in Agile environments. While these techniques show potential in addressing scalability and improving stakeholder collaboration, their complexity and the need for specialized knowledge pose challenges for widespread adoption and for handling changes in requirements effectively. Additionally, dependency management is not adequately addressed, leaving a crucial aspect of Agile prioritization insufficiently explored.

VIII. CONCLUSION

This paper has conducted a comprehensive systematic literature review of AI-driven techniques for requirements prioritization within Agile methodologies. This paper addresses a significant gap by analysing 32 key studies spanning 2010 to 2024. The SLR finds the strengths and weaknesses of Fuzzy Logic, Machine Learning, Optimization, and Natural Language Processing (NLP) techniques. Our findings reveal that each method has its distinct limitations. These limitations are particularly in terms of scalability, accuracy, and simplicity of implementation. These AI-based approaches offer promising solutions to the challenges of requirements prioritization in Agile environments.

In Future work, the analysis underscores the necessity of developing hybrid AI-based techniques that integrate the strengths of Fuzzy Logic, Machine Learning, Natural Language Processing, and Optimization to create more scalable, and efficient prioritization methods. The integrated approach could better handle the complexities of modern software development in Agile methodology. That approach should handle the rapidly changing of the Agile environments where continuous prioritization and stakeholder collaboration are critical. The approach should focus on empirical validation of AI-based requirements prioritization techniques through case studies and real-world applications to ensure their practical relevance. Additionally, there is a need to align this approach to be more closely with stakeholder expectations and to meet the business need, and to ensure that the prioritization process in Agile technically and contextually appropriate. Such efforts should prioritize creating scalable, efficient, and adaptive methods that consider dependencies in requirements and align closely with stakeholder needs and Agile practices and to align with the continuous improvement. By addressing these challenges, it will be possible to develop more effective and adaptable requirements prioritization techniques that can be broadly implemented in diverse, real-world software development environments.

ACKNOWLEDGMENT

I would like to express my sincere gratitude to my supervisors, for their invaluable guidance. I also extend my thanks to the Software Engineering Department whose unwavering support during my qualification year played a pivotal role in the successful completion of this research.

REFERENCES

[1] T. Ambreen, N. Ikram, M. Usman, and M. Niazi, "Empirical research in requirements engineering: trends and opportunities," Requir Eng, vol. 23, pp. 63–95, 2018.

[2] W. Jiang, H. Ruan, L. Zhang, P. Lew, and J. Jiang, "For user-driven software evolution: Requirements elicitation derived from mining online reviews," in Advances in Knowledge Discovery and Data Mining: 18th Pacific-Asia Conference, PAKDD 2014, Tainan, Taiwan, May 13-16, 2014. Proceedings, Part II 18, Springer, 2014, pp. 584–595.

[3] P. A. Laplante and M. Kassab, Requirements engineering for software and systems. Auerbach Publications, 2022.

[4] A. Radwan, A. Abdo, and S. A. Gaber, "An Approach for Requirements Engineering Analysis using Conceptual Mapping in Healthcare Domain," International Journal of Advanced Computer Science and Applications, vol. 12, no. 8, 2021.

[5] A. Gupta, G. Poels, and P. Bera, "Using conceptual models in agile software development: a possible solution to requirements engineering challenges in agile projects," IEEE Access, vol. 10, pp. 119745–119766, 2022.

[6] N. H. Borhan, H. Zulzalil, and N. M. A. Sa'adah Hassan, "Requirements prioritization techniques focusing on agile software development: a systematic literature," International Journal of Scientific and Technology Research, vol. 8, no. 11, pp. 2118–2125, 2019.

[7] W. R. Fitriani, P. Rahayu, and D. I. Sensuse, "Challenges in agile software development: A systematic literature review," in 2016 International Conference on Advanced Computer Science and Information Systems (ICACSIS), IEEE, 2016, pp. 155–164.

[8] Alawneh, L, "Requirements prioritization using hierarchical dependencies." Information technology-new generations: 14th International Conference on Information Technology. Springer International Publishing, 2018.

[9] K. AbdElazim, R. Moawad, and E. Elfakharany, "A framework for requirements prioritization process in agile software development," in Journal of Physics: Conference Series, IOP Publishing, 2020, p. 012001.

[10] J. M. Mendel, "Fuzzy logic systems for engineering: a tutorial," Proceedings of the IEEE, vol. 83, no. 3, pp. 345–377, 1995.

[11] R. Anwar and M. B. Bashir, "A Systematic Literature Review of AI-based Software Requirements Prioritization Technique," IEEE Access, 2023.

[12] M. D. Nagpal, K. Malik, and A. Kalia, "A comprehensive analysis of requirement engineering utilizing machine learning techniques," Design Engineering, pp. 2662–2678, 2021.

[13] W. Khan, A. Daud, J. A. Nasir, and T. Amjad, "A survey on the state-of-the-art machine learning models in the context of NLP," Kuwait journal of Science, vol. 43, no. 4, 2016.

[14] K.-L. Du and M. N. S. Swamy, Search and optimization by metaheuristics, vol. 1. Springer, 2016.

[15] N. H. Borhan, H. Zulzalil, N. M. Ali, A. B. M. Sultan, and R. Bahsoon, "Stakeholder Analysis using Fuzzy Logic Operations for Integrated User Story Prioritisation Approach in Agile-Scrum Method," Journal of Advanced Research in Applied Sciences and Engineering Technology, vol. 47, no. 2, pp. 76–93, 2024.

[16] S. Abusaeed, S. U. R. Khan, and A. Mashkoor, "A Fuzzy AHP-based approach for prioritization of cost overhead factors in agile software development," Appl Soft Comput, vol. 133, p. 109977, 2023.

[17] G. D. Rottoli and C. Casanova, "Multi-criteria group requirement prioritization in software engineering using fuzzy linguistic labels.," in ICAI Workshops, 2021, pp. 16–28.

[18] R. V. Anand and M. Dinakaran, "Handling stakeholder conflict by agile requirement prioritization using Apriori technique," Computers & Electrical Engineering, vol. 61, pp. 126–136, 2017.

[19] A. Hudaib and F. Alhaj, "Self-Organizing Maps for Agile Requirements Prioritization," in 2019 2nd International Conference on new Trends in Computing Sciences (ICTCS), IEEE, 2019, pp. 1–5.

[20] V. Veerappa and E. Letier, "Clustering stakeholders for requirements decision making," in Requirements Engineering: Foundation for Software Quality: 17th International Working Conference, REFSQ 2011, Essen, Germany, March 28-30, 2011. Proceedings 17, Springer, 2011, pp. 202–208.

[21] P. Belsis, A. Koutoumanos, and C. Sgouropoulou, "PBURC: a patterns-based, unsupervised requirements clustering framework for distributed agile software development," Requir Eng, vol. 19, pp. 213–225, 2014.

[22] P. Achimugu, A. Selamat, and R. Ibrahim, "A Clustering Based Technique for Large Scale Prioritization during Requirements Elicitation. 2014," Cham: Springer International Publishing.

[23] B. Kumar, U. K. Tiwari, D. C. Dobhal, and H. S. Negi, "User Story Clustering using K-Means Algorithm in Agile Requirement Engineering," in 2022 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES), IEEE, 2022, pp. 1–5.

[24] S. Sachdeva, A. Arya, P. Paygude, S. Chaudhary, and S. Idate, "Prioritizing user requirements for agile software development," in 2018 International Conference On Advances in Communication and Computing Technology (ICACCT), IEEE, 2018, pp. 495–498.

[25] S. Shafiq, A. Mashkoor, C. Mayr-Dorn, and A. Egyed, "NLP4IP: Natural language processing-based recommendation approach for issues prioritization," in 2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), IEEE, 2021, pp. 99–108.

[26] M. A. Sami, Z. Rasheed, M. Waseem, Z. Zhang, T. Herda, and P. Abrahamsson, "Prioritizing Software Requirements Using Large Language Models," arXiv preprint arXiv:2405.01564, 2024.

[27] S. Sharma and D. Kumar, "Agile release planning using natural language processing algorithm," in 2019 Amity International Conference on Artificial Intelligence (AICAI), IEEE, 2019, pp. 934–938.

[28] F. M. Kifetew, A. Perini, A. Susi, A. Siena, D. Muñante, and I. Morales-Ramirez, "Automating user-feedback driven requirements prioritization," Inf Softw Technol, vol. 138, p. 106635, 2021.

[29] A. R. Asghar, S. N. Bhatti, A. Tabassum, Z. Sultan, and R. Abbas, "Role of requirements elicitation & prioritization to optimize quality in scrum agile development," International Journal of Advanced Computer Science and Applications, vol. 7, no. 12, 2016.

[30] R. Kumar, K. Singh, and S. K. Jain, "A combined AHP and TOPSIS approach for prioritizing the attributes for successful implementation of agile manufacturing," International Journal of Productivity and Performance Management, vol. 69, no. 7, pp. 1395–1417, 2020.

[31] A. Muhammad, A. Siddique, M. Mubasher, A. Aldweesh, and Q. N. Naveed, "Prioritizing non-functional requirements in agile process using multi criteria decision making analysis," IEEE Access, vol. 11, pp. 24631–24654, 2023.

[32] R. Agrawal, D. Singh, and A. Sharma, "Prioritizing and optimizing risk factors in agile software development," in 2016 ninth international conference on contemporary computing (IC3), IEEE, 2016, pp. 1–7.

[33] B. Prakash and V. Viswanathan, "ARP–GWO: an efficient approach for prioritization of risks in agile software development," Soft comput, vol. 25, no. 7, pp. 5587–5605, 2021.

[34] J. M. Chaves-Gonzalez, M. A. Perez-Toledano, and A. Navasa, "Software requirement optimization using a multiobjective swarm intelligence evolutionary algorithm," Knowl Based Syst, vol. 83, pp. 105–115, 2015.

[35] L. Brezočnik, I. Fister Jr, and V. Podgorelec, "Scrum task allocation based on particle swarm optimization," in International Conference on Bioinspired Methods and Their Applications, Springer, 2018, pp. 38–49.

[36] L. Brezočnik, I. Fister, and V. Podgorelec, "Solving agile software development problems with swarm intelligence algorithms," in New Technologies, Development and Application II 5, Springer, 2020, pp. 298–309.

[37] J. Del Sagrado, I. M. Del Águila, and F. J. Orellana, "Multi-objective ant colony optimization for requirements selection," Empir Softw Eng, vol. 20, pp. 577–610, 2015.

[38] J. C. B. Somohano-Murrieta, J. O. Ocharán-Hernández, Á. J. Sánchez-García, X. Limón, and M. de los Ángeles Arenas-Valdés, "Improving the analytic hierarchy process for requirements prioritization using evolutionary computing," Programming and Computer Software, vol. 47, pp. 746–756, 2021.

[39] N. Govil and A. Sharma, "Information extraction on requirement prioritization approaches in agile software development processes," in 2021 5th International Conference on Computing Methodologies and Communication (ICCMC), IEEE, 2021, pp. 1097–1100.

[40] T. Kamal, Q. Zhang, M. A. Akbar, M. Shafiq, A. Gumaei, and A. Alsanad, "Identification and prioritization of agile requirements change management success factors in the domain of global software development," IEEE Access, vol. 8, pp. 44714–44726, 2020.

[41] F. Kifetew, D. Munante, A. Perini, A. Susi, A. Siena, and P. Busetta, "DMGame: a gamified collaborative requirements prioritisation tool," in 2017 IEEE 25th International Requirements Engineering Conference (RE), IEEE, 2017, pp. 468–469.

[42] M. Perkusich et al., "Intelligent software engineering in the context of agile software development: A systematic literature review," Inf Softw Technol, vol. 119, p. 106241, 2020.

[43] R. H. AL-Ta'ani and R. Razali, "Prioritizing requirements in agile development: A conceptual framework," Procedia Technology, vol. 11, pp. 733–739, 2013.

[44] M. A. Alkandari and A. Al-Shammeri, "Enhancing the Process of Requirements Prioritization in Agile Software Development-A Proposed Model.," J. Softw., vol. 12, no. 6, pp. 439–453, 2017.

[45] N. Saeed, E. Yasmin, A. R. Riaz, N. Khalid, Y. Hafeez, and I. Rubab, "Enabling the Requirements Prioritization Techniques to Improve the Quality of Agile Practices.," Technical Journal of University of Engineering & Technology Taxila, vol. 28, no. 4, 2023.