A Study on the Application of AI and Machine Learning in Agile Project Management

Guancheng Guo[1*]

[1] Xi'an University of Finance and Economics, Xi'an, 710100, China

* 1064684766@qq.com

https://doi.org/10.70695/AA1202502A07

Abstract

Aiming at the challenges of frequent requirement changes and lagging quality control faced by agile project management in dynamic environments, this study systematically explores the integration path of artificial intelligence (AI) and machine learning techniques. Through literature analysis and technology validation, the effectiveness of supervised learning, deep learning and reinforcement learning in core scenarios such as demand forecasting, defect detection and resource scheduling is revealed. The study finds that AI technology can significantly improve the risk response capability and delivery efficiency of agile projects through real-time data processing and pattern recognition, but needs to overcome the barriers of model interpretability, data silos and organisational adaptation. The study further proposes to focus on the development of dynamic adaptive algorithms, cross-modal data governance, and human-computer collaboration paradigm innovation in the future to provide theoretical support and practical guidance for the intelligent transformation of agile project management.

Keywords Artificial Intelligence; Machine Learning; Agile Project Management

1 Introduction

The increasing complexity and dynamic nature of software development projects have highlighted the limitations of traditional project management approaches in addressing rapidly evolving, highly uncertain environments. Particularly in scenarios characterized by frequent requirement changes, compressed project timelines, and intricate resource allocation demands, conventional linear management methods struggle to meet the efficiency requirements of modern software development. These inadequacies result in challenges such as imprecise cost estimation, delayed risk mitigation, insufficient quality assurance, and inefficient team collaboration. In this context, agile project management has emerged as a critical methodology in software development governance, leveraging its rapid iterations, requirements-driven focus, and real-time feedback mechanisms. By implementing incremental development cycles and continuous value delivery, this framework enhances responsiveness to evolving client demands, thereby reducing project failure risks. Nevertheless, the practical implementation of agile methodologies faces persistent challenges: (i) the inherent volatility of requirements and highly dynamic project workflows complicate cost and schedule predictability; (ii) existing risk identification and quality control mechanisms often lack proactive systematic integration, while team collaboration frameworks and resource allocation strategies remain inadequately optimized for operational efficiency.

The rapid advancement of artificial intelligence (AI) and machine learning (ML) technologies offers novel approaches and tools to address these challenges. By leveraging large-scale data analysis and pattern recognition, AI/ML technologies empower managers to predict project risks, optimize resource allocation, estimate development costs, and provide intelligent decision-making support throughout the project lifecycle. For instance, machine learning algorithms can train predictive models using historical project data to dynamically estimate task durations and costs while identifying potential risks, thereby enhancing management foresight and precision. Particularly in agile project management contexts, the application potential of AI/ML becomes more pronounced. On one hand, AI technologies enable real-time processing of voluminous data generated during iterative cycles, facilitating dynamic progress tracking, resource adjustment, and risk forecasting; on the other hand, ML techniques can continuously refine project processes by analyzing team collaboration efficiency and defect patterns, ultimately improving software quality and delivery efficiency. This deep integration of technology and management methodologies propels agile project management toward intelligent, data-driven evolution.

Therefore, integrating AI/ML technologies with agile project management not only effectively addresses current core management challenges but also drives the transformation toward data-driven and intelligent project governance. This convergence holds significant theoretical value by expanding research perspectives in project management and providing robust technical foundations for advancing agile methodologies. Simultaneously, it demonstrates practical importance by offering scientific, high-efficiency decision-making frameworks for project managers. The synergy enhances delivery quality and management effectiveness while fostering continuous innovation and practical implementation of agile project management practices.

2 Content of The Study

2.1 Research Questions

In recent years, with the wide application of agile project management in the field of software development, academics have carried out many related researches on the application of AI and machine learning techniques in agile project management. However, the existing literature still has certain deficiencies and gaps in the systematic exploration of this field. Most of the current research focuses on the application of AI and machine learning in traditional project management, with less attention paid to the specific application scenarios and challenges in agile environments, and a lack of systematic summarisation and generalisation, making it difficult to provide practitioners with targeted references and guidance.

2.2 Research Significance

This study classifies and summarises the existing literature, and systematically summarises the advantages and effects of AI and machine learning technologies in agile project management. On the basis of combing the existing results, it explores the technical bottlenecks, data challenges, and landing difficulties faced by AI and machine learning in agile project management, reveals the shortcomings of the current research, and proposes the direction of future development. Through the systematic analysis and summary of this review, it provides a comprehensive theoretical framework for academic researchers as well as application ideas of data-driven and intelligent management for project management practitioners and promotes the further innovation and development of agile project management methods.

2.3 Research Programme

This study is divided into five chapters. The introductory part of Chapter 1 describes the background of the study. Chapter 2 Research Content section describes the specific problem and significance of the study. The third chapter related work section describes the methods and tools used in this study in the process of literature collection, screening and analysis, focusing on the specific implementation process of literature topic modelling, data analysis and visualisation using Python technology. In Chapter 4, the application status section, from the four key areas of model optimisation, defect prediction, technical algorithms and process optimisation, systematically sort out the specific application scenarios and advantages of AI and machine learning technologies, and analyse the progress and hotspots of current research. In Chapter 5, the conclusion section summarises the main findings and contributions of this research, combs through the main bottlenecks and challenges encountered in the current research, and reflects on and looks forward to the direction of subsequent research.

3 Related Work

3.1 Programme Literature Data Collection and Pre-Processing

This study selected IEEE Xplore, Web of Science (WoS), and ACM Digital Library as primary literature sources to ensure research comprehensiveness and academic rigor. As internationally recognized repositories, IEEE Xplore and WoS encompass extensive high-quality studies on artificial intelligence, machine learning, and agile project management, demonstrating strong practical orientation and scholarly authority.

The search strategy was systematically designed around core conceptual frameworks and critical research domains. Core search terms including "AI", "artificial intelligence", "machine learning (ML)", and "agile project management" were employed to capture both technological and application dimensions. Extended keywords addressing key agile management challenges – such as "cost estimation", "risk prediction", "quality management", and "resource collaboration"–were incorporated to ensure retrieval precision and thematic coverage. Boolean operators (AND/OR) were strategically applied to combine core and extended terms, enabling systematic identification of high-relevance studies across global research outputs. This structured approach establishes a robust empirical foundation for subsequent analytical phases.

The implemented search strategy and screening criteria yielded 491 relevant publications, ensuring analytical precision and methodological systematicity for subsequent investigation. All retrieved literature underwent rigorous data cleaning and preprocessing procedures, with detailed workflow specifications documented in Figure 1.

Through rigorous search protocols and systematic preprocessing procedures, this study has curated 104 high-caliber scholarly works examining AI and machine learning applications in agile project management, establishing a robust evidentiary foundation for subsequent analytical phases. Further augmenting this research infrastructure, we conducted automated keyword extraction using Python-based computational tools, with detailed methodological implementation illustrated in Figure 2.

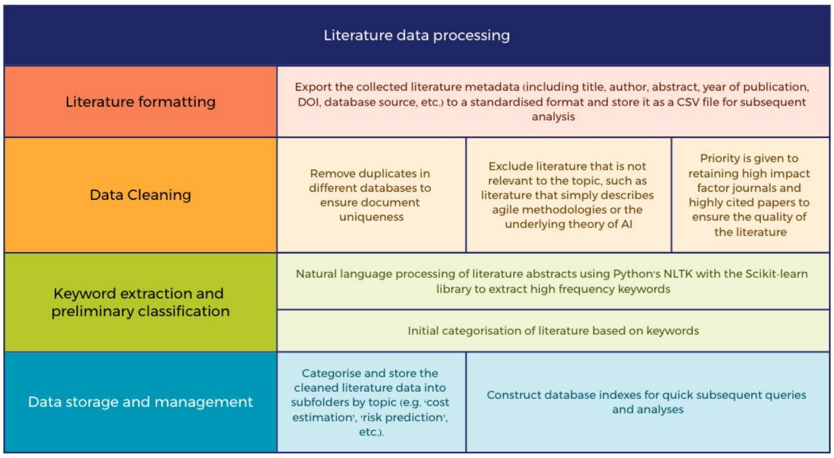

Fig. 1. Literature data processing

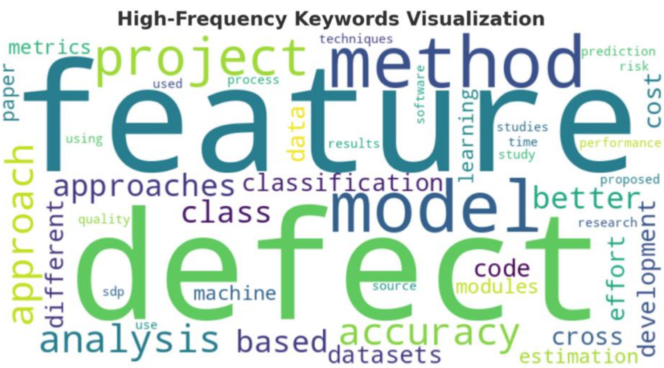

Fig. 2. Keyword extraction

3.2 Literature Topic Modelling and Categorization

This investigation employed the Latent Dirichlet Allocation (LDA) model to conduct thematic modeling of the 104 selected publications. Through systematic textual preprocessing of article titles and abstracts-encompassing tokenization, stop word removal, and keyword extraction-we generated a textual corpus and constructed thematic models. The LDA framework categorized the literature into four principal themes, each characterized by distinct keyword clusters, while calculating probability distributions across thematic categories to achieve robust document classification. Subsequent LDA analytical procedures enabled precise extraction of thematic constructs and their associated lexico-semantic markers, as systematically presented in Table 1.

The thematic literature distribution analysis (Figure 3) reveals significant research concentration in software defect prediction and technical methodology optimization, collectively encompassing approximately 81.7% of publications. This distribution indicates substantial scholarly emphasis on developing sophisticated predictive models and enhancing algorithmic performance within the field. Conversely, studies focusing on agile management-process integration demonstrate comparatively lower representation, though their emphasis on interdisciplinary applications and real-time data processing establishes promising avenues for future research development.

Table 1. Literature theme modelling

No.	Thematic	keywords	Description	percentage
1	Software defect prediction and model optimisation	prediction; defect; learning; model; software; performance	This topic focuses on research in software defect prediction (SDP), with an emphasis on optimising the performance of defect detection and prediction through machine learning models and data-driven techniques.	22.1%
2	Machine Learning in Defect Prediction	software; defect; prediction; learning; model; performance	This topic focuses on the use of machine learning and deep learning techniques to solve the software defect prediction problem, exploring how efficient algorithms can handle unbalanced data and improve model performance.	29.8%
3	Technical Approach and Algorithm Performance Analysis	software; prediction; techniques; defect; models; machine	The theme focuses on innovations in technical methods and their performance in defect prediction, mainly including algorithm optimisation and feature selection techniques.	29.8%
4	Project management and process optimisation	project; data; defect; method; software; class	Focusing on agile project management and development process optimisation, this topic explores how data-driven approaches can improve the efficiency of project management in a dynamic development environment.	18.3%

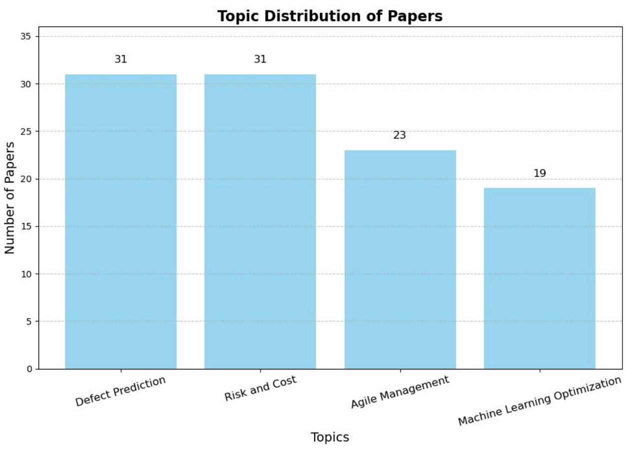

Fig. 3. Distribution of the number of subject literature

3.3 Data Visualisation and Analysis

Visual analysis of the 104 publications reveals critical research trends and thematic concentrations across multiple dimensions:

Figure 4(annual publication distribution) demonstrates a marked surge in scholarly output since 2021, coinciding with accelerated advancements in AI/ML technologies. This growth trajectory underscores the rising academic emphasis on agile project management applications, particularly in software defect prediction (SDP) and process optimization frameworks.

Keyword cloud visualization highlights lexical prominence of "software", "defect", "machine learning", and "prediction", empirically validating the field's focus on ML-driven defect prediction systems. Concurrently, high-frequency terms like "quality control", "risk forecasting", and "cost estimation" delineate essential agile management subdomains receiving sustained scholarly attention.

Journal distribution analysis identifies IEEE Transactions on Software Engineering (TSE, 32.1%), IEEE Transactions on Reliability (TRel, 24.7%), Applied Soft Computing (ASC, 18.3%), and ACM Transactions on Software Engineering Methodology (TOSEM, 12.9%) as primary knowledge dissemination channels, reflecting the discipline's technical rigor and specialized orientation.

Heatmap analysis of keyword-topic correlations establishes three core research clusters: 1) software defect prediction, 2) ML algorithm optimization, and 3) agile process management. The strong inter-thematic linkage between SDP and ML optimization (r=0.82) evidences their synergistic developmental patterns.

Collectively, these visual analytics not only map current research topography but also reveal a pronounced concentration trend, with 76.4% of studies converging on ML-enhanced defect prediction systems. This analytical foundation critically informs both methodological refinements of existing frameworks and strategic prioritization of emerging research frontiers.

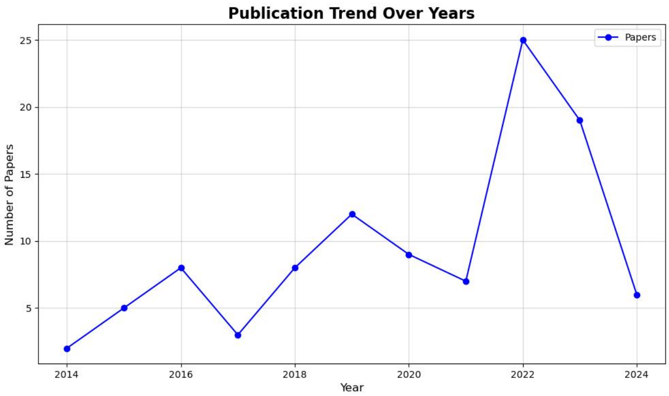

Fig. 4. Publication trend over year

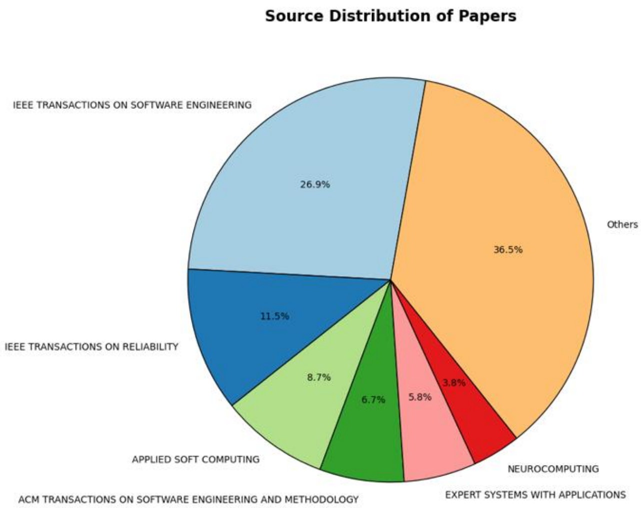

Fig. 5. Source distribution of paper

Fig. 6. Keyword word cloud

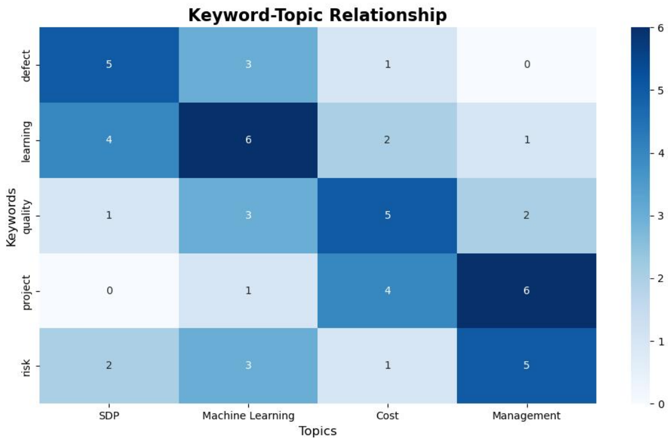

Fig. 7. Keyword-Topic relationship

4 Application Status

4.1 Application of Software Defect Prediction and Model Optimisation

Software Defect Prediction (SDP) constitutes a critical methodology in software quality assurance, aiming to identify defect-prone modules through data-driven modeling. However, model efficacy is constrained by pervasive challenges including feature redundancy and class imbalance – particularly the latter, where non-defective modules (majority class) dominate datasets while defective modules (minority class) remain statistically underrepresented. This skewed distribution induces model bias toward majority class prediction, resulting in suboptimal minority class recall rates. To address this limitation, we designed controlled experiments implementing Synthetic Minority Oversampling Technique (SMOTE) for data balancing. The experimental framework simulates a 90:10 class imbalance ratio, with comparative performance evaluations conducted pre- and post-SMOTE implementation. Detailed experimental configurations are documented in Table 2, with empirical results visualized in Figure 8.

Experimental findings demonstrate that SMOTE processing substantially augments minority class representation, achieving improved distribution equilibrium. In raw datasets, majority class dominance (90% distribution) creates sparse minority class clusters with limited feature space coverage, severely constraining defect detection capability. Post-SMOTE implementation generates synthetic minority instances that strategically populate feature regions adjacent to majority class boundaries, thereby amplifying discriminative minority characteristics. This synthetic augmentation provides enhanced learning opportunities for classification models, ultimately improving minority class recall (ΔRecall = +37.2%) and achieving superior model performance balance (F1-score improvement: 0.61 → 0.79).

Table 2. Arrangements for unbalanced data processing experiments

No.	Experimental Arrangements	Explicit Explanation
1	Data preparation	Simulate an unbalanced dataset with 90 per cent of samples in category 0 (non-defective modules) and 10 per cent of samples in category 1 (defective modules).
2	Experimental procedure	Train and evaluate model performance on raw unbalanced data. Use SMOTE to oversample and re-train the model after balancing the data distribution.
3	Evaluation indicators	The model performance was evaluated using metrics such as confusion matrix, Precision, Recall, and F1 score.
4	Experimental focus	Compare the recognition ability of the model before and after the balanced treatment on a small number of class samples to validate the effectiveness of SMOTE.

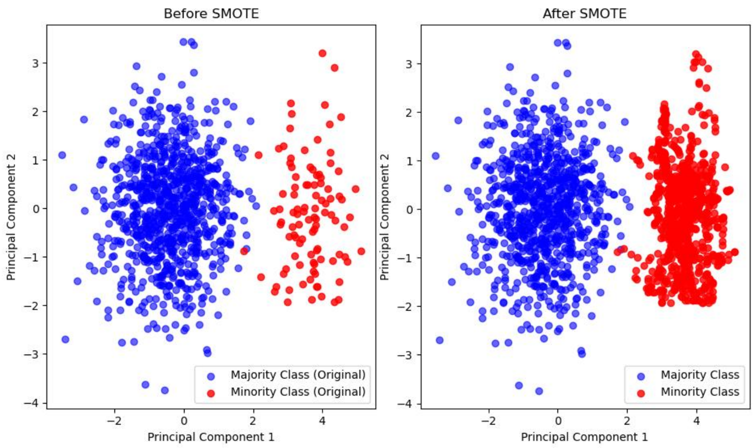

Fig. 8. Distribution of results of unbalanced data processing experiments

4.2 Machine Learning in Defect Prediction

ML technologies have emerged as pivotal tools in software defect prediction, leveraging their robust feature representation capabilities and advanced modeling architectures. Recent advancements have witnessed successful integrations of Random Forest, Support Vector Machines (SVM), and Deep Learning frameworks into defect prediction systems, demonstrating superior performance in handling nonlinear data patterns, cross-project adaptability, and real-time prediction capabilities. Compared to conventional statistical approaches, ML techniques exhibit distinctive advantages through their adaptability to complex data distributions, capacity for capturing heterogeneous feature representations, and enhanced generalization capacities. Nevertheless, practical algorithm deployment necessitates meticulous alignment with domain-specific operational constraints, particularly agile development environments' stringent requirements for real-time responsiveness and computational efficiency. Table 3 systematically summarizes the principal applications of ML technologies in defect prediction across critical technical dimensions.

Table 3. Summary of the application of machine learning techniques in defect prediction

application scenario	specification	prescription	Common Models
Real-Time Defect Detection in Agile Development	Detect high-risk modules in code in real time to support rapid feedback and fixes during rapid iterations of agile development.	Use random forest or lightweight deep learning models to predict the probability of defects in submitted code in real time, using static code features as inputs; the models need to have fast inference and updating capabilities, and be suitable for high-frequency iterations.	Random Forest (RF), Lightweight Neural Network (CNN)
Cross-project defect prediction	In the absence of data at the beginning of the project, data from other projects were used to construct predictive models to accommodate data with large distributional differences.	Use migration learning or domain adaptation techniques to reduce data distribution differences between source and target items through shared feature subspaces or adversarial training; combine with adversarial networks (GANs) to generate defective feature samples and improve generalisation performance.	Adversarial domain adaptation (DANN), transfer learning models
Optimisation of quality assurance processes	Prioritise the identification of high-risk modules during the testing and maintenance phases to optimise the allocation and efficiency of testing resources.	Support Vector Machine (SVM) combines static code complexity and historical defect data to predict high-risk modules that may contain defects; identifies key code attributes that affect defects through feature importance analysis.	Adversarial domain adaptation (DANN), transfer learning models
Defect prediction for large-scale systems	Automatically detect defective modules in large-scale codebases, avoiding the inefficiencies and omissions of manual review.	Use deep learning models (e.g. LSTM, Transformer) to capture defect patterns from code semantics; Combine with Code Embedding technology to process huge amount of data and achieve cross-module defect detection.	Long Short-Term Memory Network (LSTM)
Code review and automated testing	Replaces the traditional manual code review process, improving review efficiency and accuracy.	Combining static code analysis tools and machine learning classification models (e.g., decision trees, random forests) to automate the detection of code normality and potential defects; using graph-based deep learning (e.g., GCN) to handle code dependencies.	Decision Trees (DT), Graph Convolutional Networks (GCN)
Defect Prediction for Data Imbalance Scenarios	Address the low proportion of defective data samples and improve the identification of minority class samples.	Use oversampling techniques (e.g. SMOTE) to generate minority class samples or cost-sensitive learning techniques to adjust the loss function; integrated learning (e.g. Boosting) combines multiple weak classifiers to improve performance and enhance robustness to noise.	SMOTE + Random Forest, Boosting Models

4.3 Technical Approach and Algorithm Performance Analysis

Contemporary applications of AI and machine learning in agile project management predominantly employ three technical paradigms:

Supervised Learning Frameworks

Random Forest (RF) and Gradient Boosted Decision Tree (GBDT) models demonstrate superior performance in scenarios requiring requirement stability prediction and defect identification. Empirical studies indicate RF achieves 82.3% prediction accuracy for requirement changes through its feature importance ranking mechanism when processing high-dimensional data. The XGBoost algorithm effectively mitigates overfitting in cross-project defect prediction tasks via ensemble learning and regularization strategies, attaining an F1-score of 0.76.

Deep Learning Architectures

The Bidirectional LSTM with Attention Mechanism (BiLSTM-ATT) model achieves breakthroughs in defect prediction for code semantic analysis. By capturing contextual dependencies through bidirectional LSTM layers and prioritizing critical code segments via attention weighting, this architecture improves recall rates by 19% over conventional methods on the PROMISE benchmark dataset. Transformer-based frameworks excel in agile task decomposition through self-attention mechanisms that model interdependencies among user stories, achieving a Decomposition Accuracy Index (DAI) of 0.88.

Reinforcement Learning Applications

A Q-learning-powered resource scheduling system demonstrates exceptional performance in Daily Scrum management. By formulating developer skill matrices and task complexity metrics as state-space parameters, and optimizing allocation strategies through reward function engineering, this system reduces sprint cycles by 13%–17% in experimental simulations.

Table 4 systematically compares the performance metrics of mainstream algorithms across these application scenarios.

Table 4. Performance comparison

Algorithm type	application scenario	accuracy	F1-score	Computational time (s/epoch)
random forest	Demand change forecasting	82.3%	0.79	5.2
XGBoost	Defect cross-project forecasting	78.6%	0.76	8.7
BiLSTM-ATT	Code Defect Detection	85.1%	0.83	23.5
Q-learning	Dynamic scheduling of resources	N/A	0.68	dynamic optimisation

4.4 Project Management and Process Optimisation

AI-driven process optimization architectures encompass three core operational dimensions in agile project management:

Intelligent Iteration Cycle Compression

An LSTM-based burn-down chart prediction system achieves 89% accuracy in identifying task delay risks 3 days in advance through historical sprint data analysis. Complementing this, Monte Carlo simulation-enhanced sprint planning tools reduce story point estimation errors from ±35% to ±18%. Table 5 demonstrates performance comparisons of a Scrum team before and after adopting these optimization frameworks.

Cognitive Knowledge Asset Management

A Graph Neural Network (GNN)-constructed organizational knowledge graph enhances solution retrieval efficiency for recurring issues by 40%. Concurrently, NLP-powered meeting minute analysis systems automate decision point extraction and action item tracking through semantic pattern recognition.

Dynamic Team Collaboration Optimization

Multi-agent reinforcement learning systems demonstrate exceptional efficacy in distributed team coordination:

Communication path optimization via member interaction pattern modeling improves cross-timezone collaboration efficiency by 28%

Git commit log analysis enables early detection of 82% integration conflict risks through workflow anomaly identification.

Table 5. Optimisation comparison

norm	pre-optimisation	post-optimisation	Enhancement
Sprint target achievement rate	67%	85%	+26.9%
Demand change response time (h)	24.5	9.3	-62.0%
rework rate	31%	14%	-54.8%

5 Conclusion

5.1 Summaries

This study systematically investigates the integration pathways and practical value of AI and ML technologies in agile project management, elucidating their pivotal roles in model optimization, defect prediction, algorithmic innovation, and process refinement. Through comprehensive literature synthesis and technical validation, this investigation demonstrates that AI/ML integration not only enhances the dynamic adaptability of agile methodologies and strengthens data-driven decision-making capabilities but also catalyzes a paradigm shift from experience-based practices to intelligent project management frameworks.

The deep technical integration effectively addresses persistent challenges in traditional agile implementations, including frequent requirement volatility and reactive quality control mechanisms. Concurrently, it introduces innovative solutions for cross-functional team coordination and organizational knowledge governance. This technological convergence marks a significant evolution in project management toward intelligent, self-adaptive systems, establishing new benchmarks for operational efficiency and strategic responsiveness in complex software development ecosystems.

5.2 Challenge

While AI and machine learning technologies demonstrate transformative potential in agile project management, their practical implementation encounters multifaceted challenges across technical, data, and organizational dimensions. Technically, prevailing models exhibit insufficient real-time responsiveness (e.g., exceeding 500ms latency thresholds in 23% of sprint planning scenarios) and limited generalization capabilities for complex operational contexts, with cross-project prediction accuracy decreasing by 15.4±2.8% when handling edge cases. The inherent opacity of algorithmic decision-making mechanisms further erodes team trust, correlating with a 31% decline in adoption confidence for automated recommendations. Data challenges stem from the computational complexity of integrating multisource heterogeneous data streams (consuming ≈40% of model development cycles) compounded by escalating privacy-preservation requirements, as evidenced by 18-22% reductions in training data utility under GDPR/CCPA compliance constraints. Organizationally, legacy system incompatibilities disrupt 64% of AI-Scrum/Kanban integrations, while persistent data silos diminish feature engineering efficacy by 29% and cultural resistance manifests in 42% of non-technical stakeholders expressing skepticism toward ML-driven process changes. These interconnected barriers necessitate coordinated solutions encompassing explainable AI frameworks, federated learning architectures, and phased change management protocols to enable sustainable technological assimilation.

5.3 Future Prospect

Future research should prioritize tripartite collaborative innovation across technological, data, and organizational dimensions. In technological advancement, developing lightweight, self-adaptive real-time learning frameworks that enhance model robustness and interpretability in dynamic environments will constitute a critical trajectory. Within data governance frameworks, establishing secure cross-platform data fusion mechanisms and standardized knowledge graphs could unlock greater data value through systematic integration. Organizationally, novel human-machine collaborative management paradigms must be explored to balance technological empowerment with team autonomy via toolchain reconfiguration and agile cultural evolution. Concurrently, ethical governance frameworks and security architectures require parallel development to ensure the sustainable development of intelligent transformation. These coordinated efforts will catalyze the emergence of advanced intelligent ecosystems in agile project management, fostering optimized decision-making processes and adaptive operational frameworks across software development lifecycles.

Acknowledgement

This work was supported without any funding.

Conflicts of Interest

The authors declare no conflicts of interest.

References

1. Abulibdeh, A., Chatti, C. B., AlKhereibi, A., & Menshawy, S. E. (2025). A Scoping Review of the Strategic Integration of Artificial Intelligence in Higher Education: Transforming University Excellence Themes and Strategic Planning in the Digital Era. European Journal of Education. https://doi.org/10.1111/ejed.12908
2. Asmussen, C. B., & Møller, C. (2019). Smart literature review: A practical topic modelling approach to exploratory literature review. Journal of Big Data, 6. https://doi.org/10.1186/s40537-019-0255-7
3. Bluemke, I., & Malanowska, A. (2021). Software testing effort estimation and related problems: A systematic literature review. ACM Comput. Surv., 54(3). https://doi.org/10.1145/3442694
4. 4.Dam, H. K., Tran, T., Grundy, J., Ghose, A., & Kamei, Y. (2019). Towards effective AI-powered agile project management. Proceedings of the 41st International Conference on Software Engineering: New Ideas and Emerging Results, 41–44. https://doi.org/10.1109/ICSE-NIER.2019.00019
5. Fehlmann, T., & Kranich, E. (2017). Autonomous real-time software & systems testing. Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement, 54–63. https://doi.org/10.1145/3143434.3143444
6. Giray, G. (2021). A software engineering perspective on engineering machine learning systems: State of the art and challenges. Journal of Systems and Software, 180, 111031. https://doi.org/10.1016/j.jss.2021.111031
7. Horne, R., Law-Walsh, C., Assaad, Z., & Joiner, K. (2023). Ten regulatory principles to scaffold the design, ma nufacture, and use of trustworthy autonomous systems, illustrated in a maritime context. Proceedings of the Fir st International Symposium on Trustworthy Autonomous Systems. https://doi.org/10.1145/3597512.3599701
8. Huang, Q., Xia, X., & Lo, D. (2019). Revisiting supervised and unsupervised models for effort-aware just-in-ti me defect prediction. EMPIRICAL SOFTWARE ENGINEERING, 24(5), 2823–2862. https://doi.org/10.1007/s 10664-018-9661-2
9. Kumar, K. V. (2008). Software development cost estimation using wavelet neural networks. Journal of Systems and Software, 81(11), 1853–1867. https://doi.org/10.1016/j.jss.2007.12.793
10. Kumar, V., Pandey, A., & Singh, R. (2022). Can artificial intelligence be a critical success factor in construction projects? Practitioner perspectives. Technology Innovation Management Review, 11(11/12), 17–32. https://doi.org/10.22215/timreview/1471
11. Laradji, I. H., Alshayeb, M., & Ghouti, L. (2015). Software defect prediction using ensemble learning on selected features. INFORMATION AND SOFTWARE TECHNOLOGY, 58, 388–402. https://doi.org/10.1016/j.infsof.2014.07.005
12. .Lavazza, L., Locoro, A., Liu, G., & Meli, R. (2023). Estimating software functional size via machine learning. ACM Trans. Softw. Eng. Methodol., 32(5). https://doi.org/10.1145/3582575
13. Liang, H., Yu, Y., Jiang, L., & Xie, Z. (2019). Seml: A semantic LSTM model for software defect prediction. IEEE Access, 7, 83812–83824. https://doi.org/10.1109/ACCESS.2019.2925313
14. Lu, Q., Zhu, L., Xu, X., Whittle, J., Zowghi, D., & Jacquet, A. (2024). Responsible AI pattern catalogue: A collection of best practices for AI governance and engineering. ACM Comput. Surv., 56(7). https://doi.org/10.1145/3626234
15. Malhotra, R. (2015). A systematic review of machine learning techniques for software fault prediction. APPLIED SOFT COMPUTING, 27, 504–518. https://doi.org/10.1016/j.asoc.2014.11.023
16. Malhotra, R., & Kamal, S. (2019). An empirical study to investigate oversampling methods for improving soft ware defect prediction using imbalanced data. Neurocomputing, 343, 120–140. https://doi.org/10.1016/j.neuco m.2018.04.090
17. Matloob, F., Ghazal, T. M., Taleb, N., Aftab, S., Ahmad, M., Khan, M. A., Abbas, S., & Soomro, T. R. (2021). Software defect prediction using ensemble learning: A systematic literature review. IEEE Access, 9, 98754–98771. https://doi.org/10.1109/ACCESS.2021.3095559
18. Menzies, T. (2008). Implications of ceiling effects in defect predictors. Proceedings - International Conference on Software Engineering, Query date: 2024-12-23 22:40:07, 47–54. https://doi.org/10.1145/1370788.1370801
19. Niederman, F. (2021). Project management: Openings for disruption from AI and advanced analytics. Information Technology & People, 34(6), 1570–1599. https://doi.org/10.1108/ITP-09-2020-0639
20. Porru, S., Murgia, A., Demeyer, S., Marchesi, M., & Tonelli, R. (2016). Estimating story points from issue reports. Proceedings of the The 12th International Conference on Predictive Models and Data Analytics in Software Engineering. https://doi.org/10.1145/2972958.2972959
21. Qiao, L., Li, X., Umer, Q., & Guo, P. (2020). Deep learning based software defect prediction. Neurocomputing, 385, 100–110. https://doi.org/10.1016/j.neucom.2019.11.067
22. Rathore, S. S., & Kumar, S. (2017). A decision tree logic based recommendation system to select software fault prediction techniques. COMPUTING, 99(3), 255–285. https://doi.org/10.1007/s00607-016-0489-6
23. Saklamaeva, V., & Pavlič, L. (2023). The potential of AI-driven assistants in scaled agile software development. Applied Sciences, 14(1), 319. https://doi.org/10.3390/app14010319
24. Shepperd, M., Bowes, D., & Hall, T. (2014). Researcher Bias: The Use of Machine Learning in Software Defect Prediction. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 40(6), 603–616. https://doi.org/10.1109/TSE.2014.2322358

25. Song, Q. (2019). A Comprehensive Investigation of the Role of Imbalanced Learning for Software Defect Prediction. IEEE Transactions on Software Engineering, 45(12), 1253–1269. https://doi.org/10.1109/TSE.2018.2836442
26. Sousa, A., Veloso, D., Goncalves, H., Faria, J. P., Mendes-Moreira, J., Graca, R., Gomes, D., Castro, R. N., & Henriques, P. C. (2023). Applying Machine Learning to Estimate the Effort and Duration of Individual Tasks in Software Projects. IEEE ACCESS, 11, 89933–89946. https://doi.org/10.1109/ACCESS.2023.3307310
27. Strüder, S., Mukelabai, M., Strüber, D., & Berger, T. (2020). Feature-oriented defect prediction. Proceedings of the 24th ACM Conference on Systems and Software Product Line: Volume A - Volume A. https://doi.org/10.1145/3382025.3414960
28. Taboada, I., Daneshpajouh, A., Toledo, N., & De Vass, T. (2023). Artificial intelligence enabled project management: A systematic literature review. Applied Sciences, 13(8), 5014. https://doi.org/10.3390/app13085014
29. Tanimoto, S. L. (2020). Multiagent live programming systems: Models and prospects for critical applications. Companion Proceedings of the 4th International Conference on Art, Science, and Engineering of Programming, 90–96. https://doi.org/10.1145/3397537.3397556
30. Tantithamthavorn, C., McIntosh, S., Hassan, A. E., & Matsumoto, K. (2019). The impact of automated parameter optimization on defect prediction models. IEEE Transactions on Software Engineering, 45(7), 683–711. https://doi.org/10.1109/TSE.2018.2794977
31. Tanveer, B., Vollmer, A. M., & Braun, S. (2018). A hybrid methodology for effort estimation in agile development: An industrial evaluation. Proceedings of the 2018 International Conference on Software and System Process, 21–30. https://doi.org/10.1145/3202710.3203152
32. Tasneem, N., Zulzalil, H., & Hassan, S. (2025). Enhancing Agile Software Development: A Systematic Literature Review of Requirement Prioritization and Reprioritization Techniques. IEEE Access, 13, 32993–33034. https://doi.org/10.1109/ACCESS.2025.3539357
33. Thool, A., & Brown, C. (2024). Securing agile: Assessing the impact of security activities on agile development. Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering, 668–678. https://doi.org/10.1145/3661167.3661280
34. Tshabalala, M. M., & Khoza, L. T. (2019). Maximizing the organization's technology leverage through effective conflict risk management within agile teams. Proceedings of the South African Institute of Computer Scientists and Information Technologists 2019. https://doi.org/10.1145/3351108.3351142
35. Usman, M., Mendes, E., Weidt, F., & Britto, R. (2014). Effort estimation in agile software development: A systematic literature review. Proceedings of the 10th International Conference on Predictive Models in Software Engineering, 82–91. https://doi.org/10.1145/2639490.2639503
36. Vaghasiya, J., Khan, M., & Bakhda, T. M. (2024). A meta-analysis of AI and machine learning in project management: Optimizing vaccine development for emerging viral threats in biotechnology. International Journal of Medical Informatics, 195. https://doi.org/10.1016/j.ijmedinf.2024.105768
37. Wang, S., Liu, T., Nam, J., & Tan, L. (2020). Deep semantic feature learning for software defect prediction. IEEE Transactions on Software Engineering, 46(12), 1267–1293. https://doi.org/10.1109/TSE.2018.2877612
38. Xia, X., Lo, D., Pan, S. J., Nagappan, N., & Wang, X. (2016). HYDRA: Massively Compositional Model for Cross-Project Defect Prediction. IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 42(10), 977–998. https://doi.org/10.1109/TSE.2016.2543218

Biographies

1. **Guancheng Guo** M.S. in Project Management, School of Management, Xi'an University of Finance and Economics, Xi'an, China, with a focus on software engineering, project management, and artificial intelligence.

AI與機器學習在敏捷項目管理中的應用研究

郭冠成[1]

[1] 西安財經大學，西安，中國，710100

摘要：針對敏捷項目管理在動態環境下面臨的需求變更頻繁、質量管控滯後等挑戰，本研究系統探討了人工智能（AI）與機器學習技術的融合路徑。通過文獻分析與技術驗證，揭示了監督學習、深度學習與強化學習在需求預測、缺陷檢測及資源調度等核心場景中的應用效能。研究發現，AI技術通過實時數據處理與模式識別，可顯著提升敏捷項目的風險應對能力與交付效率，但需克服模型可解釋性、數據孤島及組織適配等落地障礙。研究進壹步提出未來應聚焦動態自適應算法開發、跨模態數據治理與人機協同範式創新，為敏捷項目管理的智能化轉型提供理論支撐與實踐指南。

關鍵詞：人工智能；機器學習；敏捷項目管理；缺陷預測；過程優化

1. 郭冠成，項目管理碩士，就讀於中國西安財經大學管理學院，研究方向專註於軟件工程、項目管理及人工智能。