International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 5, pp 394-399, May-2025

https://doi.org/10.47001/IRJIET/2025.905044

Software Effort Estimation in the Context of Methodology Agile and Software Development

[1]Shahad Wissam Abdulfattah Khattab, [2]*Jamal Salahaldeen Alneamy

[1,2]Department of Software Engineering, College of Computers Sciences and Mathematics, University of Mosul, Iraq

*Corresponding Author's E-mail: shahad.23csp11@student.uomosul.edu.iq

***Abstract* - Recent studies have increasingly focused on enhancing the accuracy of software project effort estimation within the Agile methodology framework. This trend emphasizes the integration of advanced machine learning and deep learning techniques, including neural networks and convolutional neural networks (CNNs). Additionally, optimization strategies—particularly in the early stages of modeling—have gained attention as a means to improve prediction outcomes. A central theme across these studies is the use of Story Points as a core metric for estimating software development effort. This research aims to explore and synthesize a selection of recent scholarly works that contribute to this evolving area, examining their methodologies, datasets, algorithms, and key findings.**

Keywords: Agile methodology, Software project, CNN, Convolutional neural networks, Machine learning, Deep learning.

I. INTRODUCTION

With the rapid advancements in the software industry and the increasing demand for efficient and effective software systems, both academics and practitioners have been motivated to develop robust methodologies aimed at enhancing the effectiveness of software project development, construction, and management.

Definition and Importance of Effort Estimation

Software effort estimation refers to the process of predicting the time and resources required to develop a software project. This process is based on multiple factors such as project size, complexity, and the development team's experience. Accurate estimation enhances resource management and reduces the risk of project delays or budget overruns (1).

Effort estimation is one of the fundamental tasks in software project management. It plays a critical role in determining the success or failure of a project. Inaccurate or low estimates often lead to delayed timelines, cost overruns, and failure to meet expected performance outcomes. Misestimating effort can result in significant resource waste (2).

Effort is typically measured in terms of man-hours or man-days, representing the cumulative time required by personnel to complete specific project tasks. It directly affects cost estimation and project timelines (3).

Influencing Factors

Numerous factors influence effort estimation, including:

- Project requirements and specifications
- Design and implementation strategies
- Experience, knowledge, and capabilities of the technical team
- Human factors such as team collaboration, technical skills, and communication efficiency (3)

Role of User Stories and Story Points in Agile

In Agile development, particularly within the Scrum framework, user stories are used to capture and describe user requirements in a simple, understandable manner. These stories play a vital role in estimating software effort and cost at a granular level.

Story points are a commonly used unit in Scrum to estimate the complexity and required effort of a user story. These estimates are relative and consider factors such as:

- Task complexity
- Team experience
- Associated risks and workload (4)

Estimation in Agile vs. Traditional Approaches

Unlike traditional methodologies that focus on estimating effort at the project level, Agile approaches emphasize estimating at the iteration or user story level. This shift introduces several challenges:

- Misallocation of resources
- Delivery of low-quality software due to poor estimation

ISSN (online): 2581-3048
Volume 9, Issue 5, pp 394-399, May-2025
https://doi.org/10.47001/IRJIET/2025.905044

- Limited estimation models adapted specifically for Agile environments (5)

Agile estimation models must account for the iterative and adaptive nature of frameworks like Scrum, where changes are frequent and estimation accuracy is essential. New Agile teams, in particular, may struggle due to inexperience and the lack of refined estimation techniques suited for Agile methods (6).

Challenges in Effort Estimation

Estimating effort in Agile environments remains complex due to:

- Lack of improvement in estimation accuracy over time despite accumulated experience
- No significant accuracy difference between tasks like feature development, bug fixing, and code refactoring
- Frequent underestimation of large tasks due to insufficient initial details
- Developers' tendency to be overconfident in their estimates
- Complexity and uncertainty associated with evolving requirements (9)

Role of Machine Learning and Advanced Techniques

Given the limitations of traditional estimation approaches (e.g., bias, inaccuracy), researchers have turned to advanced technologies such as machine learning, deep learning, and artificial intelligence to improve estimation accuracy. These technologies learn from historical data and identify hidden patterns to make more precise predictions.

Recent studies propose ensemble models that combine multiple machine learning techniques. For example, the Random Forest algorithm has been used as a super learner to enhance the predictive performance of effort estimation models (10).

Different Types of Methods Used in Estimating Software Effort

1. Algorithmic Methods

These methods rely on mathematical and statistical models to estimate the effort required for software development. They are often based on quantitative inputs and formula-based calculations.

Examples include:

COCOMO-II (Constructive Cost Model)

Putnam SLIM (Software Lifecycle Management)

The primary input to these models is the size of the software, which can be measured using:

- Function Points
- Lines of Source Code (LOC)
- Use Case Points

2. Non-Algorithmic Methods

These methods depend more on qualitative data and expert evaluations rather than mathematical models. They are often derived from historical project data and human judgment.

Examples include:

- Expert Judgment
- Planning Poker

These methods are especially useful in Agile environments, where flexibility and collaboration are essential.

3. Machine Learning Methods

Machine learning is a subfield of artificial intelligence that enables systems to learn from data and improve their performance without being explicitly programmed. It has wide applications in software engineering, including effort and cost estimation.

Machine learning is considered an alternative to traditional algorithmic models and includes techniques such as:

- Artificial Neural Networks (ANN)
- Case-Based Reasoning (CBR)
- Support Vector Regression (SVR)
- Decision Trees (DT)
- Genetic Algorithms (GA)

These methods can model complex, nonlinear relationships between input features and output effort values.

4. Deep Learning Methods

Deep learning is an advanced branch of machine learning that uses deep neural networks composed of multiple layers. These models are highly effective in extracting patterns and making predictions from large datasets.

Key technique:

- Long Short-Term Memory (LSTM) Networks: LSTM networks are particularly suited for handling sequential data. They are capable of retaining information over long

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 5, pp 394-399, May-2025

https://doi.org/10.47001/IRJIET/2025.905044

periods and are used to model temporal dependencies in software projects.

Deep learning has demonstrated strong performance in fields such as:

- Natural Language Processing (NLP)
- Image Recognition
- Medical Diagnosis

Its use in software effort estimation enables learning from large-scale project data to produce highly accurate predictions. (Sources: 11, 12, 13)

Software Development Life Cycle (SDLC)

The SDLC is a foundational framework for organizing the phases of software development. It includes:

- Planning
- Design
- Implementation
- Testing
- Deployment and Maintenance

The goal is to ensure high-quality, reliable, and cost-effective software products delivered within the project schedule. SDLC models form the backbone of systematic software engineering.

Traditional Methodologies

1. Waterfall Model: A linear, sequential approach where each phase must be completed before the next begins. While it features thorough documentation, it lacks flexibility.

2. Iterative Model: Involves incremental development and continuous feedback, allowing improvements with each cycle.

3. V-Model: An extension of the Waterfall model that places greater emphasis on validation and verification at each stage.

Modern (Agile) Methodologies

Agile methodologies focus on adaptive development, iterative cycles, and customer collaboration, enabling teams to quickly respond to changing requirements. Examples include:

- Scrum: A widely used Agile framework that organizes development into short cycles (Sprints), encouraging team collaboration.
- DevOps: A practice that combines development and IT operations to streamline collaboration and accelerate software delivery.(14, 15, 16, 17)

II. RELATED WORK

Estimating software effort in Agile methodology is an essential and fundamental component to ensure effective planning and efficient management of the resources involved in the project under development. This process is characterized by flexibility throughout the software development and construction phases. Effort estimation in Agile is typically based on collaborative and non-traditional approaches, such as the use of story points or task size comparisons. These techniques aim to improve estimation accuracy and support adaptability to continuous changes during the project lifecycle.

Neural Network Models for Agile Software Effort Estimation Based on Story Points (2015)

Researchers: Aditi Panda, Shashank MouliSatapathy, Santanu Kumar Rath

Data: 21 projects from 6 software companies – Variables: number of story points, project velocity, actual effort

Algorithms: GRNN, GMDH, Cascade Correlation

Results: GRNN: MSE = 0.0244, R^2 = 0.7125, MMRE = 0.3581, PRED = 85.91%

A Deep Learning Model for Estimating Story Points (2016)

Researchers: Morakot Choetkiertikul, HoaKhanh Dam, Truyen Tran, Trang Pham, Aditya Ghose, Tim Menzies

Data: 23,313 software issues from 16 open-source projects sourced from major repositories

Algorithms: LD-RNN (LSTM + RHN)

Results: Accuracy = 2.09 vs. 2.84 for conventional methods, SA = 52.66%

Estimating Story Points from Issue Reports (2016)

Researchers: Simone Porru, Alessandro Murgia, Serge Demeyer, Michele Marchesi, Roberto Tonelli

Data: One industrial project and eight open-source projects

Algorithms: Machine learning classifier

Results: Classifier was proven effective through experimental evaluation

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 5, pp 394-399, May-2025

https://doi.org/10.47001/IRJIET/2025.905044

A Proposed Framework for Enhancing Story Points in Agile Software Projects (2018)

Researchers: NismaGaffar, Hanan Moussa, Amr Kamel, Galal H. Galal-Edeen

Data: 12 epics and 88 user stories

Algorithms: Framework for Enhanced Story Points (ESP) based on uncertainty, functionality, and complexity

Results: Effort variance reduced from 61% to 9%, improved MRE and EV

SE3M: Story Point Level Classification by Text Level Graph Neural Network (2022)

Researchers: Hung Phan, Ali Jannesari

Data: Over 23,000 software issues from 16 projects

Algorithms: TextLevelGNN, TF-IDF-RF

Results: TF-IDF-RF accuracy: 80.25%, TextLevelGNN: 78.63%

Investigating the Effectiveness of Clustering for Story Point Estimation (2022)

Researchers: Vali Tawosi, Afnan Al-Subaihin, Federica Sarro

Data: 31,960 issues from 26 open-source projects

Algorithms: LDA, Hierarchical Clustering

Results: Performance similar to advanced models, better in some cases

Enhancing Software Effort Estimation with Pre-Trained Word Embeddings (2024)

Researchers: Issa Atoum, Ahmed Ali Otoom

Data: Small dataset with user stories

Algorithms: FastText, GPT-2, SVM

Results: MAE reduced by 5–18%, comparable to deep learning models

Search-based Optimization of LLM Learning Shots for Story Point Estimation (2024)

Researchers: Vali Tawosi, SalwaAlamir, Xiaomo Liu, Wan

Data: Three datasets for agile software tasks

Algorithms: SBSE to optimize few-shot examples for LLMs

Results: Performance improved by 59.34% compared to zero-shot learning

year	Research name	accuracy	Data size	Data type	Algorithms
2015	Neural Network Models for Agile Software Effort Estimation based on Story Points	MSE=0.0244, R^2=0.7125, MMRE=0.3581, PRED=85.91%	21 projects, 3 main variables	Story Points, Project Velocity, Actual Effort	GRNN, GMDH, Cascade Correlation
2016	A Deep Learning Model for Estimating Story Points	MAE=2.09, SA=52.66%	23,313 cases from 16 projects	Software Problem Description, Story Points	LSTM, RHN, LD-RNN
2018	A Proposed Framework for Enhancing Story Points in Agile Software Projects	59.34% improvement compared to traditional estimates	12 Epic and 88 User Stories	User Stories, Enhanced Story Points (ESP)	ESP Framework
2022	Investigating the Effectiveness of Clustering for Story Point Estimation	MMRE=0.0747, PRED=95.9052%	31,960 cases from 26 open source projects	Issues, titles and descriptions of software problems	LDA, Hierarchical Clustering, Cluster-based Estimation
2022	Story Point Level Classification by Text Level Graph Neural Network	TextLevelGNN: 78.63%, TFIDF-RF: 80.25%	23,313 cases from 16 projects	Issues, titles and descriptions of software problems	TextLevelGNN, TFIDF-RF
2024	Enhancing Software Effort Estimation with Pre-Trained Word Embeddings: A Small-Dataset Solution for	MAE between 2.18% and 2.92%, RMSE between 2.92% and 3.00%	21,070 samples from 16 projects	User Stories, Story Points	FastText, SVM, GPT-2, XGBoost

ISSN (online): 2581-3048
Volume 9, Issue 5, pp 394-399, May-2025
https://doi.org/10.47001/IRJIET/2025.905044

	Accurate Story Point Prediction				
2024	Search-based Optimization of LLM Learning Shots for Story Point Estimation	Improving estimation using CoGEE and NSGA-II algorithm	Data from Jira for Agile Projects	Jira Data for Agile Projects	GPT-4, CoGEE, NSGA-II

III. CONCLUSION

This study has reviewed various software effort estimation approaches within Agile development environments, with a particular focus on emerging deep learning-based techniques. Analysis of the existing literature reveals that while traditional estimation methods are still widely adopted, they face significant limitations—particularly in terms of accuracy and adaptability to the dynamic nature of Agile projects.

In contrast, artificial intelligence techniques, especially deep learning models, have demonstrated substantial potential in enhancing estimation precision and minimizing human biases. Nevertheless, several ongoing challenges remain, such as the requirement for high-quality datasets, the alignment of these models with the fast-paced and iterative nature of Agile development, and the need to strike a balance between predictive accuracy and model interpretability.

To address these challenges, fostering collaboration between academic researchers, software engineering practitioners, and Agile development teams is essential. Such interdisciplinary cooperation can drive the creation of more robust and practical effort estimation models, thereby contributing to more successful Agile project execution.

Furthermore, story points continue to serve as a foundational metric in Agile methodology. Unlike time-based estimation, story points account for complexity, risk, and uncertainty, thus enabling more adaptive and realistic planning. Leveraging deep learning models in conjunction with story point estimation presents a promising pathway to achieving highly accurate and context-aware effort predictions in Agile software development.

REFERENCES

[1] Ensemble Effort Estimation for Novice Agile Teams.

[2] Software Effort Estimation Using Machine Learning Technique.

[3] Analysis of Software Effort Estimation by Machine Learning Techniques.

[4] Survey of Cost Estimating Software Development Using Machine Learning.

[5] A Comprehensive Research Analysis of Software Development Life Cycle (SDLC): Agile & Waterfall Model Advantages.

[6] Disadvantages and Application Suitability in Software Quality Engineering.

[7] Neural Network Models for Agile Software Effort Estimation Based on Story Points.

[8] Investigating the Effectiveness of Clustering for Story Point Estimation.

[9] Story Point-Based Agile Software Effort Estimation Using Various SVR Kernel Methods.

[10] A Proposed Framework for Enhancing Story Points in Agile Software Projects.

[11] Story Point-Based Agile Software Effort Estimation Using Various SVR Kernel Methods.

[12] Story Point Effort Estimation by Text-Level Graph Neural Network.

[13] Enhancing Software Effort Estimation with Pre-Trained Word Embeddings.

[14] Search-Based Optimization of LLM Learning Shots for Story Point Estimation.

[15] An Optimized LSTM Neural Network for Accurate Estimation of Software Development Effort.

[16] Data-Driven Effort Estimation Techniques of Agile User Stories: A Systematic Literature Review.

[17] Review and Empirical Analysis of Machine Learning-Based Software Effort Estimation.

[18] Survey of Cost Estimating Software Development Using Machine Learning.

[19] Software Effort Estimation Using Ensemble Learning.

[20] Significant Factors in Agile Software Development of Effort Estimation.

[21] An Efficient Framework for Cost and Effort Estimation of Scrum Projects.

[22] Ensemble Effort Estimation for New Agile Teams.

[23] Analysis of Software Effort Estimation by Machine Learning Techniques.

[24] Estimating Efforts for Various Activities in Agile Software Development: An Empirical Study.

[25] Using Intelligence Techniques to Automate Oracle Testing.

[26] A Fundamental Diagram-Based Hybrid Framework for Traffic Flow Estimation and Prediction by Combining a Markovian Model with Deep Learning.

[27] Software Effort Estimation Using Machine Learning Technique.

ISSN (online): 2581-3048

Volume 9, Issue 5, pp 394-399, May-2025

https://doi.org/10.47001/IRJIET/2025.905044

[28] A Comprehensive Research Analysis of SDLC Agile & Waterfall Model: Advantages, Disadvantages, and Application Suitability in Software Quality Engineering.

[29] A Comprehensive Research Analysis of SDLC Agile & Waterfall Model: Advantages, Disadvantages, and Application Suitability in Software Quality Engineering.

[30] Survey of Brain Tumor Image Segmentation Using Artificial Intelligence Techniques.

[31] Introduction to Machine Learning, Neural Networks, and Deep Learning.

Citation of this Article:

Shahad Wissam Abdulfattah Khattab, & Jamal Salahaldeen Alneamy. (2025). Software Effort Estimation in the Context of Methodology Agile and Software Development. *International Research Journal of Innovations in Engineering and Technology - IRJIET*, 9(5), 394-399. Article DOI https://doi.org/10.47001/IRJIET/2025.905044

****

