Journal of Cyber Forensics and Security (CYFORS) *ISSN 3032-5870*
Vol.3, No.1 (2026)

SOFTWARE EFFORT ESTIMATION: A SYSTEMATIC LITERATURE AND RESEARCH DIRECTIONS

Agung Budi Prasetio [1] **, Luthfia Saskia** [2] **, Kenya Damayanti Priyatna** [3]

[1,2,3] *Institut Teknologi Tangerang Selatan*
email : agung@itts.ac.id[1],saskia@itts.ac.id[2], kenyadamayanti@itts.ac.id[3]

Abstract

Without sufficient effort, the development of a software project will be hindered and may even significantly fail, thus jeopardizing the quality of the software developers. Therefore, Software Effort Estimation (SEE) is a crucial activity in software engineering. Numerous studies have been conducted on SEE, resulting in a significant amount of new literature in a very short time. To gain a comprehensive understanding of current trends and frameworks in Software Effort Estimation (SEE) research, identifying the key areas of study is crucial. This research employed a systematic literature review (SLR) as a comprehensive approach to uncovering and analyzing all the prominent research topics within this domain. Six research topics were identified from 60 journals, including algorithmic techniques, machine learning implementation, statistical approaches, expert judgment, dataset analysis, and metric evaluation. The algorithmic techniques and machine learning approaches are the most frequently discussed topics

Keywords **:** *software effort estimation, literature review, research topic*

Introduction

In software engineering, estimating software development effort and costs is crucial for the success of a project. Both overestimation and underestimation pose problems for future software development. [1]. Software Development Effort Estimation (SDEE), also referred to as Software Effort Estimation (SEE) or various synonymous terms such as cost estimation, cost prediction, time estimation, and effort prediction [2]. Initially, software applications were confined to the domains of data processing and scientific computing. However, the past few decades have seen significant growth in software applications across various domains such as healthcare, food, and defense [3]. The 2020 Standish Report stated that only 31% of all software projects were successful, while 50% faced challenges [4]. In recent years, various SEE techniques have been classified and widely used [5], categorized into six groups: expert judgment methods, such as the Delphi technique [6] parametric models, such as the Constructive Cost Model (COCOMO) [7]; regression-based methods [8]; machine learning techniques [9]; analogy-based estimation [10]; and dynamic-based models [11]. Expert judgment methods include the Delphi technique, a structured communication technique. Parametric models, like COCOMO, use mathematical algorithms to calculate costs and effort. Input variables and project costs are connected through regression- based methods using statistical techniques [12]. A review article on SEE by Rashid [13] discusses the widely used frameworks in SEE, highlighting the advantages and disadvantages associated with using these frameworks. Carbonera et al. [6] present a comprehensive review of SEE approaches, identifying research gaps, challenges, and trends. A total of 120 primary studies were selected, analyzed, and categorized after a meticulous screening process from a sample of 3746 candidate studies to answer six research questions. Another study revealed that among 52 papers on recent research trends related to SEE, Artificial Neural Network (ANN) models and Constructive Cost Model (COCOMO)- based approaches have become the preferred techniques [14], underscoring the importance of cost and effort estimation in software development. Jadhav et al. [15] proposed a generic automated text mining framework to investigate research trends by analyzing the titles, keywords, and abstracts of 1015 selected research papers on Software Development Effort and Cost Estimation (SDECE) published over the past five decades. To update the summary of all papers from the preset search directory on SEE from 2020 to 2024, this research employs a systematic literature review (SLR). A systematic literature review (SLR) is a method for discovering, evaluating, and understanding all available research on a specific subject [16].

Research Methods

A. Review Method

A Systematic Literature Review (SLR), a well-known review method in the field of software engineering, is defined as the process of collecting, evaluating, and analyzing all existing research evidence to answer specific research questions [17]. This literature review was conducted following the guidelines for systematic literature reviews proposed by Kitchenham et al. [18].

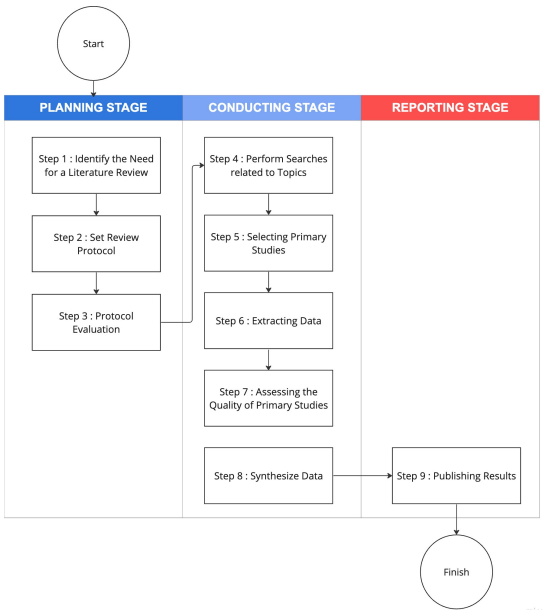


Fig. 1. Systematic Literature Review Steps

As shown in Figure 1, an SLR is conducted in three stages: planning, execution, and reporting the review results. The first step is to establish criteria for the systematic review (Step 1). A review protocol is developed to organize the review's execution and minimize researcher bias (Step 2). This protocol includes the formulation of research questions, search strategies, study selection process with inclusion and exclusion criteria, quality assessment, and the data extraction and synthesis process.

B. Research Questions

In scientific research, questions serve many important functions. They guide the research and ensure that the researcher focuses on relevant and specific aspects of the topic or issue under review [19]. The PICOC framework (Population, Intervention, Comparison, Outcome, and Context), previously applied by Abusaeed [20] and Diego [21] as a reference for SLR research, is used to develop research questions. Research questions are created using the PICOC platform, and relevant keywords for searching primary studies on SEE are identified. Table 1 illustrates the PICOC structure used.

Table 1. PICOC CRITERIA

PICOC	**Detail**
Population	Research focusing on Software Effort Estimation(SEE)
Intervention	Analysis of research trends and topics discussed in SEE-related publications.
Comparison	No explicit comparison is made in this context asthe primary goal is to identify and analyze trends.
Outcome	Identification of research trends in SEE over thepast five years (2020-2024). Analysis of the most frequently discussed topics in SEE research.
Context	Scientific publications, including journals, conferences, and papers published between 2020 and 2024.

After establishing the PICOC framework, various research questions have been identified along with their respective objectives, as illustrated in Table 2.

Table II. RESEARCH QUESTION

ID	***Research Question***	***Motivation***
RQ1	What have been the research trends in SEE over the past five years and what topics have been most discussed?	To discover SEE research trends from 2020 to 2024 and analyze the most discussed topics in SEEresearch.
RQ2	What research issues, methods, and datasets have been explored in SEE?	The focus of this question is to analyze and explored research isseus, methods, and datasets datasets have been widely used in SEE

C. Research Strategy

The process of determining keywords begins by referring to terms originating from the PICOC framework, especially from the "Population" and "Intervention" sections. The next step is to extract relevant terms from the research question andlook for words that have the same or similar meaning. “(software OR system) AND (effort OR cost) AND (predictedOR machine learning OR estimating OR estimation OR strategy)” are keywords that are generated by using logical operators to get the appropriate results. There are several digital scientific databases, such as Science Direct, IEEE, Springer, and Wiley, searched with these keywords. The focusof this research is English-language Q1 and Q2 journals published between 2020 and 2024

D. Inclusion and Exclusion Criteria

The selection of primary studies for this research involved a meticulous process of defining inclusion and exclusion criteria. Inclusion criteria encompassed full-length journal and conference papers, as well as research publications specifically focused on SEE. Conversely, exclusion criteria eliminated non-scientific publications and works that addressed cost estimation beyond the scope of this research, such as those unrelated to software engineering. From this selection process, 60 primary studies were identified and chosen. Based on titles and abstracts, the search results from the search strategy were manually re-screened, with the criteria for inclusion and exclusion shown in Table 3. Only journal papers were selected after the peer-review assessment.

Table III. INCLUSION AND EXCLUSION CRITERIA

Selection	***Criteria***
Inclusion	Journal Paper Public Available or Private
Exclusion	Conference paper, review, survey, book chapter, and comparative study The topic outrange of SEE

After identifying relevant studies through search results and manual selection (Figure 2), the research process moved to data extraction and synthesis. Data extraction involved meticulously gathering all the information needed to answer the research questions.

This could include details like methodologies, results, and conclusions from the selected studies. Data synthesis then built upon this extracted information. It involved analyzing the findings across all the included studies and grouping them into key research themes. Through this process, the research was able to identify and categorize the most significant topics explored within the field of SEE.

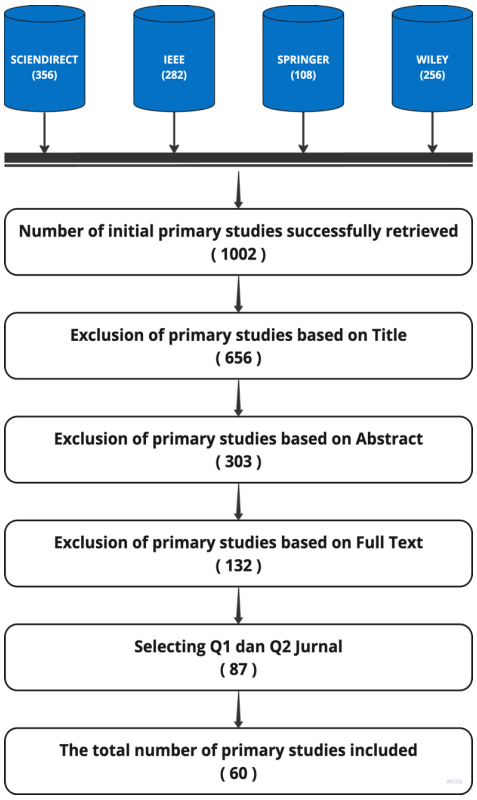


Fig. 2. Search and Selection of Primary Studies

Results and Discussion

This section explores the key findings gleaned from the reviewed papers, directly addressing the two research questions we set out to answer. To ensure the review's rigor, we focused on papers published in high-impact Q1 and Q2 journals. As seen in Figure 3, the number of SEE-related publications has steadily increased from 2020 to 2024, with atotal of 60 papers identified within this timeframe.

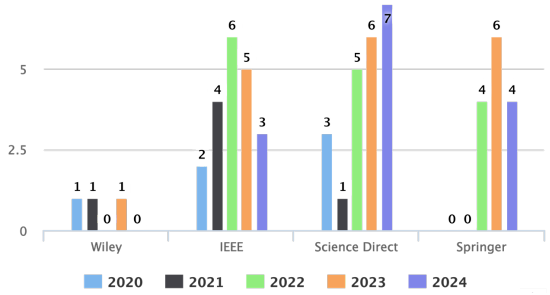


Fig. 3. Paper publication

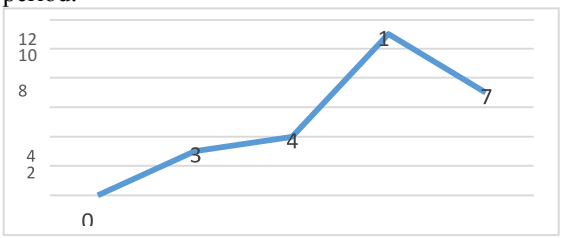
A. Research Trends

To address RQ1, the authors identified six research topics, which are represented in Table 4. The most frequently discussed topic is the algorithm technique approach with 27 papers, followed by machine learning implementation with 25 papers, statistical approach with 11 papers, expert judgment with 8 papers, dataset analysis with 5 papers, and finally 3 papers on matrix evaluation.

TABLE IV. SIX RESEARCH TOPICS

ID	*Research Topics*	*Numberof Paper*	*References*
RT1	Machine Learning Implementation	25	[11], [12], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44]
RT2	Dataset analysis	6	[45], [46], [47], [47], [48], [49]
RT3	StatisticalApproach	11	[37], [38], [44], [47], [48], [50], [51], [52], [53], [54], [55]
RT4	Evaluation Matric	3	[3], [56], [57]
RT5	Algoritmic Technique	27	[10], [11], [11], [14], [20], [27], [31], [32], [33], [34], [35], [37], [38], [47], [53], [58], [59], [60], [61], [62], [63], [64], [65], [66], [67], [68], [69]
RT6	Expert Judgement	8	[49], [55], [65], [66], [70], [71], [72], [73]

The implementation of machine learning (RT1) for SEE has been extensively explored by researchers. The number of papers published on this research topic from 2020 to 2024 is 25. As depicted in Figure 4, this data indicates fluctuations in the number of studies, which could be attributed to technological advancements and researcher interest during that period.



The most widely explored topics include Extreme Learning Machine (ELM) ,Deep Neural Network (DNN) [23], Artificial Neural Network (ANN) [24], hybrid model [27], [11], ensemble learning [27], dan Deep Learning [32]. Rankovic et al. [40] compared different GNN models (LSTM, GGNN, GGSNN) with various settings and analyzed the results using SHAP, achieving a balance between efficiency and accuracy using the Taguchi optimization method. Additionally, there are opportunities for further studies on the challenges of identifying reliable machine learning approaches for software development effort estimation. Statistical approaches (RT4) play a crucial role in SEE, providing robust methods for predicting the time and resources required to complete a project. Numerous studies have been conducted in this area, includingAlqasrawi et al. [50] which employed Locally weighted regression (LWR) with different kernels, followed by estimation models using a trained embedding model [51], multiple linear regression [52], and cluster

analysis [54]. Further research can explore other statistical approaches such as classification and association analysis. Three papers were successfully summarized, including J. A. Khan et al. [23] which achieved the best results using MMRE and Pred(25) on eight attributes, security matrix evaluation in SEE [57], and [3] which identified MMRE, PRED, and RMSE as the three most popular performance metrics. Research on evaluation metrics (RT4) has not been extensively discussed in recent years, but it remains an important area for exploration.

B. Research issues, methods, and datasets in SEE

Various studies have been conducted by researchers to answer this problem on RQ2, further issues related to the evaluation matrix in SEE, three papers were successfully summarized, including J. A. Khan et al. [23] which achieved the best results using MMRE and Pred(25) on eight attributes, security matrix evaluation in SEE [57], and [3] which identified MMRE, PRED, and RMSE as the three most popular performance metrics. Research on evaluation metrics (RT4) has not been extensively discussed in recent years, but it remains an important area for exploration. In (RT2), six papers were found to discuss dataset analysis. This will also discuss RQ2. No discussion of this topic was found in 2021, 2022, and 2024. It was observed that all datasets were public, and the identified topics included data comparison [45], feature selection [46], [47], feature relevance [47], data size [48], and dataset quality [49]. The limitations of dataset availability remain an interesting trend for further research, especially in the area of feature selection. Other potential topics for further exploration include: Methods for improving feature elimination in datasets, and Synthetic data augmentation for software development effort estimation. Algorithmic techniques (RT5) for SEE are the most extensively researched issues and method in this SLR. There were 27 papers published on this topic from 2020 to 2024. As can be seen in Figure 5, the most research was conducted in 2023 with 12 papers, indicating that algorithmic techniques remain a highly interesting area for further analysis in recent years.

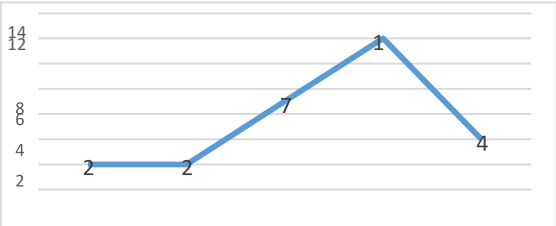


Fig. 5. Algorithmic Technique

Among the numerous algorithms employed for SEE, Artificial Bee Colony [47], [59] is the most frequently discussed trend topic. However, other trends in algorithmic techniques include Dolphin Algorithm [58], Particle Swarm Optimization [61], fuzzy C-means [62], dimensional success factors [63], wavelet neural network and metaheuristic algorithm [27], CoGEE [64], Fuzzy AHP [20], genetic algorithm [31], Use Case Points [66], Advanced Bayesian Network [37], function point [67], TSoptEE [68], and finally Self-Organizing Migration [14]. The two papers that implemented the Bee Colony algorithm took different approaches. The first paper, by Matsubara et al. [47] proposed binary artificial bee colony based feature selection (BAFS), and the authors stated that the Binary Artificial Bee Colony-based Feature Selection (BAFS) method outperforms existing feature selection models and regression techniques in terms of software development effort estimation performance. BAFS employs four regression models (Linear, LASSO, Ridge, ElasticNet) for more effective feature selection. Shah et al. [59] discussed Ensembling Artificial Bee Colony With Analogy-Based (BABE), and the authors stated that the model improves estimation performance compared to previous models, particularly on the International Software Benchmarking Standards Group (ISBSG) dataset. Future research could explore combining Artificial Bee Colony (ABC) with other estimation techniques besides Analogy-Based Estimation. This could involve integrating ABC with regression techniques, machine learning methods, or other analytical approaches to investigate whether such combinations can produce more accurate or efficient results. Expert judgment (RT6) also plays a prominent role in SEE research, encompassing task size-based estimation [70], analogy-based estimation [71], [65], developer experience , SMO [55], and UCP [66]. Iqbal et al. [49] shed light on issues in SEE using story-based estimation, identifying internal factors such as communication, team expertise, and team composition as crucial determinants of estimation accuracy. The authors emphasize the importance of considering the interplay of various aspects, including team dynamics, task complexity, and task engineering practices, to achieve accurate estimation. Further research opportunities can focus on standardizing estimation protocols and leveraging supporting technologies for SEE.

Conclusions and Recommendations

This study guided a systematic literature review to identify current trends in Software Effort Estimation (SEE) research. The review process involved a comprehensive analysis of 60 academic journals published among 2020 and 2024. The findings revealed a clear focus on leveraging computational techniques for SEE. The six most prevalent research topics, in descending order of popularity, reflected this trend:

1. Algorithmic Techniques (27 papers): This field examines various algorithms, including the Artificial Bee Colony and genetic algorithms, to automate the estimation process and potentially enhance accuracy.

2. Machine Learning Implementation (25 papers): This research direction explores the use of machine learning models, such as Deep Learning and Support Vector Machines, to learn from historical data and provide more accurate effort estimations.
3. Statistical Approach (11 papers): This traditional approach applies statistical methods, like linear regression and cluster analysis, to identify relationships between project characteristics and effort requirements.
4. Expert Judgment (8 papers): This area recognizes the importance of human expertise in software effort estimation (SEE) and investigates techniques such as story-based estimation, which leverage the knowledge and experience of software development teams.
5. Dataset Analysis (6 papers): Although fewer studies exist in this area, it underscores the importance of high-quality data for training computational models to achieve accurate SEE. Research focuses on data cleaning, feature selection, and identifying potential biases.
6. Evaluation Metrics (3 papers): This less explored area is essential for evaluating the effectiveness of different SEE approaches. Research here focuses on developing and applying appropriate metrics to assess the accuracy and efficiency of estimation methods.

The observed dominance of computational techniques suggests a growing interest in automating and potentially improving the accuracy of SEE. However, the continued exploration of traditional approaches and the importance of human expertise and data quality demonstrate the need for a comprehensive research landscape that considers various facets of SEE.

References

[1] S. S. Gautam and V. Singh, "The state-of-the-art in software development effort estimation," J. Softw. Evol. Process, vol. 30, no. 12, p. e1983, Dec. 2018, doi: 10.1002/smr.1983.

[2] Y. Swandari, R. Ferdiana, and A. E. Permanasari, "Research Trends in Software Development Effort Estimation," in 2023 10th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), Palembang, Indonesia: IEEE, Sep. 2023, pp. 625–630. doi: 10.1109/EECSI59885.2023.10295716.

[3] S. Shukla and S. Kumar, "Study of Learning Techniques for Effort Estimation in Object-Oriented Software Development," IEEE Trans. Eng. Manag., pp. 1–17, 2022, doi: 10.1109/TEM.2022.3217570.

[4] K. Rak, Ž. Car, and I. Lovrek, "Effort estimation model for software development projects based on use case reuse," J. Softw. Evol. Process, vol. 31, no. 2, p. e2119, Feb. 2019, doi: 10.1002/smr.2119.

[5] T. Menzies, Y. Yang, G. Mathew, B. Boehm, and J. Hihn, "Negative results for software effort estimation," Empir. Softw. Eng., vol. 22, no. 5, pp. 2658–2683, Oct. 2017, doi: 10.1007/s10664-016-9472-2.

[6] C. Eduardo Carbonera, K. Farias, and V. Bischoff, "Software development effort estimation: a systematic mapping study," IET Softw., vol. 14, no. 4, pp. 328–344, Aug. 2020, doi: 10.1049/iet-sen.2018.5334.

[7] S. K. Sehra, Y. S. Brar, N. Kaur, and S. S. Sehra, "Research patterns and trends in software effort estimation," Inf. Softw. Technol., vol. 91, pp. 1–21, Nov. 2017, doi: 10.1016/j.infsof.2017.06.002.

[8] Z. Shahpar, V. K. Bardsiri, and A. K. Bardsiri, "Polynomial analogy- based software development effort estimation using combined particle swarm optimization and simulated annealing," Concurr. Comput. Pract. Exp., vol. 33, no. 20, p. e6358, Oct. 2021, doi: 10.1002/cpe.6358.

[9] A. Jadhav and S. K. Shandilya, "Reliable machine learning models for estimating effective software development efforts: A comparative analysis," J. Eng. Res., vol. 11, no. 4, pp. 362–376, Dec. 2023, doi: 10.1016/j.jer.2023.100150.

[10] K. H. Kumar and K. Srinivas, "Preliminary performance study of a brief review on machine learning techniques for analogy based software effort estimation," J. Ambient Intell. Humaniz. Comput., vol. 14, no. 3, pp. 2141–2165, Mar. 2023, doi: 10.1007/s12652-021- 03427-y.

[11] S. Sharma and S. Vijayvargiya, "Modeling of software project effort estimation: a comparative performance evaluation of optimized soft computing-based methods," Int. J. Inf. Technol., vol. 14, no. 5, pp. 2487–2496, Aug. 2022, doi: 10.1007/s41870-022-00962-5.

[12] H. L. T. K. Nhung, V. Van Hai, P. Silhavy, Z. Prokopova, and R. Silhavy, "Incorporating statistical and machine learning techniques into the optimization of correction factors for software development effort estimation," J. Softw. Evol. Process, p. e2611, Aug. 2023, doi: 10.1002/smr.2611.

[13] C. H. Rashid et al., "Software Cost and Effort Estimation: Current Approaches and Future Trends," IEEE Access, vol. 11, pp. 99268– 99288, 2023, doi: 10.1109/ACCESS.2023.3312716.

[14] D. Bajusova, P. Silhavy, and R. Silhavy, "Enhancing Software Effort Estimation With Self-Organizing Migration Algorithm: A Comparative Analysis of COCOMO Models," IEEE Access, vol. 12, pp. 67170–67188, 2024, doi: 10.1109/ACCESS.2024.3399060.

[15] A. Jadhav, M. Kaur, and F. Akter, "Evolution of Software Development Effort and Cost Estimation Techniques: Five Decades Study Using Automated Text Mining Approach," Math. Probl. Eng., vol. 2022, pp. 1–17, May 2022, doi: 10.1155/2022/5782587.

[16] R. Van Dinter, B. Tekinerdogan, and C. Catal, "Automation of systematic literature reviews: A systematic literature review," Inf. Softw. Technol., vol. 136, p. 106589, Aug. 2021, doi: 10.1016/j.infsof.2021.106589.

[17] R. S. Wahono, "A Systematic Literature Review of Software Defect Prediction: Research Trends, Datasets, Methods and Frameworks," J. Softw. Eng., vol. 1, no. 1, 2015.

[18] B. Kitchenham et al., "Systematic literature reviews in software engineering – A tertiary study," Inf. Softw. Technol., vol. 52, no. 8, pp. 792–805, Aug. 2010, doi: 10.1016/j.infsof.2010.03.006.

[19] Y. Mahmood, N. Kama, and A. Azmi, "A systematic review of studies on use case points and expert-based estimation of software development effort," J. Softw. Evol. Process, vol. 32, no. 7, p. e2245, Jul. 2020, doi: 10.1002/smr.2245.

[20] 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, Jan. 2023, doi: 10.1016/j.asoc.2022.109977.

[21] "Fernandez-Diego et al. - 2020 - An Update on Effort Estimation in Agile Software D.pdf."

[22] H. D. P. De Carvalho, R. Fagundes, and W. Santos, "Extreme Learning Machine Applied to Software Development Effort Estimation," IEEE Access, vol. 9, pp. 92676–92687, 2021, doi: 10.1109/ACCESS.2021.3091313.

[23] M. S. Khan, F. Jabeen, S. Ghouzali, Z. Rehman, S. Naz, and W. Abdul, "Metaheuristic Algorithms in Optimizing Deep Neural Network Model for Software Effort Estimation," IEEE Access, vol. 9, pp. 60309–60327, 2021, doi: 10.1109/ACCESS.2021.3072380.

[24] N. Rankovic, D. Rankovic, M. Ivanovic, and L. Lazic, "A New Approach to Software Effort Estimation Using Different Artificial Neural Network Architectures and Taguchi Orthogonal Arrays," IEEE Access, vol. 9, pp. 26926–26936, 2021, doi: 10.1109/ACCESS.2021.3057807.

[25] S. S. Gautam and V. Singh, "Adaptive Discretization Using Golden Section to Aid Outlier Detection for Software Development Effort Estimation," IEEE Access, vol. 10, pp. 90369–90387, 2022, doi: 10.1109/ACCESS.2022.3200149.

[26] A. Kaushik, P. Kaur, N. Choudhary, and Priyanka, "Stacking regularization in analogy-based software effort estimation," Soft Comput., vol. 26, no. 3, pp. 1197–1216, Feb. 2022, doi: 10.1007/s00500-021-06564-w.

[27] A. Kaushik and N. Singal, "A hybrid model of wavelet neural network and metaheuristic algorithm for software development effort estimation," Int. J. Inf. Technol., vol. 14, no. 3, pp. 1689–1698, May 2022, doi: 10.1007/s41870-019-00339-1.

[28] I. Abnane, A. Idri, I. Chlioui, and A. Abran, "Evaluating ensemble imputation in software effort estimation," Empir. Softw. Eng., vol. 28, no. 2, p. 56, Mar. 2023, doi: 10.1007/s10664-022-10260-0.

[29] S. S. Ali, J. Ren, K. Zhang, J. Wu, and C. Liu, "Heterogeneous Ensemble Model to Optimize Software Effort Estimation Accuracy," IEEE Access, vol. 11, pp. 27759–27792, 2023, doi: 10.1109/ACCESS.2023.3256533.

[30] U. S. B. and R. Sadam, "How far does the predictive decision impact the software project? The cost, service time, and failure analysis from a cross-project defect prediction model," J. Syst. Softw., vol. 195, p. 111522, Jan. 2023, doi: 10.1016/j.jss.2022.111522.

[31] S. Hameed, Y. Elsheikh, and M. Azzeh, "An optimized case-based software project effort estimation using genetic algorithm," Inf. Softw. Technol., vol. 153, p. 107088, Jan. 2023, doi: 10.1016/j.infsof.2022.107088.

[32] H. T. Hoc, R. Silhavy, Z. Prokopova, and P. Silhavy, "Comparing Stacking Ensemble and Deep Learning for Software Project Effort Estimation," IEEE Access, vol. 11, pp. 60590–60604, 2023, doi: 10.1109/ACCESS.2023.3286372.

[33] A. Jadhav, S. K. Shandilya, and I. Izonin, "Effective Software Effort Estimation Leveraging Machine Learning for Digital Transformation," vol. 11, 2023.

[34] S. Kassaymeh, M. Alweshah, M. A. Al-Betar, A. I. Hammouri, and M. A. Al-Ma'aitah, "Software effort estimation modeling and fully connected artificial neural network optimization using soft computing techniques," Clust. Comput., vol. 27, no. 1, pp. 737–760, Feb. 2024, doi: 10.1007/s10586-023-03979-y.

[35] K. H. Kumar and K. Srinivas, "An accurate analogy based software effort estimation using hybrid optimization and machine learning techniques," Multimed. Tools Appl., vol. 82, no. 20, pp. 30463–30490, Aug. 2023, doi: 10.1007/s11042-023-14522-x.

[36] A. O. Sousa et al., "Applying Machine Learning to Estimate the Effort and Duration of Individual Tasks in Software Projects," IEEE Access, vol. 11, pp. 89933–89946, 2023, doi: 10.1109/ACCESS.2023.3307310.

[37] M. Turic, S. Celar, S. Dragicevic, and L. Vickovic, "Advanced Bayesian Network for Task Effort Estimation in Agile Software Development," Appl. Sci., vol. 13, no. 16, p. 9465, Aug. 2023, doi: 10.3390/app13169465.

[38] A. G. Priya Varshini and K. Anitha Kumari, "Software Effort Estimation Using Stacked Ensemble Technique and Hybrid Principal Component Regression and Multivariate Adaptive Regression Splines," Wirel. Pers. Commun., vol. 134, no. 4, pp. 2259–2278, Feb. 2024, doi: 10.1007/s11277-024-11010-9.

[39] N. Rankovic and D. Rankovic, "Power of LSTM and SHAP in the Use Case Point Approach for Software Effort and Cost Estimation," in 2024 IEEE 22nd World Symposium on Applied Machine Intelligence and Informatics (SAMI), Stará Lesná, Slovakia: IEEE, Jan. 2024, pp. 000059–000064. doi: 10.1109/SAMI60510.2024.10432878.

[40] N. Rankovic, D. Rankovic, M. Ivanovic, and J. Kaljevic, "Interpretable software estimation with graph neural networks and orthogonal array tunning method," Inf. Process. Manag., vol. 61, no. 5, p. 103778, Sep. 2024, doi: 10.1016/j.ipm.2024.103778.

[41] B. Alsaadi and K. Saeedi, "Ensemble effort estimation for novice agile teams," Inf. Softw. Technol., vol. 170, p. 107447, Jun. 2024, doi: 10.1016/j.infsof.2024.107447.

[42] C. Anitha and N. Parveen, "Deep artificial neural network based multilayer gated recurrent model for effective prediction of software development effort," Multimed. Tools Appl., Jan. 2024, doi: 10.1007/s11042-024-18120-3.

[43] K. Harish Kumar and K. Srinivas, "An improved analogy-rule based software effort estimation using

HTRR-RNN in software project management," Expert Syst. Appl., vol. 251, p. 124107, Oct. 2024, doi: 10.1016/j.eswa.2024.124107.

[44] E. I. Mustafa and R. Osman, "A random forest model for early-stage software effort estimation for the SEERA dataset," Inf. Softw. Technol., vol. 169, p. 107413, May 2024, doi: 10.1016/j.infsof.2024.107413.

[45] N. Mittas and L. Angelis, "Data-driven benchmarking in software development effort estimation: The few define the bulk," J. Softw. Evol. Process, vol. 32, no. 9, p. e2258, Sep. 2020, doi: 10.1002/smr.2258.

[46] S. Tariq, M. Usman, and A. C. M. Fong, "Selecting best predictors from large software repositories for highly accurate software effort estimation," J. Softw. Evol. Process, vol. 32, no. 10, p. e2271, Oct. 2020, doi: 10.1002/smr.2271.

[47] P. Manchala, M. Bisi, and S. Agrawal, "BAFS: binary artificial bee colony based feature selection approach to estimate software development effort," Int. J. Inf. Technol., vol. 15, no. 6, pp. 2975– 2986, Aug. 2023, doi: 10.1007/s41870-023-01369-6.

[48] W. Rosa and S. Jardine, "Data-driven agile software cost estimation models for DHS and DoD☆," 2023.

[49] M. Iqbal et al., "Exploring issues of story-based effort estimation in Agile Software Development (ASD)," Sci. Comput. Program., vol. 236, p. 103114, Sep. 2024, doi: 10.1016/j.scico.2024.103114.

[50] Y. Alqasrawi, M. Azzeh, and Y. Elsheikh, "Locally weighted regression with different kernel smoothers for software effort estimation," Sci. Comput. Program., vol. 214, p. 102744, Feb. 2022, doi: 10.1016/j.scico.2021.102744.

[51] E. M. De Bortoli Fávero, D. Casanova, and A. R. Pimentel, "SE 3 M: A model for software effort estimation using pre-trained embedding models," Inf. Softw. Technol., vol. 147, p. 106886, Jul. 2022, doi: 10.1016/j.infsof.2022.106886.

[52] H. L. T. K. Nhung, V. Van Hai, R. Silhavy, Z. Prokopova, and P. Silhavy, "Parametric Software Effort Estimation Based on Optimizing Correction Factors and Multiple Linear Regression," IEEE Access, vol. 10, pp. 2963–2986, 2022, doi: 10.1109/ACCESS.2021.3139183.

[53] W. Rosa, B. K. Clark, R. Madachy, and B. W. Boehm, "Empirical Effort and Schedule Estimation Models for Agile Processes in the US DoD," IEEE Trans. Softw. Eng., vol. 48, no. 8, pp. 3117–3130, Aug. 2022, doi: 10.1109/TSE.2021.3080666.

[54] V. Van Hai, H. L. T. K. Nhung, Z. Prokopova, R. Silhavy, and P. Silhavy, "Toward Improving the Efficiency of Software Development Effort Estimation via Clustering Analysis," IEEE Access, vol. 10, pp. 83249–83264, 2022, doi: 10.1109/ACCESS.2022.3185393.

[55] T. Xia, R. Shu, X. Shen, and T. Menzies, "Sequential Model Optimization for Software Effort Estimation," IEEE Trans. Softw. Eng., vol. 48, no. 6, pp. 1994–2009, Jun. 2022, doi: 10.1109/TSE.2020.3047072.

[56] J. A. Khan, S. U. R. Khan, J. Iqbal, and I. U. Rehman, "Empirical Investigation About the Factors Affecting the Cost Estimation in Global Software Development Context," IEEE Access, vol. 9, pp. 22274–22294, 2021, doi: 10.1109/ACCESS.2021.3055858.

[57] E. Venson, B. Clark, and B. Boehm, "The effects of required security on software development effort," J. Syst. Softw., vol. 207, p. 111874, Jan. 2024, doi: 10.1016/j.jss.2023.111874.

[58] A. A. Fadhil, R. G. H. Alsarraj, and A. M. Altaie, "Software Cost Estimation Based on Dolphin Algorithm," IEEE Access, vol. 8, pp. 75279–75287, 2020, doi: 10.1109/ACCESS.2020.2988867.

[59] M. A. Shah, D. N. A. Jawawi, M. A. Isa, M. Younas, A. Abdelmaboud, and F. Sholichin, "Ensembling Artificial Bee Colony With Analogy-Based Estimation to Improve Software Development Effort Prediction," IEEE Access, vol. 8, pp. 58402–58415, 2020, doi: 10.1109/ACCESS.2020.2980236.

[60] S. P. Singh, V. P. Singh, and A. K. Mehta, "Differential evolution using homeostasis adaption based mutation operator and its application for software cost estimation," J. King Saud Univ. - Comput. Inf. Sci., vol. 33, no. 6, pp. 740–752, Jul. 2021, doi: 10.1016/j.jksuci.2018.05.009.

[61] A. Ardiansyah, R. Ferdiana, and A. E. Permanasari, "MUCPSO: A Modified Chaotic Particle Swarm Optimization with Uniform Initialization for Optimizing Software Effort Estimation," Appl. Sci., vol. 12, no. 3, p. 1081, Jan. 2022, doi: 10.3390/app12031081.

[62] S. K. Gouda and A. K. Mehta, "Software cost estimation model based on fuzzy C-means and improved self adaptive differential evolution algorithm," Int. J. Inf. Technol., vol. 14, no. 4, pp. 2171– 2182, Jun. 2022, doi: 10.1007/s41870-022-00882-4.

[63] N. Govil and A. Sharma, "Estimation of cost and development effort in Scrum-based software projects considering dimensional success factors," Adv. Eng. Softw., vol. 172, p. 103209, Oct. 2022, doi: 10.1016/j.advengsoft.2022.103209.

[64] V. Tawosi, F. Sarro, A. Petrozziello, and M. Harman, "Multi- Objective Software Effort Estimation: A Replication Study," IEEE Trans. Softw. Eng., vol. 48, no. 8, pp. 3185–3205, Aug. 2022, doi: 10.1109/TSE.2021.3083360.

[65] N. Pal, M. P. Yadav, and D. K. Yadav, "Appropriate number of analogues in analogy based software effort estimation using quality datasets," Clust. Comput., Jan. 2023, doi: 10.1007/s10586-023-03967-2.

[66] R. Silhavy, M. Bures, M. Alipio, and P. Silhavy, "More Accurate Cost Estimation for Internet of Things Projects by Adaptation of Use Case Points Methodology," IEEE Internet Things J., vol. 10, no. 21, pp. 19312–19327, Nov. 2023, doi: 10.1109/JIOT.2023.3281614.

[67] J. F. Vijay, "Retraction Note: Enrichment of accurate software effort estimation using fuzzy-based function point analysis in business data analytics," Neural Comput. Appl., vol. 35, no. 6, pp. 4797–4797, Feb. 2023, doi: 10.1007/s00521-022-08159-4.

[68] P. Manchala and M. Bisi, "TSoptEE: two-stage optimization technique for software development effort estimation," Clust. Comput., Apr. 2024, doi: 10.1007/s10586-024-04418-2.

[69] M. Azzeh, A. Alkhateeb, and A. Bou Nassif, "Software effort estimation using convolutional neural network and fuzzy clustering," Neural Comput. Appl., May 2024, doi: 10.1007/s00521-024-09855- z.

[70] M. Jørgensen and T. Halkjelsvik, "Sequence effects in the estimation of software development effort," J.

Syst. Softw., vol. 159, p. 110448, Jan. 2020, doi: 10.1016/j.jss.2019.110448.

[71] P. Phannachitta, "On an optimal analogy-based software effort estimation," Inf. Softw. Technol., vol. 125, p. 106330, Sep. 2020, doi: 10.1016/j.infsof.2020.106330.

[72] S. A. Butt et al., "A software-based cost estimation technique in scrum using a developer's expertise," Adv. Eng. Softw., vol. 171, p. 103159, Sep. 2022, doi: 10.1016/j.advengsoft.2022.103159.

[73] J. Cubillos, J. Aponte, D. Gomez, and E. Rojas, "Agile effort estimation in Colombia: An assessment and opportunities for improvement," Sci. Comput. Program., vol. 236, p. 103115, Sep. 2024, doi: 10.1016/j.scico.2024.103115.