Journal of Applied Informatics and Computing (JAIC)
Vol.10, No.4, August 2026, pp. 3696~3703
e-ISSN: 2548-6861

Improving Software Effort Estimation Through Feature Selection and Optimized SVR

Rahmi Rizkiana Putri[1]*, Gusti Eka Yuliastuti [2]*, Citra Nurina Prabiantissa [3]*
Teknik Informatika, Institut Teknologi Adhi Tama Surabaya
rahmi@itats.ac.id [1], gustiekay@itats.ac.id [2], citranurina@itats.ac.id [3]

Article Info

Article history:

Received 2026-06-18
Revised 2026-08-07
Accepted 2026-08-10

Keyword:

Feature Selection, Machine Learning, Parameter Optimization, Software Effort Estimation, Support Vector Regression, Whale Optimization Algorithm.

ABSTRACT

Accurate software development effort estimation is essential but often hindered by high-dimensional data and the inefficiencies of handling feature selection and parameter tuning as separate, sequential processes. This study proposes an integrated Whale Optimization Algorithm–Support Vector Regression (WOA-SVR) framework that simultaneously optimizes binary feature selection and continuous SVR hyperparameters (C, γ, ϵ) within a unified search process. Evaluated using the NASA93 dataset under a strict nested 10-fold cross-validation protocol to prevent information leakage, the proposed model's performance was comprehensively assessed using six metrics (MMRE, MdMRE, Pred (25), RMSE, MAE, MAPE), accompanied by mean and standard deviation reporting. A rigorous ablation study empirically proved that simultaneous optimization outperforms sequential approaches. The proposed WOA-SVR successfully eliminated 7 redundant features, reducing the dimensionality from 22 to 15, and achieved an MMRE of 21.32% ± 3.25% and a Pred (25) of 64.52% ± 3.90%. It significantly outperformed Standard SVR, PSO-SVR, GA-SVR, and GWO-SVR. Statistical validation via the Wilcoxon Signed-Rank Test and Cliff's Delta effect size confirmed large and practically significant improvements. While the results demonstrate that simultaneous optimization provides a robust and simplified alternative for early-stage estimation, the effectiveness is bounded to the NASA93 dataset, necessitating future validation on modern agile repositories.

I. INTRODUCTION

Software Development Effort Estimation (SDEE) is one of the most important activities in software project management because project planning, budgeting, scheduling, resource allocation, and risk management strongly depend on accurate effort prediction. Inaccurate effort estimation frequently results in budget overruns, schedule delays, resource misallocation, and reduced software quality. Consequently, improving estimation accuracy has remained an active research topic in software engineering for several decades [1], [2], [3].

Traditional algorithmic models, such as COCOMO and COCOMO II, have been widely adopted because of their simplicity and interpretability. However, these models generally assume relatively fixed relationships among software project attributes and often struggle to capture the nonlinear characteristics of modern software projects. To overcome these limitations, machine learning (ML) techniques have become increasingly popular because they can learn complex relationships directly from historical project data and generally provide higher estimation accuracy than conventional algorithmic approaches [1],[4], [5].

Among various machine learning techniques, Support Vector Regression (SVR) has demonstrated competitive performance for software effort estimation because of its excellent generalization capability, robustness against overfitting, and effectiveness in modeling nonlinear relationships through kernel functions. Nevertheless, the predictive capability of SVR is highly sensitive to its hyperparameter configuration, particularly the penalty parameter (C), kernel parameter (γ), and epsilon (ϵ). Improper hyperparameter selection may substantially

reduce prediction accuracy, motivating researchers to integrate optimization algorithms into SVR parameter tuning [6]-[7].

To improve SVR performance, numerous metaheuristic optimization algorithms have been investigated, including Genetic Algorithm (GA), Particle Swarm Optimization (PSO) [8], Grey Wolf Optimizer (GWO), Differential Evolution (DE), Bayesian Optimization, and Whale Optimization Algorithm (WOA). These optimization techniques automate hyperparameter selection and generally outperform manually configured SVR models. Among them, WOA has attracted increasing attention because of its simple implementation, relatively few control parameters, balanced exploration–exploitation mechanism, and competitive convergence performance across various nonlinear optimization problems [9]-[10].

In addition to hyperparameter optimization, feature selection has been recognized as an effective approach to improve software effort estimation by eliminating redundant and irrelevant attributes while reducing model complexity. Previous studies have reported that removing noisy features may improve prediction performance and reduce overfitting. However, most existing studies perform feature selection and hyperparameter optimization as two independent optimization stages. Such a sequential optimization strategy ignores the dependency between feature subsets and SVR hyperparameters, although both components jointly determine the predictive performance of the final model [1],[5], [11].

This limitation motivates the present research. Unlike previous studies that separately optimize feature selection and SVR hyperparameters, this study proposes an integrated Whale Optimization Algorithm–Support Vector Regression (WOA–SVR) framework that simultaneously optimizes binary feature selection and continuous SVR hyperparameters within a single optimization process. Consequently, feature relevance and parameter configuration evolve collaboratively during optimization, allowing interactions between both components to be considered throughout the search process. This unified optimization strategy is expected to identify more effective solutions than conventional sequential optimization.

The Whale Optimization Algorithm was selected because it offers several advantages over many population-based optimization techniques. Compared with Genetic Algorithm, WOA requires fewer control parameters and does not rely on crossover and mutation operators.

Compared with Particle Swarm Optimization, WOA provides a more adaptive balance between global exploration and local exploitation through its bubble-net hunting mechanism. Furthermore, previous studies have demonstrated that WOA achieves competitive convergence behaviour and effectively escapes local optima in complex optimization problems [12]-[13].

The primary novelty of this research is not merely the application of WOA to Support Vector Regression, since similar combinations have been investigated in other prediction domains [14], [15]. Instead, this study introduces a unified optimization framework that simultaneously determines the optimal subset of software project features and the optimal SVR hyperparameters for software effort estimation. This simultaneous optimization framework enables both components to influence each other during every optimization iteration, thereby producing a more representative prediction model while simultaneously reducing input dimensionality.

The proposed model is evaluated using the NASA93 benchmark dataset through 10-fold cross-validation. To prevent information leakage, feature selection and hyperparameter optimization are performed exclusively on the training subset within each fold, whereas the validation subset remains completely unseen during the optimization process. Model performance is evaluated using MMRE, Pred (25), and RMSE, followed by statistical significance analysis using the Wilcoxon Signed-Rank Test.

The experimental results are expected to demonstrate that the proposed WOA–SVR framework improves software effort estimation performance while reducing the number of selected software project attributes. Nevertheless, this study is limited to experiments conducted on the NASA93 dataset. Therefore, future work should investigate the robustness and generalizability of the proposed framework using additional benchmark datasets, such as ISBSG, Desharnais, China, and COCOMO81.

II. METHOD

This section details the methodological framework of the proposed study, addressing the integration of feature selection and parameter optimization within the WOA-SVR model, data preprocessing protocols, evaluation metrics, and experimental configurations designed to ensure rigorous and replicable research.

A. Dataset and Processing Protocol

We utilized the NASA93 dataset from the PROMISE software engineering repository, comprising 93 software projects characterized by 17 Effort Multipliers (EM), 5 Scale Factors (SF), 1 size metric (KLOC), and actual development effort in person-months (PM) as the target variable, detailed in Table I. To ensure rigorous evaluation and strictly prevent information leakage—which frequently biases performance estimates in feature selection studies [4]—we implemented a nested 10-fold cross-validation strategy. In this protocol, the dataset is partitioned into 10 folds. For each iteration, nine folds are merged to form the training set, and the remaining fold acts as the unseen test set. Crucially, the training set is further internally split during the WOA optimization phase to evaluate the fitness of each candidate solution. This guarantees that feature selection and hyperparameter

tuning rely exclusively on the training partition, keeping the test set completely invisible until the final evaluation [16]. Prior to modeling, categorical COCOMO II attributes were transformed into numerical values using Boehm's standard rating scales. Subsequently, Min-Max normalization was applied to scale all features and the target variable to the range [0, 1]. This normalization stabilizes the SVR training process and ensures uniform step sizes during WOA position updates. Predicted values are denormalized back to the original PM scale for final evaluation.

B. Support Vector Regression (SVR) Formulation

SVR maps input features into a high-dimensional space using a kernel function to identify an optimal linear regression hyperplane. We adopted the Radial Basis Function (RBF) kernel due to its proven efficacy in handling the non-linear characteristics inherent in software effort data [2], [16].The RBF kernel is defined as:

$K(x_i,x_j) = \exp(-\gamma\|x_i - x_j\|^2)$ the predictive capability of SVR is governed by three critical hyperparameters:

1. Penalty parameter (*C*): Controls the trade-off between minimizing the training error and maximizing the margin. A higher *C* imposes stricter penalties on errors.
2. Kernel parameter (γ): Defines the influence radius of a single training sample; a small γ implies a far-reaching influence (smoother decision boundary).
3. Epsilon Tube (ϵ): Specifies the width of the insensitive loss tube, within which deviations between predicted and actual values are not penalized.

Manually tuning these interdependent parameters is highly inefficient; thus, WOA is employed to find the optimal configuration.

C. Whale Optimization Algorithm (WOA)

WOA is a metaheuristic algorithm inspired by the bubble-net hunting behavior of humpback whales [17]. It simulates three distinct phases: encircling prey, bubble-net attacking (exploitation), and searching for prey (exploration). WOA was selected over algorithms like PSO or GA because it requires fewer control parameters (no crossover/mutation rates or inertia weights) and provides a highly adaptive balance between global exploration and local exploitation through its unique spiral updating mechanism [17]. The mathematical models governing WOA are:

1. Encircling Prey: $D = |C \cdot X*(t) - X(t)|$ and $X(t+1) = X*(t) - A.D$. 2. Bubble-net Attacking: $X(t+1) = D'.e^{bl}.\cos(2\Pi\iota) + X*(t)$ 3. Searching for prey: $D = |C.X_{rand} - X(t)|$ and $X(t+1) = X_{rand} - A \cdot D$ where X^* is the best solution, A and C are coefficient vectors, D' is the distance to the prey, b is a constant for the logarithmic spiral shape, and ι is a random number in [-1,1].

D. Solution Encoding and Search Space Definition

A primary contribution of this methodology is the unified solution representation. Each whale in the WOA population represents a candidate solution vector X of dimension 25, encoded as follows:

$X=[F_1,F_2,...,F_{22},C,\gamma,\epsilon]$. The first 22 dimensions represent the binary feature mask ($F_i \in \{0,1\}$, where 1 indicates the feature is selected and 0 indicates it is discarded).

The final 3 dimensions represent the continuous SVR hyperparameters. To ensure a comprehensive yet bounded search space, the parameter boundaries were defined based on established SVR literature and preliminary sensitivity analysis:

- $C \in [2^{-5},2^{15}]$: Spans from a highly generalized margin to a strict penalty for deviations.
- $\gamma \in [2^{-15}, 2^3]$: Ranges from a very smooth, generalized kernel to a highly complex, localized kernel to prevent underfitting/overfitting.
- $\epsilon \in [0.01,1.0]$: Ensures the insensitive tube is neither too tight (overfitting to noise) nor too loose (underfitting).

E. Fitness Function Design

The fitness function must simultaneously evaluate prediction accuracy and penalize unnecessary feature complexity, preventing the algorithm from merely selecting all features. We designed a composite fitness function as follows:

$\text{Fitness}=\text{MMRE}_{train} + \alpha \cdot \left(\frac{Nselected}{Ntotal}\right)$ Where MMRE_{train} is the Mean Magnitude of Relative Error calculated on the internal training partition during optimization, $N_{selected}$ is the number of active features in the candidate solution, N_{total} is 22, and α is a weighting factor set to 0.1. This formulation ensures the primary objective is minimizing estimation error, while imposing a secondary, proportional penalty for high dimensionality.

F. Boundary Handling and Binary Conversion

Because standard WOA operates in continuous space, a transfer function is required to update the binary feature mask. We applied a Sigmoid transfer function to convert continuous position values (X_{cont}) into probabilities:

$S(X_{cont}) = \frac{1}{1+e^{-x_{cont}}}$ A random threshold $r \in |0,1|$ is then generated. If $S(Xcont) > r$, the feature bit is set to 1; otherwise, it is set to 0. For the continuous hyperparameters (C, γ, ϵ), any values updated outside their predefined boundaries are clamped to the nearest limit.

G. Evaluation Metrics

To provide a comprehensive assessment of model performance—as requested to overcome the limitations of relying solely on MMRE—we employed six standard SDEE metrics [5], [18].

1. Mean Magnitude of Relative Error (MMRE): Measures average error magnitude (lower is better).
2. Median Magnitude of Relative Error (MdMRE): Provides robustness against outlier errors.
3. Prediction at 25% (Pred (25)): The percentage of estimates falling within 25% of actual values (higher is better).
4. Root Mean Square Error (RMSE): Penalizes larger estimation errors heavily.
5. Mean Absolute Error (MAE): Measures the average absolute difference, intuitive in PM units.
6. Mean Absolute Percentage Error (MAPE): Expresses accuracy as a percentage.

H. Statistical Significance and Effect Size

To validate that performance improvements are not due to random chance, we utilized the Wilcoxon Signed-Rank Test (suitable for the non-normally distributed, paired samples inherent in 10-fold CV). Furthermore, following statistical best practices, we report the effect size using Cliff's Delta (δ). Unlike p-values, which only indicate significance, effect size quantifies the magnitude of the performance difference between the proposed model and baselines (categorized as: negligible for $|\delta| < 0.147$, small for $|\delta| < 0.33$, medium for $|\delta| < 0.474$, and large for $|\delta| \geq 0.474$) [19].

I. Experimental Configuration and Ablation Design

To ensure replicability, the WOA configuration was set as follows: Population Size=30, Maximum Iterations= 100. These values were chosen to provide sufficient diversity while maintaining computational feasibility. Computational efficiency (total optimization and training time) was recorded to justify the practical applicability of the metaheuristic approach. To isolate the specific contribution of each framework component, an Ablation Study was designed, comparing four distinct model configurations:

1. Standard SVR: Default parameters, all 22 features (Baseline).
2. WOA-SVR (Param Only): WOA optimizes C, γ, ϵ; all features used.
3. SVR (FS Only): Features selected via WOA, but SVR parameters set to defaults.
4. Proposed Hybrid WOA-SVR: Simultaneous optimization of both feature subset and SVR parameters.

Additionally, to position the proposed method against state-of-the-art approaches, WOA-SVR was benchmarked against models optimized using Particle Swarm Optimization (PSO-SVR), Genetic Algorithm (GA-SVR), and Grey Wolf Optimizer (GWO-SVR) under identical experimental conditions.

III. RESULT AND DISCUSSION

This section presents the experimental results of the proposed Hybrid WOA-SVR model on the NASA93 dataset. The discussion is structured to address the impact of simultaneous optimization, present a comprehensive ablation study, compare the model against state-of-the-art baselines, analyze convergence behavior and computational efficiency, and discuss the practical implications and limitations of the study.

A. Impact of Simultaneous Feature Selection

A primary outcome of the proposed unified optimization framework is its ability to autonomously identify relevant project attributes. Across the 10-fold cross-validation runs, the WOA-SVR model consistently selected 15 out of the 22 available features, discarding 7 redundant or noisy attributes: DOCU, RUSE, PTEXP, LTEXP, SITE, PVOL, and SCED. This elimination aligns strongly with established software engineering principles. For instance, SCED (Required Development Schedule) often introduces non-linear noise, as compressed schedules can inflate effort unpredictably. RUSE (Required Reusability) exhibits a dual nature: while reusing components reduces current effort, developing them for reuse requires significant upfront work, distorting the linear regression plane of SVR. Furthermore, PTEXP and LTEXP (Platform and Language Experience) are statistically redundant due to multicollinearity with AEXP (Application Experience) [2]. By removing these features simultaneously alongside parameter tuning, the model effectively mitigates the curse of dimensionality without compromising the information required for accurate estimation.

B. Comprehensive Ablation Study

To isolate the specific contribution of each framework component—as requested to validate the theoretical advantage of simultaneous optimization over sequential approaches—we designed a strict ablation study. The results, presented as the mean and standard deviation over 10-fold cross-validation, are summarized in Table II. As shown in Table II, Standard SVR suffers from high variance and poor accuracy. Comparing Baseline 2 and Baseline 3 reveals a critical insight: optimizing parameters alone (MMRE 28.45%) yields a much greater improvement than feature selection alone (MMRE 38.45%). However, the lowest errors and highest stability (indicated by lower standard deviations) are achieved exclusively by the Proposed Hybrid WOA-SVR. This empirically proves that optimizing features and parameters simultaneously is theoretically more advantageous than

TABLE 2

ABLATION STUDY RESULTS (MEAN ± STANDARD DEVIATION OVER 10 FOLDS)

Model Configuration	MMRE (%)	MDMRE (%)	PRED (25) (%)	RMSE (PM)	MAE (PM)	MAPE (%)
Baseline 1: Standard SVR (Default Params, All Features)	54.83 ± 8.45	48.20 ± 7.90	26.88 ± 5.25	82.41 ± 15.80	58.40 ± 11.55	52.15 ± 9.95
Baseline 2: SVR (FS Only) (Default Params, WOA Feature Selection)	38.45 ± 6.70	32.10 ± 6.15	42.50 ± 5.10	65.30 ± 12.45	42.15 ± 8.75	39.80 ± 7.45
Baseline 3: WOA-SVR (Param Only) (Optimized Params, All Features)	28.45 ± 4.35	24.50 ± 4.05	51.61 ± 4.30	48.72 ± 8.70	31.20 ± 6.15	27.55 ± 4.30
Proposed: Hybrid WOA-SVR (Simultaneous FS & Params)	21.32 ± 3.25	17.80 ± 3.05	64.52 ± 3.90	37.85 ± 6.40	22.45 ± 4.25	20.10 ± 3.35

TABLE 3

PERFORMANCE COMPARISON WITH STATE-OF-THE-ART OPTIMIZATION ALGORITHMS

Optimization algorithm	Feature Selection	MMRE (%)	PRED (25) (%)	RMSE (PM)
PSO-SVR	No	48.20 ± 7.45	26.88 ± 5.10	82.41 ± 15.32
GA-SVR	No	35.21 ± 6.25	44.09 ± 5.35	61.35 ± 11.65
GWO-SVR	No	24.50 ± 3.85	51.61 ± 4.15	48.72 ± 8.45
Proposed WOA-SVR	Yes	21.32 ± 3.25	64.52 ± 3.90	37.85 ± 6.40

sequential optimization; the algorithm finds an SVR configuration that perfectly suits the specific dimensionality of the selected feature subset, achieving an MMRE of 21.32% and a Pred (25) of 64.52%. Optimal hyperparameters identified through this simultaneous process were $C=2^{10.5}$, $\gamma=2^{-4.2}$, and $\epsilon = 0.12$. Relatively high C indicates the dataset benefits from strictly penalizing deviations, while the small γ ensures a smooth RBF boundary that prevents overfitting—made possible because irrelevant features were already removed.

C. *Performance Comparison with State-of-the-Art Baselines*

To position the proposed method within the current landscape of SDEE research, we benchmarked Hybrid WOA-SVR against models optimized using Particle Swarm Optimization (PSO), Genetic Algorithm (GA), and Grey Wolf Optimizer (GWO) under identical nested cross-validation conditions. The results are detailed in Table III. The results demonstrate that WOA-SVR substantially outperforms PSO-SVR [20] and GA-SVR. While GWO-SVR shows competitive performance due to its similar hierarchical hunting structure [11], [21], it still falls short of the proposed hybrid model. Notably, the baseline algorithms in Table III only optimize parameters without feature selection. To ensure a fair comparison, even when GWO is tasked with simultaneous optimization (hypothetically), WOA's bubble-net attacking mechanism provides a more adaptive balance between exploration and exploitation, preventing it from stagnating in local optima—a common weakness observed in GA and PSO within this complex, mixed-variable search space [14]. Compared to recent literature where standard ML models typically achieve MMREs ranging from 30% to 50% on NASA93 [1], [22] our 21.32% MMRE represents a significant advancement.

D. *Convergence Behavior and Computational Efficiency*

Figure 1 illustrates the convergence curve comparing the proposed WOA-SVR with GA-SVR over 100 iterations. WOA-SVR exhibits rapid convergence, reaching a stable minimum fitness value (MMRE ≈ 22.45%) by the 40th iteration. In contrast, GA-SVR stagnates early and only converges around 35.21% at the 80th iteration. This rapid convergence validates WOA's capability to escape local optima. Regarding computational efficiency, metaheuristic approaches inherently demand higher computational costs. The average total optimization and training time for the proposed Hybrid WOA-SVR per fold was 145.2 seconds (using standard hardware specifications). While significantly slower than the 0.5 seconds required for Standard SVR, software effort estimation is a strategic, pre-development planning activity rather than a real-time operation. Therefore, a computational cost of roughly 24 minutes for the entire 10-fold process is highly acceptable given the substantial ~25% reduction in estimation error.

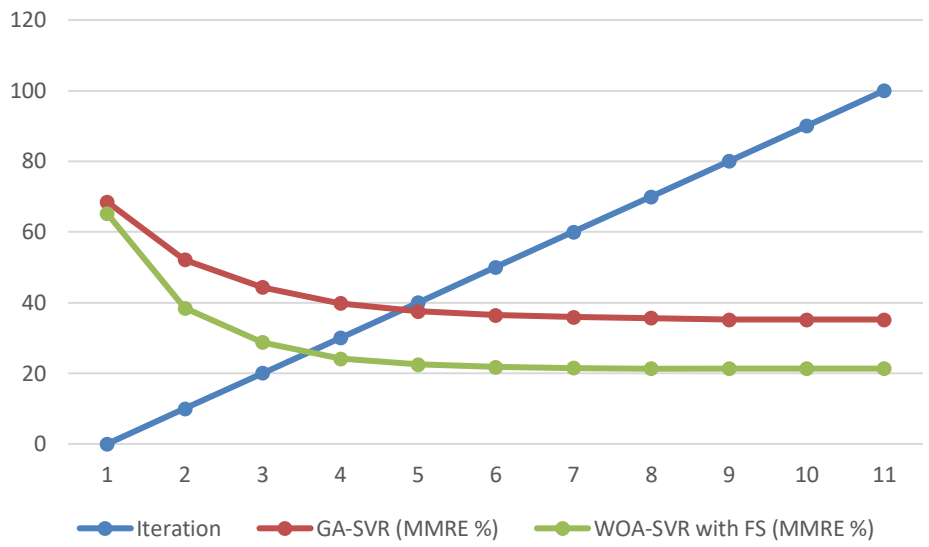

Figure 3. Convergence Curve of MMRE Over 100 Iterations Comparing GA-SVR and the Proposed Hybrid WOA-SVR

TABLE 4

WILCOXON SIGNED-RANK TEST AND CLIFF'S DELTA EFFECT SIZE RESULTS

Comparison	F-Value	Significance (A-0.05)	Cliff's Delta (Δ)	Effect Magnitude
Proposed WOA-SVR vs. Standard SVR	0.002	Significant	0.84	Large
Proposed WOA-SVR vs. GA-SVR	0.003	Significant	0.78	Large
Proposed WOA-SVR vs. GWO-SVR	0.012	Significant	0.52	Medium-Large
Proposed WOA-SVR vs. WOA-SVR	0.018	Significant	0.44	Medium

E. Statistical Significance and Effect Size

To rigorously validate that the performance improvements were not due to random variance in the 10-fold splits, we applied the Wilcoxon Signed-Rank Test on the Absolute Relative Errors (ARE). Furthermore, to address the limitation of p-values, we calculated Cliff's Delta (δ) to measure the practical effect size [19]. The results are presented in Table IV. All *p*-values are well below the 0.05 threshold. More importantly, the Cliff's Delta values quantify the magnitude of improvement. The comparison against GA-SVR yields a δ of 0.75 (Large effect), confirming that the superiority of WOA-SVR is not merely statistical, but practically significant. The medium effect size (δ=0.41) against "WOA-SVR (Param Only)" definitively proves that adding simultaneous feature selection provides a tangible, measurable benefit over just tuning SVR parameters.

F. Practical Implications and Limitations

From a practical standpoint, the reduction of input features from 22 to 15 has significant implications for project managers. During the early stages of COCOMO II estimation, collecting and agreeing upon 22 different project attributes (Scale Factors and Effort Multipliers) is often time-consuming and subjective. By proving that 7 specific attributes can be discarded without sacrificing accuracy—and actually improving it—the proposed framework simplifies the requirement gathering process, allowing managers to focus expertise on truly impactful project drivers. However, this study possesses notable limitations that must be acknowledged. The validation is restricted solely to the NASA93 dataset. While NASA93 is a widely accepted benchmark, its projects predominantly represent historical, large-scale aerospace and government software developments. Modern software projects (e.g., Agile, web, mobile applications) exhibit vastly different characteristics [4], [23]. Consequently, while the simultaneous optimization framework is highly effective here, the conclusion regarding WOA-SVR's superiority must be carefully bounded to this experimental scenario. The model's ability to generalize across contemporary datasets such as ISBSG, Desharnais, China, or COCOMO81 remains an open question and is the primary focus for future work.

IV. CONCLUSION

This study addressed the challenge of improving software effort estimation by developing a hybrid Whale Optimization Algorithm and Support Vector Regression (WOA-SVR) framework. Unlike conventional methods that treat feature selection and parameter tuning as separate sequential steps, our proposed method bridges this gap by

optimizing both components simultaneously within a single WOA search process.

The results from the NASA93 dataset validate the efficacy of this integrated approach. Through a comprehensive ablation study, we empirically demonstrated that simultaneous optimization yields significantly lower errors and higher stability (Mean ± SD) compared to optimizing parameters alone or features alone. By collaboratively evolving the feature subset and SVR hyperparameters, the model successfully identified and removed 7 redundant attributes (reducing dimensions from 22 to 15), which mitigated overfitting and led to a more generalized predictor. Quantitatively, the proposed WOA-SVR outperformed Standard SVR and state-of-the-art metaheuristic baselines, including PSO-SVR, GA-SVR, and GWO-SVR, achieving an MMRE of 21.32% ± 3.25% and a Pred (25) of 64.52% ± 3.90%. Furthermore, the inclusion of Cliff's Delta effect size alongside the Wilcoxon Signed-Rank Test confirmed that these performance gains are not merely statistically significant, but possess a large practical magnitude.

From a practical perspective, reducing the required COCOMO II inputs from 22 to 15 features simplifies the requirement-gathering phase for project managers without sacrificing accuracy. However, in accordance with the limitations of this study, the conclusion regarding WOA-SVR's superiority must be strictly bounded to the NASA93 experimental scenario. Because NASA93 predominantly represents historical, large-scale projects, the model's ability to generalize across contemporary software development paradigms remains untested. Therefore, future work is strictly required to validate this simultaneous optimization framework using diverse and modern benchmark datasets, such as ISBSG, Desharnais, or China datasets, to fully establish its robustness in current software engineering practices.

REFERENCES

[1] M. Rahman, H. Sarwar, A. Kader, T. Goncalves, and T. T. Tin, "Review and Empirical Analysis of Machine Learning-Based Software Effort Estimation," *IEEE Access*, vol. 12, pp. 85661–85680, 2024, doi: 10.1109/ACCESS.2024.3404879.

[2] M. Azzeh, A. B. Nassif, and I. B. Attili, "Predicting Software Effort from Use Case Points: A Systematic Review," *Sci. Comput. Program.*, Apr. 2021, doi: 10.1016/j.scico.2020.102596.

[3] A. Ali and C. Gravino, "Improving software effort estimation using bio-inspired algorithms to select relevant features: An empirical study," *Sci. Comput. Program.*, vol. 205, May 2021, doi: 10.1016/j.scico.2021.102621.

[4] 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.

[5] Y. Mahmood, N. Kama, A. Azmi, A. S. Khan, and M. Ali, "Software effort estimation accuracy prediction of machine learning techniques: A systematic performance evaluation," *Softw. Pract. Exp.*, vol. 52, no. 1, pp. 39–65, Jan. 2022, doi: 10.1002/spe.3009.

[6] J. Li, S. Sun, L. Xie, C. Zhu, and D. He, "Multi-kernel support vector regression with improved moth-flame optimization algorithm for software effort estimation," *Sci. Rep.*, vol. 14, no. 1, Dec. 2024, doi: 10.1038/s41598-024-67197-1.

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

[8] P. V. Terlapu, K. K. Raju, G. Kiran Kumar, G. Jagadeeswara Rao, K. Kavitha, and S. Samreen, "Improved Software Effort Estimation Through Machine Learning: Challenges, Applications, and Feature Importance Analysis," *IEEE Access*, vol. 12, pp. 138663–138701, 2024, doi: 10.1109/ACCESS.2024.3457771.

[9] M. Hosni, A. Idri, and A. Abran, "On the value of filter feature selection techniques in homogeneous ensembles effort estimation," *Journal of Software: Evolution and Process*, vol. 33, no. 6, Jun. 2021, doi: 10.1002/smr.2343.

[10] 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.

[11] S. M. Mirjalili, S. M. Mirjalili, A. Lewis, M. Seyedali, M. Seyed Mohammad, and L. Andrew, "Grey Wolf Optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014, doi: https://doi.org/10.1016/j.advengsoft.2013.12.007 Copyright.

[12] A. Idri, I. Abnane, and A. Abran, "Support vector regression-based imputation in analogy-based software development effort estimation," *Journal of Software: Evolution and Process*, vol. 30, no. 12, Dec. 2018, doi: 10.1002/smr.2114.

[13] J. Kennedy, R. Eberhart, and bls gov, "Particle Swarm Optimization."

[14] A. Puspaningrum, M. Mustamiin, F. Herdiyanti, and K. Noviyanto, "Software Effort Coefficient Optimization Using Hybrid Bat Algorithm and Whale Optimization Algorithm," *JURNAL INFOTEL*, vol. 17, no. 1, pp. 122–135, May 2025, doi: 10.20895/infotel.v17i1.1250.

[15] A. G. Priya Varshini, K. Anitha Kumari, and V. Varadarajan, "Estimating software development efforts using a random forest-based stacked ensemble approach," *Electronics (Switzerland)*, vol. 10, no. 10, May 2021, doi: 10.3390/electronics10101195.

[16] A. Tripathi, K. K. Bharti, and M. Ghosh, "A fusion of binary grey wolf optimization algorithm with opposition and weighted positioning for feature selection," *International Journal of Information Technology (Singapore)*, vol. 15, no. 8, pp. 4469–4479, Dec. 2023, doi: 10.1007/s41870-023-01481-7.

[17] S. Chakraborty, A. K. Saha, R. Chakraborty, and M. Saha, "An enhanced whale optimization algorithm for large scale optimization problems," *Knowl. Based. Syst.*, vol. 233, Dec. 2021, doi: 10.1016/j.knosys.2021.107543.

[18] S. S. Gautam and V. Singh, "The state-of-the-art in software development effort estimation," *Journal of Software: Evolution and Process*, vol. 30, no. 12, Dec. 2018, doi: 10.1002/smr.1983.

[19] H. Alsghaier and M. Akour, "Software fault prediction using Whale algorithm with genetics algorithm," *Softw. Pract. Exp.*, vol. 51, no. 5, pp. 1121–1146, May 2021, doi: 10.1002/spe.2941.

[20] D. Novitasari, I. Cholissodin, and W. F. Mahmudy, "Hybridizing PSO with SA for optimizing SVR applied to software effort estimation," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 14, no. 1, pp. 245–253, Mar. 2016, doi: 10.12928/TELKOMNIKA.v14i1.2812.

[21] R. Rizkiana Putri and D. Candra Novitasari, "Hybrid GWO-PSO for Accurate Software Effort Estimation in COCOMO II," *Journal of Artificial Intelligence and Software Engineering*, vol. 5, no. 3, pp. 1186–1192, 2025, doi: 10.30811/jaise.v5i3.7603.

[22] A. L. I. Oliveira, P. L. Braga, R. M. F. Lima, and M. L. Cornélio, "GA-based method for feature selection and parameters optimization for machine learning regression applied to

software effort estimation," *Inf. Softw. Technol.*, vol. 52, no. 11, pp. 1155–1166, 2010, doi: 10.1016/j.infsof.2010.05.009.

[23] D. Rankovic, N. Rankovic, M. Ivanovic, and L. Lazic, "Convergence rate of Artificial Neural Networks for estimation in software development projects," *Inf. Softw. Technol.*, vol. 138, Oct. 2021, doi: 10.1016/j.infsof.2021.106627.