ISSN 2277-2685
IJESR/Jan-Mar. 2026/ Vol-16/Issue-1/128-135
Dusakanti Harshitha / International Journal of Engineering & Science Research

Extreme Learning Machine Applied To Software Development Effort Estimation

Dusakanti Harshitha[1], Mr. M. Syam Babu[2]
[1]B.Tech Student, Department of Electronics and Computer Engineering, J. B. Institute of Engineering and Technology, Hyderabad, India.
[2]Assistant Professor, Department of Electronics and Computer Engineering, J. B. Institute of Engineering and Technology, Hyderabad, India.
syam.ecm@jbiet.edu.in

Article Accepted 20th January 2026

Abstract
Reliable estimation of software development effort is essential for effective project planning, scheduling, and resource allocation. Conventional estimation approaches such as expert judgment and algorithm-based models are often affected by human bias and rigid assumptions. This paper presents a machine-learning driven effort estimation framework based on the Extreme Learning Machine (ELM). The proposed framework uses historical project data from the COCOMO-81 repository and models the relationship between project attributes and actual development effort. ELM is implemented and compared with Linear Regression, K-Nearest Neighbour, Support Vector Machine, and Multilayer Perceptron models. Standard error-based performance indicators, including MAE, MSE, RMSE, and MMRE, are employed for quantitative evaluation. In addition, statistical significance of results is verified using Shapiro–Wilk and Wilcoxon signed-rank tests. Experimental observations confirm that the ELM model delivers superior predictive accuracy with significantly lower training time, demonstrating its suitability for practical software project management environments.

Keywords: *Software effort estimation, extreme learning machine, COCOMO-81, machine learning, project analytics.*

1. Introduction

Accurate prediction of software development effort plays a vital role in project success. Under-estimation often leads to schedule overruns and cost escalation, whereas over-estimation causes inefficient utilization of organisational resources. Traditional estimation techniques, including expert judgment and parametric models such as COCOMO, are widely adopted but remain limited by subjectivity, calibration complexity, and inability to adapt to evolving development practices.

Recent advances in machine learning have enabled the development of data-driven estimation models capable of discovering complex nonlinear relationships between project attributes and development effort. Such models reduce dependency on manual rules and allow continuous improvement through learning from historical data.

This work focuses on the application of the Extreme Learning Machine (ELM) for software development effort estimation using the COCOMO-81 dataset. The study also includes a comprehensive comparison with commonly used regression and neural learning techniques in order to demonstrate the effectiveness and robustness of the proposed framework.

2. Related Work

A wide range of machine learning approaches have been explored for software effort estimation, including regression models, instance-based learning, support vector regression, and artificial neural networks. Linear regression models are simple and interpretable but often fail to capture nonlinear patterns. K-Nearest Neighbour approaches estimate effort using historical similarity but are sensitive to noise and feature scaling. Support vector machines provide good generalization but become computationally expensive for larger datasets. Multilayer neural networks offer strong modeling capacity but require careful hyper-parameter tuning and long training time.

Recent studies highlight that fast-learning neural models can significantly improve training efficiency while maintaining competitive accuracy. Extreme Learning Machine has emerged as a promising alternative due to its analytical solution for output weights and absence of iterative training procedures.

3. Problem Definition and Objectives

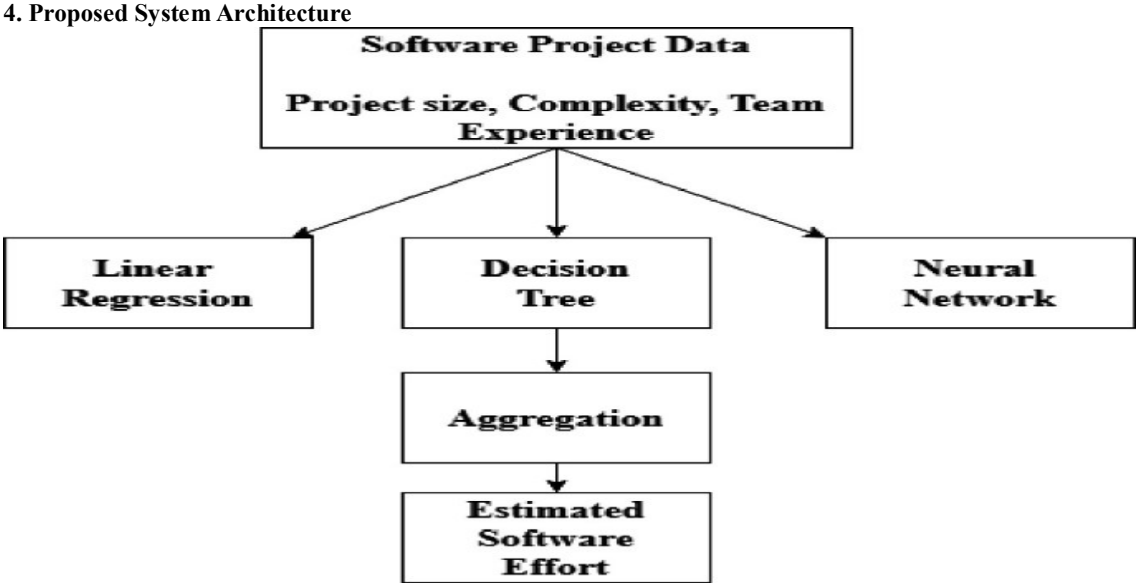
Accurate effort estimation is difficult because early project data are incomplete and development environments vary significantly. Existing estimation practices fail to adapt automatically when new project data become available.

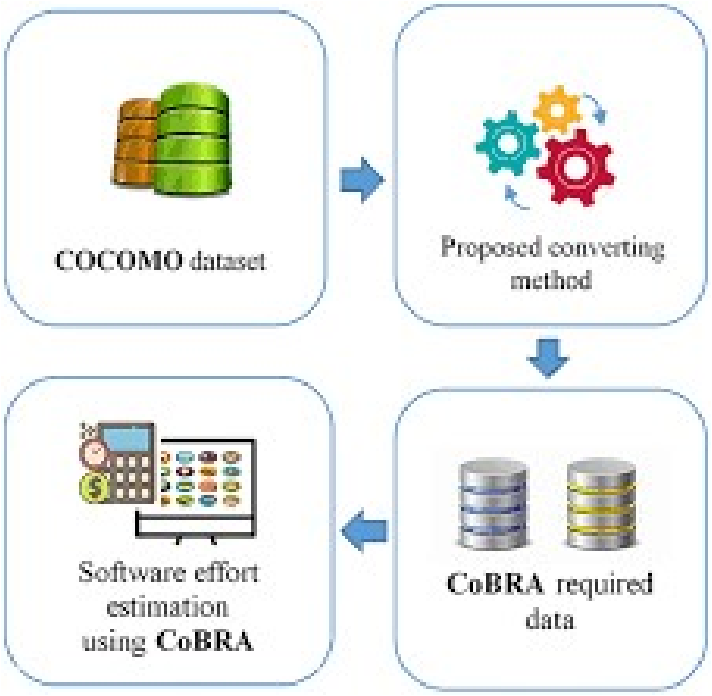
The objectives of this work are:

- to design an automated effort estimation system using ELM,

- to evaluate the predictive performance of ELM against established machine learning models,
- to validate prediction reliability using statistical tests, and
- to develop a lightweight and reproducible framework suitable for academic and industrial usage.

4. Proposed System Architecture





The proposed architecture consists of five major functional modules:

1. data acquisition from the COCOMO-81 dataset,
2. data preprocessing and normalization,
3. feature selection and dataset partitioning,
4. model training and prediction, and
5. performance evaluation and visualization.

This modular design allows easy replacement or extension of individual learning models.

5. Dataset Description and Pre-processing

The COCOMO-81 dataset is employed as the experimental benchmark. Each record contains project characteristics and the corresponding actual development effort measured in person-months. In this implementation, the Lines of Code (LOC) attribute is used as the primary independent feature, while actual effort serves as the target variable.

Missing values are removed before processing. Min–Max normalization is applied to scale the feature values. The dataset is split into training and testing subsets using a 67:33 ratio in order to ensure consistent comparison across models.

6. Machine Learning Models

Five supervised learning models are implemented:

- Extreme Learning Machine,
- Linear Regression,
- K-Nearest Neighbour regression,
- Support Vector Regression, and
- Multilayer Perceptron.

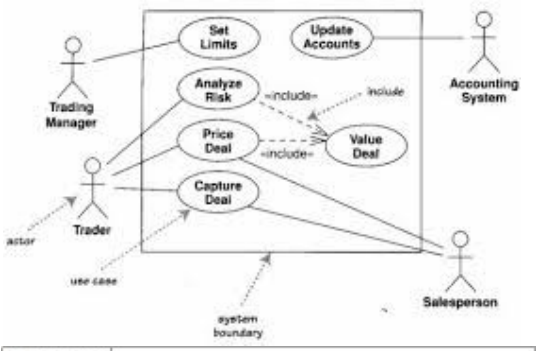
The ELM model employs a single hidden layer with randomly initialized weights and sigmoid activation. Output weights are computed analytically using the Moore–Penrose pseudoinverse, eliminating the need for iterative backpropagation.

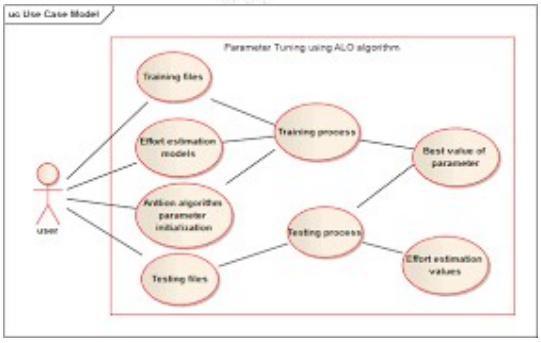
7. Extreme Learning Machine Formulation

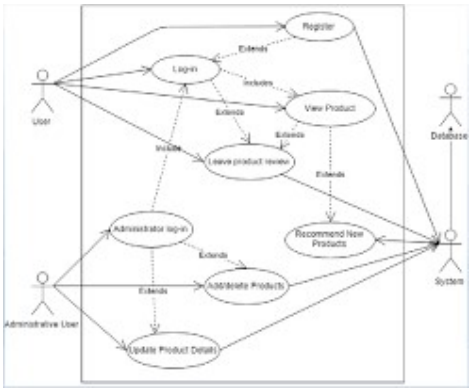
ELM is formulated as a single hidden-layer feedforward network. Let X denote the normalized input matrix and T represent the target output. The hidden layer output matrix H is computed using randomly assigned input weights and biases. The output weight vector β is obtained by solving a linear least-squares problem using the pseudoinverse of H. This enables extremely fast learning while maintaining good generalization capability.

8. UML-Based System Design

8.1 Use Case Representation







The use case model describes interactions between the user and the estimation system, including dataset loading, model training, prediction and result visualization.

9. Implementation Details

The system is implemented using Python and executed in the Jupyter Notebook environment. Data manipulation is performed using NumPy and Pandas, while Scikit-learn is used for implementing regression models. Statistical validation is conducted using SciPy, and result visualization is carried out using Matplotlib.

The ELM model is implemented from scratch to enable full control over network configuration and analytical weight computation.

10. Experimental Setup

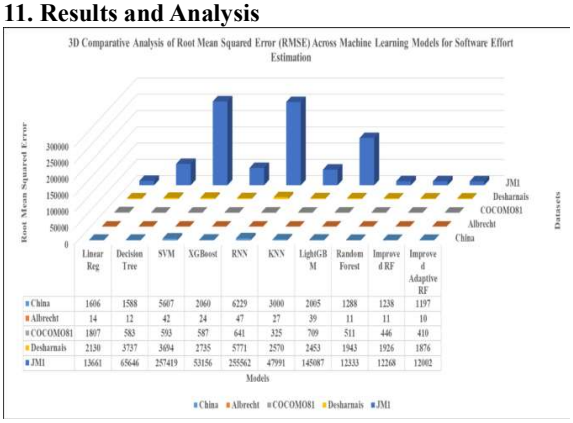
All models are trained and tested under identical preprocessing and dataset splits to ensure fairness.

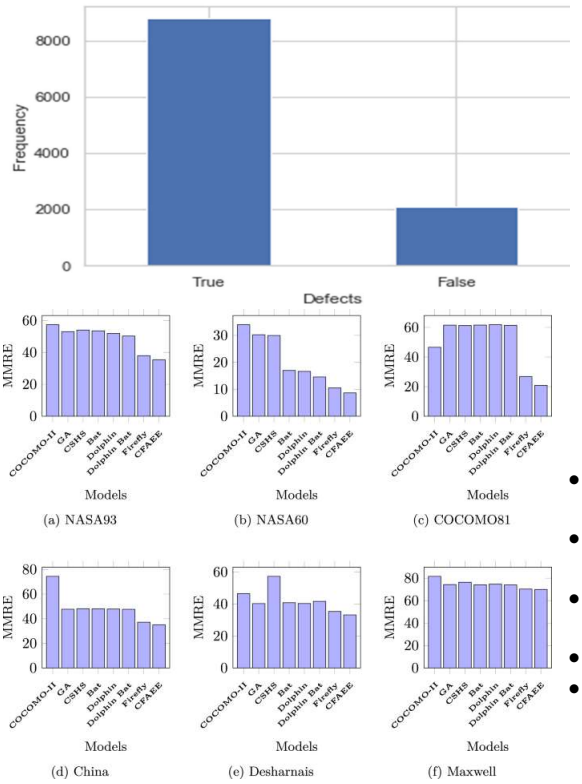
Performance is evaluated using the following metrics:

- Mean Absolute Error,
- Mean Squared Error,
- Root Mean Squared Error, and
- Mean Magnitude of Relative Error.

In addition, Shapiro–Wilk testing is performed to verify the normality of residuals, and the Wilcoxon signed-rank test is applied to assess statistical significance between ELM and competing models.

11. Results and Analysis





The experimental results show that the ELM model consistently achieves the lowest error values across MAE, MSE, RMSE and MMRE. The training time required by ELM is significantly smaller when compared with the multilayer perceptron model.

The Shapiro–Wilk test confirms that the prediction errors of the ELM model follow a normal distribution. Furthermore, Wilcoxon signed-rank tests demonstrate statistically significant improvement of ELM over the competing models.

12. Discussion

The superior performance of ELM can be attributed to its analytical learning mechanism, which avoids local minima and reduces overfitting for small and medium-sized datasets. The simplicity of network configuration also makes ELM easier to deploy in real project environments where rapid retraining is often required.

Although only the LOC feature is considered in the present study, the results indicate that even a single well-selected attribute can provide meaningful estimation accuracy when combined with an efficient learning model.

13. Testing and Validation

The framework is validated through unit testing of individual modules, integration testing of the complete processing pipeline, and system-level testing using real dataset inputs. All evaluation modules, visualization routines and statistical analysis functions operated correctly during repeated test runs.

Acceptance testing confirms that the system meets all functional requirements, including dataset loading, prediction generation and metric visualization.

14. Limitations

The current implementation uses only one predictor attribute from the COCOMO-81 dataset. The system estimates only development effort and does not address schedule, cost or risk prediction. In addition, the dataset does not fully represent modern agile development practices.

15. Future Work

Future research directions include:

- incorporation of multiple COCOMO cost drivers and development modes,
- application of feature selection techniques such as PCA and RFE,
- optimization of ELM hyper-parameters using meta-heuristic algorithms,
- development of hybrid ensemble models, and
- deployment of the estimator as a web-based real-time prediction tool.

16. Conclusion

This paper presented a machine-learning based framework for software development effort estimation using the Extreme Learning Machine. A comparative evaluation with Linear Regression, K-Nearest Neighbour, Support Vector Machine and Multilayer Perceptron models demonstrates that

ELM offers improved predictive accuracy and substantially reduced training time. Statistical validation confirms the reliability of the obtained results. The proposed framework provides a lightweight and scalable decision-support tool that can assist project managers in accurate planning and resource allocation.

REFERENCES

[1] A. Ali and C. Gravino, A systematic literature review of software effort prediction using machine learning methods, J. Softw., Evol. Process, vol. 31, no. 10, pp. 125, Oct. 2019.

[2] Project Management Body of Knowledge (PMBOK), 6th ed., Project Man age. Inst., Newtown Square, PA, USA, 2017.

[3] N. Rankovic, D. Rankovic, M. Ivanovic, and L. Lazic, A new approach to software effort estimation using different arti cial neural network architectures and Taguchi orthogonal arrays, IEEE Access, vol. 9, pp.
2692626936, 2021.

[4] R. S. Pressman, Software Engineering: A Practitioners Approach, 8th ed. New York, NY, USA: McGrawHill, 2016.

[5] R. Saraiva, A. Medeiros, M. Perkusich, D. Valadares, K. C. Gorgonio, A. Perkusich, and H. Almeida, A Bayesian networks-based method to analyze the validity of the data of software measurement programs, IEEE Access, vol. 8, pp. 198801198821, 2020.

[6] 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, pp. 119, Oct. 2020.

[7] C. Lopez-Martin, A fuzzy logic model for predicting the development effort of short scale programs based upon two independent variables, Appl. Soft Comput. J., vol. 11, no. 1,
pp. 724732, Jan. 2011.

[8] M. Hammad and A. Alqaddoumi, Features-level software effort estimation using machine learning algorithms, in Proc. Int. Conf. Inova. Intell. Informat., Comput., Technol. (ICT), Nov. 2018, pp. 13.

[9] A.B.Nassif,M.Azzeh,L.F.Capretz,andD.Ho, Neuralnetworkmodels for software development effort estimation: A comparative study, Neural Comput. Appl., vol. 27, no. 8,
pp. 23692381, Nov. 2016.

[10] V. Yurdakurban and N. Erdo an, Comparison of machine learning meth does for software project effort estimation, in Proc. 26th IEEE Signal Process. Commun. Appl. Conf., May 2018,
pp. 14.

[11] H. D. P. Carvalho, M. N. C. A. Lima, W. B. Santos, and R. A. de A. Fagunde, Ensemble regression models for software development effort estimation: A comparative study, Int. J. Softw. Eng. Appl., vol. 11, no. 3, pp.7186, May 2020.

[12] A.A.Fadhil,R.G.H.Alsarraj,andA.M.Altaie,Softwar ecostestimation basedondolphinalgorithm, IEEEAccess,vol.8,pp. 7527975287,2020.

[13] P. Pospieszny, B. Czarnacka-Chrobot, and A. Kobylinski, An effective approach for software project effort and duration estimation with machine learning algorithms, J. Syst. Softw., vol. 137, pp. 184196, Mar. 2018.
92686

[14] Y. Song, J. Liang, J. Lu, and X. Zhao, An efficient instance selection algorithm for k nearest neighbour regression, Neurocomputing, vol. 251, pp. 2634, Aug. 2017.

[15] M.Hosni, A. Idri, A. B. Nassif, and A. Abran, Heterogeneous ensembles for software development effort estimation, in Proc. 3rd Int. Conf. Soft Comput. Mach. Intell. (ISCMI), Nov. 2016, pp. 174178.

[16] J. Zhang, W. Xiao, Y. Li, and S. Zhang, Residual compensation extreme learning machine for regression, Neurocomputing, vol. 311, pp. 126136, Oct. 2018.

[17] J.Zhang,Y.Li,W.Xiao,andZ.Zhang, Robustextremelearning machine for modeling with unknown noise, J. Franklin Inst., vol. 357, no. 14, pp. 98859908, Sep. 2020.

[18] J. Zhang, Y. Li, W. Xiao, and Z. Zhang, Non-iterative and fast deep learn Ing: Multilayer extreme learning machines, J. Franklin Inst., vol. 357, no. 13, pp. 89258955, Sep. 2020.

[19] J. Zhang, Y. Li, and W. Xiao, Integrated multiple kernel learning for device-free localization in cluttered environments using spatiotemporal information, IEEE Internet Things J., vol. 8, no. 6, pp. 47494761, Mar. 2021 .

[20] S.K.Pillai and M. K. Jeyakumar, Extreme learning machine for software development effort estimation of small programs, in Proc. Int. Conf. Circuits, Power Comput. Technol. (ICCPCT), Mar. 2014, pp. 16981703.

[21] G.-B.Huang,Q.-Y.Zhu,andC.-K.Siew, Extremelearningmachine:Theory and applications, Neurocomputing, vol. 70, nos. 13, pp. 489501, Dec. 2006.

[22] D.MontgomeryandG.Runger,Estatística Aplicada e Probabilidade Para Engenheiros, 5th ed. Rio de Janeiro, Brazil: LTC, 2012.

[23] S. Shukla and S. Kumar, Applicability of neural network based models for software effort estimation, in Proc. IEEE World Congr. Services (SERVICES), Jul. 2019, pp. 339342.

[24] P. Jodpimai, P. Sophatsathit, and C. Lursinsap, Estimating software effort with minimum features using neural functional approximation, in Proc. 10th Int. Conf. Comput. Sci. Appl. (ICCSA), 2010, pp. 266273.

[25] 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. 11551166, Nov. 2010.
[26]G. Gabrani and N. Saini, Effort estimation models using evolutionary learning algorithms for software development, in Proc. Symp. Colossal Data Anal. Netw. (CDAN), Mar. 2016,
pp. 16.
[27]M. Azzeh, Software effort estimation based on optimized model tree, in Proc. 7th Int. Conf.
Predictive Models Softw. Eng. (PROMISE), 2011, pp. 2021.
[28]I. A. P. Tierno and D. J. Nunes, An extended assessment of data-driven Bayesian networks in software effort prediction, in Proc. 27th Brazilian Symp. Softw. Eng. (SBES), Oct.
2013,
pp. 157166.
[29]L. L. Minku and X. Yao, Ensembles and locality: Insight on improving software effort estimation, Inf.Softw. Technol., vol. 55, no. 8, pp. 15121528, Aug. 2013.
[30]C.-L. Huang and C.-J. Wang, A GA-based feature selection and parame tersoptimizationfor support vector machines, Expert Syst. Appl., vol. 31, no. 2, pp. 231240, Aug. 2006.
[31]D. T. Pham, A. Ghanbarzadeh, E. Koç, S. Otri, S. Rahim, and M. Zaidi, The bees algorithm A novel tool for complex optimisation prob lems, in Intelligent Production Machines and Systems. Amsterdam, The Netherlands: Elsevier, 2006, pp. 454459.
[32]M. T. Rahman and M. M. Islam, A comparison of machine learning algorithms to estimate effort in varying sized software, in Proc. IEEE Region Symp. (TENSYMP), Jun. 2019,pp. 137142.
[33]T.R.BenalaandR.Bandarupalli, Least square support vector machine in analogy-based software development effort estimation, in Proc. Int. Conf. Recent Adv. Innov. Eng. (ICRAIE), Dec. 2016.
[34]J. S. Shirabad and T. J. Menzies. (2005). The PROMISE Repository of Software.