Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Volume 13, Issue 12, December 2024

Predicting Software Development Effort using Machine Learning and Deep Learning

Rohan Bhatia
Vellore Institute of Technology, Vellore

***Abstract* - Accurate software development effort estimation is vital for effective project management, impacting timelines, budgets, and resources. Traditional methods often fall short in capturing the complexities of modern projects. This study explores the use of machine learning (ML) and deep learning (DL) models, including Random Forests and LSTM networks, to improve prediction accuracy. By leveraging large datasets and advanced algorithms, these models reveal complex patterns in project data. Despite challenges like data quality and model interpretability, advancements in hybrid models and explainable AI highlight the potential of ML and DL to transform effort estimation and enhance project outcomes.**

***Keywords*—Software effort estimation, machine learning, deep learning, Random Forest, LSTM, project management, prediction models, hybrid models, explainable AI.**

I. INTRODUCTION

"Prediction is not just about anticipating the future; it's about shaping it with the right tools and insights." An essential component of software project planning and management, software development effort estimation has a direct impact on timelines, budgets, and resource allocation. Accurate effort evaluation is essential in reducing risks including cost overruns, delivery delays, and inefficient resource management. However, conventional estimation techniques like Function Point Analysis and the Constructive Cost Model (COCOMO) frequently rely significantly on historical data, expert judgment, and oversimplified statistical models. Although helpful, these methods usually fall short of capturing the intricacies and nonlinear interactions present in contemporary software development projects. The necessity for more sophisticated, data-driven methods that can manage the dynamic and complex nature of software engineering is highlighted by this constraint.

II. RELATED WORK

In a number of fields, including software effort estimation, machine learning (ML) and deep learning (DL) have become revolutionary techniques. ML models that forecast development effort more accurately than traditional models include regression analysis, decision trees, support vector machines (SVM), and ensemble techniques like random forests (RF) and stochastic gradient boosting (SGB). By spotting complex patterns and connections in big datasets, deep learning techniques—in particular, artificial neural networks (ANN) and long short-term memory (LSTM) networks—have significantly improved prediction accuracy. These models offer significant advantages by learning complex relationships and adapting to project-specific characteristics, making them ideal for estimating effort in large-scale and complex software projects.

The growing interest in leveraging ML and DL for software effort estimation has led to extensive research and experimentation. Several studies have explored the comparative performance of ML and DL models using datasets such as COCOMO II, ISBSG, and NASA's software project benchmarks. These datasets offer a strong basis for training and testing prediction models because they include project-specific characteristics including lines of code (LOC), function points (FP), development time, and team size. Research continuously shows that ML and DL models perform better than conventional techniques, with random forests and neural networks often showing the highest accuracy and dependability across a range of performance metrics, such as R-squared (R^2), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE).

III. METHODOLOGY

Choosing the best model for a project is one of the main difficulties in effort estimating. Because of their interpretability and capacity to manage smaller information efficiently, machine learning models like decision trees and random forests are beneficial. These models partition the data and generate predictions through a series of logical decisions, making them well-suited for straightforward projects. Conversely, deep learning models like LSTM and ANN excel in capturing nonlinear relationships and temporal dependencies, making them particularly effective for large and complex projects where traditional models may struggle. However, deep learning models require larger datasets and higher computational power, which can be a limiting factor for smaller software firms or projects with limited historical data.

Hybrid approaches that combine multiple ML and DL models have also shown promise in enhancing effort estimation accuracy. For example, integrating feature selection techniques with ANN or combining genetic algorithms (GA) with deep neural networks (DNN) can improve prediction accuracy by optimizing model inputs and reducing overfitting. These hybrid methods leverage the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Volume 13, Issue 12, December 2024

strengths of various algorithms, resulting in models that are both accurate and robust. Ensemble methods, such as stacking and boosting, further enhance model performance by aggregating predictions from multiple base models, mitigating individual model biases, and increasing overall prediction reliability.

Feature engineering plays a critical role in the effectiveness of ML and DL models for software effort estimation. Key features such as project size, team experience, development environment, and complexity metrics significantly influence prediction accuracy. By ensuring that models concentrate on the most pertinent characteristics, effective feature selection lowers noise and enhances generalization. More precise and comprehensible models have been produced by refining feature sets using sophisticated approaches like principal component analysis (PCA) and recursive feature elimination (RFE). The predictive power of ML and DL models is further increased by the addition of domain-specific elements like risk factors, software design, and project management techniques.

IV. RESULTS AND DISCUSSION

To guarantee the dependability of effort estimating models, model evaluation and validation are crucial. To evaluate model performance and avoid overfitting, cross-validation ways like k-fold and leave-one-out cross-validation are often used. Performance measures that highlight the accuracy and consistency of forecasts include RMSE, MAE, and R2. Research continuously shows how well LSTM networks and random forests function, with LSTM networks performing well in large-scale projects and random forests providing strong performance on smaller datasets. The use of confusion matrices, precision, recall, and F1 scores further aids in evaluating model effectiveness, particularly for classification-based effort estimation tasks.

V. CHALLENGES AND FUTURE DIRECTIONS

Despite the advancements in ML and DL for effort estimation, several challenges remain. Data quality and availability continue to be significant constraints, as effort estimation models heavily rely on comprehensive and high-quality datasets. Incomplete or inconsistent data might lower model reliability and produce erroneous predictions. Furthermore, project managers and other stakeholders that need clear and understandable predictions find it difficult to deal with the interpretability of deep learning models. To tackle this problem, methods like SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations) have been created, which offer insights into model choices and boost confidence in forecasted results.

Expanding the range of datasets, integrating real-time data, and investigating the integration of ML and DL approaches with agile and DevOps methodologies are anticipated to be the main areas of future study in software effort estimate. Enhancing cross-project effort estimation may be possible through the use of transfer learning, a technique that involves adapting models developed on one project to another. Furthermore, there is growing interest in the use of reinforcement learning (RL) in effort estimates, where models continuously learn and adjust in response to feedback and project outcomes. This strategy might result in effort estimating systems that are more responsive and dynamic, further bridging the gap between theory and practice.

VI. CONCLUSION

The results of this study highlight how ML and DL may revolutionize the assessment of software development effort. Software development teams can improve project planning, cut expenses, and increase overall project success rates by using data-driven models to make more accurate forecasts. Even though there are still obstacles to overcome, the ongoing development of ML and DL technologies holds the potential to completely transform effort estimating procedures and open the door to more dependable and effective software project management.

REFERENCES

[1] Predicting software effort estimation using machine learning techniques. (2018, July 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/8486222

[2] Varshini, A. G. P., Kumari, K. A., Janani, D., & Soundariya, S. (2021). Comparative analysis of Machine learning and Deep learning algorithms for Software Effort Estimation. Journal of Physics Conference Series, 1767(1), 012019. https://doi.org/10.1088/1742-6596/1767/1/012019

[3] Khan, M. A., Elmitwally, N. S., Abbas, S., Aftab, S., Ahmad, M., Fayaz, M., & Khan, F. (2022). Software Defect Prediction Using Artificial Neural Networks: A Systematic Literature Review. Scientific Programming, 2022, 1–10. https://doi.org/10.1155/2022/2117339

[4] Jadhav, A., & Shandilya, S. K. (2023). Reliable machine learning models for estimating effective software development efforts: A comparative analysis. Journal of Engineering Research, 11(4), 362–376. https://doi.org/reliable-machine-learning-models