

International Journal of Engineering Research and Science & Technology



ISSN : 2319-5991

www.ijerst.com

Email: editor@ijerst.com or editor.ijerst@gmail.com

REVIEW AND EMPIRICAL ANALYSIS OF MACHINE LEARNING-BASED SOFTWARE EFFORT ESTIMATION

¹ Priyanka Medavarapu, ² Pagidipalli Madhu, ³ Ajay Kumar Dharmaram, ⁴ Md Wazahetullah

^{1,2,3} Assistant Professors, Department of Computer Science and Engineering, Kasireddy Narayanreddy College Of Engineering And Research, Abdullapur (V), Abdullapurmet(M), Rangareddy (D), Hyderabad - 501 505

⁴ student, Department of Computer Science and Engineering, Kasireddy Narayanreddy College Of Engineering And Research, Abdullapur (V), Abdullapurmet(M), Rangareddy (D), Hyderabad - 501 505

ABSTRACT

Software effort estimation is a necessary component of software development projects that belong to industrial software systems and digital transformation initiatives. Digital transformation refers to the process of integrating digital technology into various components of a company or organization in order to improve operations, procedures, customer experiences, and overall performance. Industrial software systems are trained software packages designed for use in industrial and manufacturing processes. The paper deals with the machine learning based effort estimation in order to create an effective and robust model for predicting effort. The paper proposes an Omni-Ensemble Learning (OEL) approach, which is a combination of static ensemble selection along with genetic algorithm and dynamic ensemble selection. The paper identifies the impact of software effort estimation in industrial software system, and works on the these attributes to implement a robust ensemble model. The proposed Omni-Ensemble Selection (OES) provides better overall performance (in terms of evaluation metrics) and on comparing with multiple machine learning models over Finnish and Maxwell datasets.

1. INTRODUCTION

The implementation of digital transformation [1] in industries is made possible, in large part, by the use of industrial software systems. The term “digital transformation” refers to the practice of adopting and

integrating digital technology [2] into many elements of corporate operations, processes, and models in order to promote innovation, enhance efficiency, improve business performance and development and obtain a

competitive edge. Industrial software systems are specialized software programs that have been built for industrial and manufacturing environments. These applications offer the foundation for digitizing and automating essential activities in a variety of industries [3], including but not limited to corporate [4], manufacturing, logistics, energy, and transportation.

The following are some of the ways that industrial software systems make digital transformation possible [5]: Automation of processes; Industrial software systems make it possible to automate a wide variety of business processes, including production planning and scheduling, inventory management, quality control, and supply chain management. Data Management and Analytics; it makes it easier to integrate and connect a variety of different devices, systems, and procedures inside an industrial setting. They make it possible for the many components of an industrial ecosystem, such as sensors, machines, control systems, and enterprise resource planning (ERP) systems, to communicate with one another and share data and information with one another [6].

Remote Monitoring and Control, as well as Predictive Maintenance, are a couple of the ways that equipment failures and downtime can be anticipated and avoided. These systems are able to spot trends and

abnormalities that suggest probable failures by analyzing historical data and monitoring real-time data from industrial assets. This enables proactive maintenance and minimizes unplanned downtime. The production of digital twins, which are digital replicas of physical assets, processes, or systems, is made possible by industrial software systems. The ability to simulate, model, and conduct analysis on real-world scenarios is made possible by digital twins. This assists in the optimization of design, as well as predictive maintenance and performance optimization.

Software Effort Estimation is an essential component of software development projects that needs to be done, and is connected to industrial software systems and with digital transformation initiatives as shown in Venn diagram Figure 1. The process of integrating digital technology into various elements of a company or organization in order to improve its operations, procedures, customer experiences, and overall performance is referred to as digital transformation. It is likely that software effort estimation will have a substantial impact on digital transformation, which may have repercussions in a number of important areas, including project planning and budgeting, resource management, project execution and

delivery, stakeholder management, and return on investment (ROI) [7].

A. SOFTWARE EFFORT ESTIMATION

Estimating the amount of effort required to produce software is an essential part of product engineering, which is the process of developing software products, applications, or solutions in response to particular consumer demands. An accurate prediction of the amount of work that needs to be done is essential for the completion of successful product engineering projects. This is because accurate estimates aid with planning, budgeting, resource allocation, quality and performance, customer satisfaction, information exchange, collaborative decision-making, and the overall management of the project. Traditional industries are being transformed into digitally enabled, data-driven, and agile operations by industrial software systems [8], which is paving the way for the future of Industry 4.0 [9]. This transformation is made possible by using the power of digital technologies.

Software Effort Estimation (SEE) is the process of estimating the amount of work, resources, and time necessary to finish a software development project [10]. To arrive at an estimate of the amount of work required to construct the software system, it is necessary to first evaluate the

scale [11], intricacy, and breadth of the project, in addition to the resources that are now at one's disposal.

Estimating the amount of work that needs to be done is an essential part of the software development process since it forms the foundation for project planning, the distribution of resources, and financial planning [12]. An accurate prediction of the amount of effort required helps businesses improve the efficiency of their software development project planning and management, which in turn increases the likelihood that the projects will be finished on time, without going over budget, and to the standard of excellence that was intended.

Estimating the amount of work required to develop software can be done using a variety of methods and strategies, such as expert judgement, analogous estimating, parametric estimation, three-point estimation, bottom-up estimation, and tool-based estimation, just to mention a few [13]. It enables businesses to manage software development projects more effectively, hence lowering risks and increasing the likelihood that the projects will be successful. It is an ongoing process that requires careful evaluation of aspects that are unique to the project, expertise, and the application of appropriate estimation

techniques in order to arrive at estimates of effort that are trustworthy and practical.

B. RESEARCH ON SEE There has been a significant amount of research conducted on software effort estimate, the goals of which have included the development of novel estimation methods as well as the evaluation of the accuracy of existing estimation methods. Function Point Analysis, Use Case Points, and the COCOMO model are three common approaches that are used to estimate the amount of effort required to develop software.

In recent years, intellectuals have put forth approaches for obtaining high effort predictability. The community has recently grown more intrigued by the use of machine learning approaches for estimating development effort. No single ML model, nevertheless, is thought to be efficient for all the software effort datasets. Finding a model that is effective for all software datasets and provides the highest performance in terms of accurate estimation is therefore always a challenging task.

Artificial Neural Networks (ANNs), Support Vector Machines (SVMs), Random Forest (RF), K-Nearest neighbors (KNN) and many more, types of machine learning techniques [14], have been applied to the estimation of the amount of effort required to

develop software, with some encouraging results.

Estimating the time and effort required to develop software continues to be a difficult undertaking that is frequently plagued by considerable estimation mistakes, despite the extensive research that has been conducted in this field. The accuracy of effort estimates can be impacted by a variety of factors, including changes in the requirements of the project, the dynamics of the team, and technological developments. As a consequence of this, a large number of researchers are always looking into new and improved techniques for the assessment of software development efforts.

C. OUR CONTRIBUTION In order to help academics and corporate companies on their way to a successful shift from traditional ways of effort estimation to Industry 4.0 digital transformation, this article offers an integrative business process management paradigm. It contains the crucial elements that are frequently ignored when integrating Industry 4.0 and offers a coordinated strategy to deal with them. Making an implementation plan requires a thorough understanding of enabling technologies their impact on digitization and design principles.

- The paper identifies the importance and impact of software effort estimation in the process of digital transformation.
- The paper explored the datasets in the field of SEE which are associated with the digitization of the industrial software system.
- The machine learning and ensemble learning models are applied on the dataset, in order to obtain higher accuracy in terms of effort estimates.
- Later, the paper explains the impact of effective effort estimation in product engineering and industrial software systems.

II. EXISTING SYSTEM

Ali et al. [15] conducted a study to reveal the effort estimation problems which are related and have a direct impact on software maintenance. Also identified problems that have a direct impact on software maintenance in order to provide a guideline for researchers to counter these problems. The study utilised multiple models extensively used for software maintenance purpose such as COCOMO, Function Point, Act model, package model etc. Tanveer et al. [8] examined and comprehended the estimating process in relation to its accuracy in the context of agile software development from the viewpoint of agile development teams, with the end goal of enhancing the efficacy of the effort estimation process.

Three agile development teams at SAP SE (a German multinational software firm) were studied through case study research. The authors findings stated, that the complexity and effect of modifications to the underlying system, as well as the developer's expertise and experience, affect both the amount and accuracy of estimation.

Tanveer et al. [16] studied the impact analysis for selecting a effort estimation strategy. According to authors, during the estimating process, the expert is assisted by a tool-based decision support system that has a two-way conversation with the expert and gives the expert with the necessary facts. The authors assessed the practicality of incorporating change effect analysis into expert judgment-based estimate for business purposes, with the help of students and practioners.

Azzeh et al. [17] conducted a survey on use case point based approach to estimate effort. After examining 75 research articles, the authors stated that the researchers have become more curious over the past two decades about the feasibility of estimating software effort using the use case point approach, and concludes that this method shows potential for more accurate initial effort prediction.

Usman et al. [18] conducted an industrial case study and identified how effort is estimated in large-scale distributed agile projects and what factors affect the precision of those estimates. The primary takeaways from this case study are; a two-step estimating approach, including re-estimation during the analysis stage, increases the precision of the effort estimations; less mature teams incur larger effort overruns; requirements priorities affect the accuracy of the effort estimates and underestimation is the main cause of project failure.

Acharya et al. [19] implemented a framework called Imp to generate encouraging empirical findings on a commercial codebase including over a million lines of C/C++ code. The author stated, that knowing the probable implications of a software change, or change impact analysis, is essential for risk assessment, developer effort estimation, and regression testing. The authors worked on a static change effect analysis methodology for industrial software system.

Polkowski et al. [20] investigated the use of machine learning techniques in software effort estimation, in decision making. The authors stressed on pre-processing data, which is crucial for accurate estimate while

keeping the dataset consistent. Secondly, for an ideal feature set identification, special focus is placed on the attributes chosen and how they influence the estimation.

Marco et al. [21] proposed AdaBoost ensemble learning and random forest (RF), as well as the Bayesian optimization method, to obtain the model's hyperparameters. The SEE model was trained and tested using the PROMISE repository and the ISBSG dataset. The AdaBoost ensemble learning and bayesian optimization-based RF approach, according to the author, outperforms. The AdaBoost-based model also rates the relevance of each feature, making it a viable tool for estimating software work.

Sinha et al. [22] examines existing machine learning methods for estimating software effort, their sphere of application, a method for estimating software costs, and an analysis of those methods. The authors stated, to get over the restrictions of algorithmic models, researchers have looked towards non-algorithmic methodology based on soft computing techniques.

Disadvantages

- The complexity of data: Most of the existing machine learning models must be able to accurately interpret large and

complex datasets to detect Effective Software Effort Estimation.

- Data availability: Most machine learning models require large amounts of data to create accurate predictions. If data is unavailable in sufficient quantities, then model accuracy may suffer.
- Incorrect labeling: The existing machine learning models are only as accurate as the data trained using the input dataset. If the data has been incorrectly labeled, the model cannot make accurate predictions.

III. PROPOSED SYSTEM

We propose a method for choosing a base regression model in an ensemble system that takes into account both its prior success in the specified domain and the reliability of its latest prediction. Because of this, we are able to combine the static and dynamic methods of ensemble selection.

This paper proposes a three-phase methodology in order to extract the best suitable machine learning model for effective effort estimation. The pictorial representation of the three phases is depicted in this system. First phase is data preprocessing, the datasets are imported and normalized using MinMax Scaling technique; This method converts the values of a feature to a scale that falls somewhere

between 0 and 1. In order to accomplish this, first the minimum value of the feature is subtracted from each value, and then those results are divided by the feature's range. Later the relevant feature set is selected based on the impact of SEE on ISS as discussed in this system. The train-test split method is imported in order to split the data into train and test data in the ratio of 70:30 respectively.

In the second phase, the paper first proposes an improved version of Static Ensemble Selection (SES) based on a genetic algorithm (termed SES-GA), which chooses the top classifiers through the collaborative optimization of accuracy and diversity, from the pool of machine learning models. The pool contains diverse machine-learning regression models which are depicted in this system. The best-selected models from the pool based on SES-GA will be the input to the third phase of our approach.

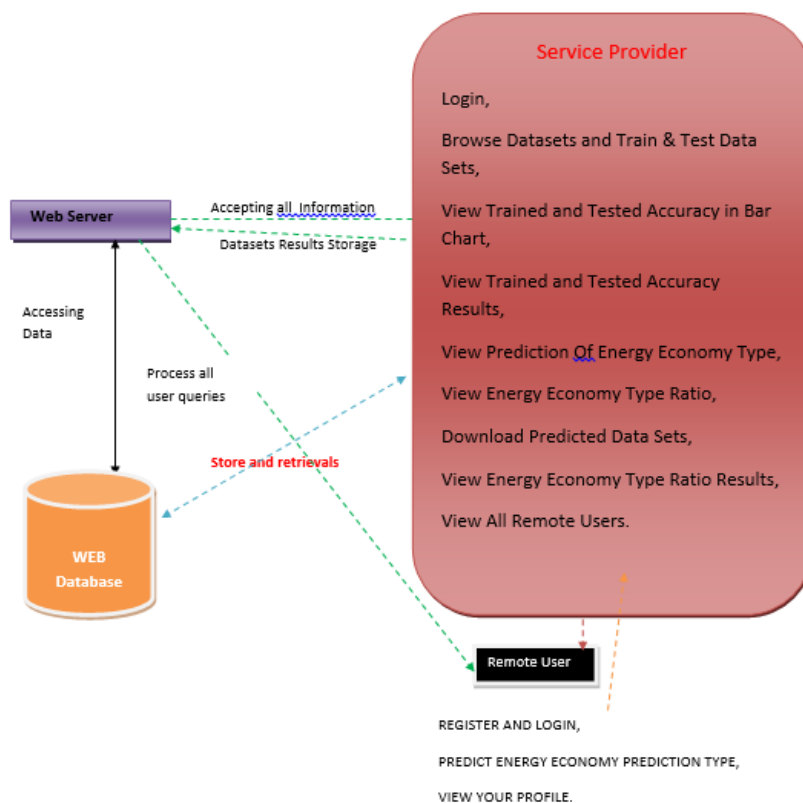
In the third phase, we employ Dynamic Ensemble Selection (DES), which selects dynamically a subset of the classifier from the pool of selected classifiers via GA based on a measure of competence based on random categorization. It involves fitting models from the second phase to the training dataset and selecting the models that are

projected to perform best when predicting a test case. The second strategy makes use of features from both the SES and DES methodologies proposing the Omni-Ensemble selection (OES) approach. The experiment is evaluated based on the evaluation metrics which are discussed in later sections. The statistical analysis of the proposed approach is also performed using Wilcoxon T-Test, which generates the p-value which indicates the probability, that the null hypothesis is true. All the experiment has been performed over Google Colaboratory or Colab, a product from Google Research. Colab makes it possible to use the extensive functionality of several

Python packages for data visualization and analysis through the browser.

Advantages

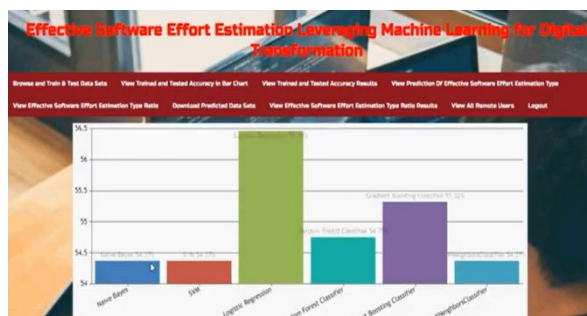
- The paper identifies the importance and impact of software effort estimation in the process of digital transformation.
- The paper explored the datasets in the field of SEE which are associated with the digitization of the industrial software system.
- The machine learning and ensemble learning models are applied on the dataset, in order to obtain higher accuracy in terms of effort estimates.
- Later, the paper explains the impact of effective effort estimation in product engineering and industrial software systems.



IV. MODULES

Service Provider

In this module, the Service Provider has to login by using valid user name and password. After login successful he can do some operations such as Login, Browse and Train & Test Data Sets, View Trained and Tested Accuracy in Bar Chart



View Trained and Tested Accuracy Results



View Prediction Of Effective Software Effort Estimation Type, View Effective Software Effort Estimation Type Ratio,



Download Predicted Data Sets, View Effective Software Effort Estimation Type Ratio Results, View All Remote Users.

View and Authorize Users

In this module, the admin can view the list of users who all registered. In this, the admin can view the user's details such as, user name, email, address and admin authorizes the users.

Remote User

In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like REGISTER AND LOGIN, Predict Effective Software Effort Estimation Type, view your profile.

V.CONCLUSION

Software effort estimation is required for software development projects associated with industrial software systems and initiatives for digital transformation. Digital

transformation is the process of incorporating digital technology into various aspects of a business or organization in order to enhance operations, procedures, consumer experiences, and overall performance. Industrial software systems are software programs that have been instructed for use in industrial and manufacturing processes. The software development industry continues to face difficulties with software effort estimation. Planning, allocating resources, and finishing a project successfully are all affected by how effectively one can estimate how much effort is required. Researchers are looking for ways to incorporate AI and automation technologies into software effort estimation as the software business develops. The purpose of this paper is to develop an effective and robust model for predicting effort based on machine learning.

To conclude, the paper proposes an Omni-Ensemble Learning (OEL) method, which combines static ensemble selection along with genetic algorithm and dynamic ensemble selection. The paper identifies the impact of software effort estimation in industrial software systems and implements a robust ensemble model based on the relevant attributes. On the basis of the impact criterion, we extracted two effort estimation datasets suitable for this ideology. The

proposed Omni-Ensemble Selection (OES) outperforms individual machine learning models over the Finnish and Maxwell datasets in terms of evaluation metrics; SMAPE, MRE, MASE, NSE and COD values. We believe that the development and implementation of Omni-Ensemble Selection (OES) models can enhance the quality of prediction and provide a significant advantage over prior studies.

In future, we intend to apply same model on some other datasets with more impacted features towards digital transformation. Also we will implement an hybrid model for estimating effort, in order to reduce the challenges faced by decision makers in software companies. This paper provides an example of how the technology can be used to automate industrial software systems and society. This paper is an effort to enhance and digitize society, a step towards digital transformation, establish the concept of intelligent software systems, and contribute to developing these technologies.

VI. REFERENCES

- [1] C. Ebert and C. H. C. Duarte, "Digital transformation," *IEEE Softw.*, vol. 35, no. 4, pp. 16–21, Jul. 2018.
- [2] H. Demirkan, J. C. Spohrer, and J. J. Welser, "Digital innovation and strategic transformation," *IT Prof.*, vol. 18, no. 6, pp. 14–18, Nov. 2016.
- [3] D. Ulas, "Digital transformation process and SMEs," *Proc. Comput. Sci.*, vol. 158, pp. 662–671, Jan. 2019.
- [4] D. Coleman, B. Lowther, and P. Oman, "The application of software maintainability models in industrial software systems," *J. Syst. Softw.*, vol. 29, no. 1, pp. 3–16, Apr. 1995.
- [5] M.-H. Delmond, F. Coelho, A. Keravel, and R. Mahl, "How information systems enable digital transformation: A focus on business models and value co-production," HEC Paris, Jouy-en-Josas, France, Res. Paper no. MOSI-2016-1161, 2016.
- [6] D. Bilgeri, F. Wortmann, and E. Fleisch, "How digital transformation affects large manufacturing companies' organization," in *Proc. Int. Conf. Inf. Syst. (ICIS)*, 2017, pp. 1–10.
- [7] E. Gökalp and V. Martinez, "Digital transformation capability maturity model enabling the assessment of industrial manufacturers," *Comput. Ind.*, vol. 132, Nov. 2021, Art. no. 103522.
- [8] B. Tanveer, L. Guzmán, and U. M. Engel, "Understanding and improving effort estimation in agile software development—An industrial case study," in *Proc. IEEE/ACM Int. Conf. Softw. Syst. Processes (ICSSP)*, May 2016, pp. 41–50.

- [9] J. Butt, "A conceptual framework to support digital transformation in manufacturing using an integrated business process management approach," *Designs*, vol. 4, no. 3, p. 17, Jun. 2020.
- [10] N. A. Khan, *Research on Various Software Development Lifecycle Models*. Cham, Switzerland: Springer, 2020.
- [11] 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.
- [12] B. Prakash and V. Viswanathan, "A survey on software estimation techniques in traditional and agile development models," *Indonesian J. Electr. Eng. Comput. Sci.*, vol. 7, no. 3, pp. 867–876, 2017.
- [13] J. S. Pinkster, "Four methods for software effort estimation," Tech. Rep., Apr. 2016. [Online]. Available: <https://ictinstitute.nl/methodsfor-software-effort-estimation/>
- [14] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," *Inf. Softw. Technol.*, vol. 54, no. 1, pp. 41–59, Jan. 2012.
- [15] S. S. Ali, M. Shoaib Zafar, and M. T. Saeed, "Effort estimation problems in software maintenance—A survey," in *Proc. 3rd Int. Conf. Comput., Math. Eng. Technol. (iCoMET)*, Jan. 2020, pp. 1–9.
- [16] B. Tanveer, A. M. Vollmer, S. Braun, and N. B. Ali, "An evaluation of effort estimation supported by change impact analysis in agile software development," *J. Softw., Evol. Process*, vol. 31, no. 5, p. e2165, May 2019.
- [17] M. Azzeh, A. Bou Nassif, and I. B. Attili, "Predicting software effort from use case points: A systematic review," *Sci. Comput. Program.*, vol. 204, Apr. 2021, Art. no. 102596.
- [18] M. Usman, R. Britto, L.-O. Damm, and J. Börstler, "Effort estimation in large-scale software development: An industrial case study," *Inf. Softw. Technol.*, vol. 99, pp. 21–40, Jul. 2018.
- [19] M. Acharya and B. Robinson, "Practical change impact analysis based on static program slicing for industrial software systems," in *Proc. 33rd Int. Conf. Softw. Eng. (ICSE)*, May 2011, pp. 746–755.
- [20] Z. Polkowski, J. Vora, S. Tanwar, S. Tyagi, P. K. Singh, and Y. Singh, "Machine learning-based software effort estimation: An analysis," in *Proc. 11th Int. Conf. Electron., Comput. Artif. Intell. (ECAI)*, Jun. 2019, pp. 1–6.