IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 3 June 2023, accepted 25 June 2023, date of publication 7 July 2023, date of current version 11 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3293432

RESEARCH ARTICLE

Effective Software Effort Estimation Leveraging Machine Learning for Digital Transformation

AKSHAY JADHAV[1], SHISHIR KUMAR SHANDILYA[1], (Senior Member, IEEE), IVAN IZONIN[2], (Senior Member, IEEE), AND MICHAL GREGUS[3], (Member, IEEE)

[1]School of Computing Science and Engineering, VIT Bhopal University, Kothri Kalan, Sehore, Madhya Pradesh 466116, India
[2]Department of Artificial Intelligence, Lviv Polytechnic National University, 79013 Lviv, Ukraine
[3]Department of Information Systems, Comenius University Bratislava, Bratislava, 82005, Slovak Republic

Corresponding author: Ivan Izonin (ivanizonin@gmail.com)

This work was supported by the British Academy's Researchers at Risk Fellowships Program.

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.

INDEX TERMS Digital transformation, industrial software system, software effort estimation, software engineering.

I. 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 specialised software programmes that have been built for industrial and manufacturing environments. These applications offer the foundation for digitising 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 analysing historical data and

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Ali Babar.

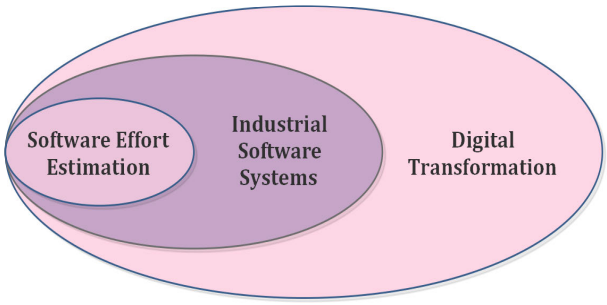

FIGURE 1. Venn Diagram.

monitoring real-time data from industrial assets. This enables proactive maintenance and minimises 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 optimisation of design, as well as predictive maintenance and performance optimisation.

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 organisation 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.

D. ARTICLE ORGANISATION

The rest of the paper is structured in the following order; Section II discusses the literature related to software effort estimation. Impact of effort estimation on industrial software system is elaborated in Section III. Section IV gives the dataset description and machine learning techniques used in the proposed approach. Proposed work is explained in Section V. Section VI discusses the results obtained and its impact on industrial systems. Lastly, the paper is concluded along with the future directions in the field of digital transformation enabled by software effort estimation in Section VII.

II. LITERATURE

The section discusses some related work by the academician and researchers in the field of software effort estimation and its impact on the software companies, industrial software systems and towards digital transformation.

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 II, 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.

Many researchers in the field of software effort estimation have tried to improve the effectiveness of the machine learning models using various feature selection and extraction techniques [23], [24], [25], the use of bio-inspired techniques for optimization and hyper-parameter tunning are also being utilised for implementing energy efficient and optimized models for effort estimation is trending among researchers [26], [27]. Many author in above stated literature have discussed the impact of effort estimation models on the industrial software system. These problems directly impacts the software companies and later it puts negative influence on Industrial software system, which affects digitization. Table 1 provides a description along with the limitations in the work of previous authors.

III. IMPACT OF SEE ON INDUSTRIAL SOFTWARE SYSTEMS

Estimating the amount of work that goes into developing software is an essential component of industry software systems. Industry software systems [19] are complicated and large-scale software applications or systems that are used by organizations for their operations, processes, and management [15], [28]. The following are some of the domains in Figure in which one may observe the role that software development effort estimation plays in industry software systems:

1) Project Planning and Budgeting
During efforts to transform digitally, accurate software effort estimation is essential for efficient project planning and budgeting in order to meet company goals. Accurate estimations provide organisations with assistance in assessing the scope of the task, locating the necessary resources, and properly allocating resources, including time and budget, in order to ensure the smooth execution of projects. Both overestimating and underestimating the amount of work required to develop software can cause delays in project completion, increased expenditures, and even significant disruptions to the process of digital transformation.

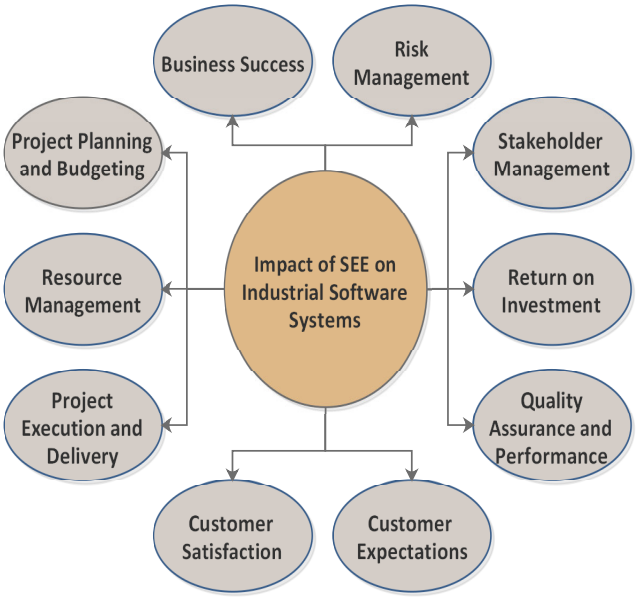

FIGURE 2. Impacts of SEE in Industrial Software Systems.

2) Resource Management
Estimating the amount of work required to produce software is an essential part of efficiently managing resources during the development of a project. It provides assistance to organisations in determining the appropriate resources, such as the appropriate number of team members, skill sets, and experience needed to effectively wrap up the project. When organisations have accurate estimations, they are able to distribute their resources effectively. This ensures that the appropriate people with the appropriate skills are accessible at the appropriate time, which in turn reduces waste of resources and boosts production.
3) Project Execution and Delivery
The estimation of the amount of work needed to complete a software project has an effect on both the execution and delivery of the project in industrial software systems. When organisations have accurate projections, they are able to set more reasonable expectations and schedules for the completion of projects, which ultimately leads to improvements in project management and delivery. It assists in the identification of potential bottlenecks and the proactive handling of those bottlenecks, which ultimately leads to the timely completion and successful delivery which leads digital transformation programmes.
4) Stakeholder Management
Software project development often involve numerous stakeholders. These stakeholders can include business leaders, executives, project managers, developers and end-users. Estimating the amount of work involved in developing software is critical for managing the expectations of stakeholders. Accurate estimations help

TABLE 1. A state-of-art comparison of related studies.

Ref	Datasets	Models	Metrics	Description	Limitations
[18]	Collected data through archival research and interviews for agile projects	Exploratory longitudinal case study	-	The case study identified the importance of effort estimation, in agile software development taking into considerations a few correlations.	Discusses a few correlations, product size, team maturity, customer priority and multi-site development.
[16]	User Stories dataset with 345 stories	Gradient Boosted Trees	MAE, MMRE. Pred	An hybrid effort estimation in Agile Software Development (HyEEAse) was developed using GBT	Discussed how well change effect analysis supports expert judgement.
[20]	Usp, Usp05-ftx, NASA, COCOMO81, SDR, China and Desharnais	Weka tool for estimation	RSME, MAE, Pred, Correlation Coefficient, RRSE, MMRE and RAE	The Relief feature selection is utilised, the datasets are used in their nominal value format using filter tool.	The research emphasises data pre-processing but does not give importance to the changes in industry.
[15]	-	A survey on importance of effort estimation on software maintenance	-	The survey discusses the software maintenance problems which affect maintenance and effort estimation task.	The research only focuses on software maintenance.
[27]	Albrecht, Desharnais and Cocomo Nasa	Random Forest, REPTree, LR, SMOReg, M5P and MLP.	MMRE	Particle swarm optimisation based feature selection is performed, with various ML models in order to estimate effort and cost.	Individual models may be biased in terms of datasets and validity. Impact on ISS is not present.
[24]	Albrecht, China, Cocomo, Finnish, Kemerer, Miyazaki, Maxwell and Nasa	SVM, RF, MLP, LR and M5P	MAE	Utilised various bio-inspired and non-bio-inspired feature selection algo -rithms with five ML models	Only focus on the importance of feature selection in estimation. Impact on ISS is not present.
[25]	Albrecht, Cocomo, Desharnais, China, Miyazaki and ISBSG	kNN, SVR, MLP and DT	-	The paper utilised three feature selection techniques (filter) and optimised ensemble model with grid search optimisation.	The validity of ML models in order to predict effort and maintain the efficacy.
[17]	-	Kitchenham and Charter based review	-	Use case point-based effort prediction in order to show the research in the field of UCP.	The research only focuses on UCP. Impact of UCP on ISS is not present.
[21]	Promise Repository and ISBSG datasets	CART, kNN, MLP, RF and SVR	MAE, RMSE and R-square	The Bayesian optimisation based Adaboost ensemble learning is performed to boost the performance of effort estimation with RF method.	Individual models may be biased in terms of datasets and validity. Impact on ISS is not present.
[23]	COCOMO, Desharnais and Maxwell	Lightning Search Algorithm (LSA) with ANN	MRE, MdMRE, Pred, and MAE	3 Layer Effort Estimation is proposed with Analogy based feature selection technique, LSA algorithm and ANN for estimation and evaluating error.	LSA is a metaheuristic optimisation technique, it is susceptible to local optimum, which in turn can cause inaccurate convergence.
[26]	COCOMO81	Artificial Neural Network	RMSE	The DHBAM (double-hidden-layers-bidirectional-associative-memory) model of prediction has been developed.	The model is prone to over-fitting, and computationally complex. Impact on ISS is not present

in the management of stakeholder communications, the establishment of confidence among stakeholders, and the establishing of realistic schedules and deliverables. Inaccurate projections can lead to dismayed expectations, which in turn can cause disappointment and make it difficult to manage relationships with stakeholders.

5) Return on Investment
The path to a digital transformation project is paved with accurate software effort estimation, which is essential in estimating the expected ROI they can offer to an organisation. Organisations can analyse the costs of development against the anticipated advantages and outputs with the aid of accurate projections, allowing them to decide for themselves whether or not software solutions are feasible and viable for a digital transformation initiative.

6) Quality Assurance and Performance
Software effort estimation has an impact on quality assurance procedures in commercial software systems, as well as the performance and quality of the product being engineered. Organisations may budget enough time for testing, quality control, and performance optimisation tasks by using accurate estimation. The result is a high-quality product that satisfies consumer expectations because it is ensured that it is extensively tested, complies with quality standards, and works preferably.

7) Customer Satisfaction and Expectations
Customer satisfaction and managing customer expectations in industry software systems are directly impacted by software effort estimation. Accurate estimating aids in establishing reasonable client expectations for

project deliverables, deadlines, and features. By delivering the product according to the schedule and features promised, it enables businesses to better manage customer expectations and foster trust, which in turn increases satisfaction among consumers.

8) Business Success
The success of industry software systems, product engineering, and business outcomes are all eventually impacted by effort estimation. The success of a project can be adversely affected by project delays, cost overruns, and resource shortages, all of which can be prevented with accurate estimation. Inaccurate estimating, on the other hand, can lead to missed deadlines, budget overruns, and degraded software quality, which can have an impact on the software system's marketability and the organization's financial health.

9) Risk Management
In order to effectively manage risks in commercial software companies, effort estimation is always crucial. Accurate estimating enables businesses to recognise and prepare for project risks including resource constraints, technical difficulties, or shifting requirements. It gives businesses the ability to proactively control risks, eliminate potential problems, and guarantee smooth project development, which paves the way to digital transformation.

IV. METHODOLOGY

This section describes the techniques utilised in the paper, along with the description of datasets utilized for experiment purpose. The paper employs a combination of ensemble selection techniques along with an evolutionary algorithm, their functioning is later discussed in this section.

A. DATASET DESCRIPTION

Each software project included in the SDEE datasets is represented by a set of cost/effort drivers. These features, which were used as inputs to the prediction algorithms and therefore affect the estimation accuracy of various ML techniques, include Lines of Codes (LOCs), Function Points (FPs), duration, efforts in man/hours or man/months, input files, output files, added files, and so on. This article is about the digitization of industrial software systems, the paper conducted research on multiple datasets based on the impact of software effort estimation on industrial software systems moving towards digitization. The paper then selected two datasets; Finnish and Maxwell in terms of the impacts stated above in Section III. The detailed description of the datasets used in the experiment in terms of a few statistical values is depicted in Table 2.

The TIEKE group gathered Finnish [29] data from nine Finland organisations' projects. The dataset was first made available in 1997. The dataset has 407 observations and 46 characteristics. In the field of software effort estimation, it is regarded to be the largest dataset with the most attributes.

TABLE 2. Dataset Description.

Datasets	Finnish	Maxwell
Attributes	46	28
Observations	407	62
Unit of measure	Function Point	Function Point
Mean Effort	5031.0148	8223.2097
Median Effort	2500	5189.5
Min Effort	55	583
Max Effort	63694	63694
Skewness	3.70	3.34
Kurtosis	18.69	13.69

The project's size is measured in Function Points. The field 'worksup' in this dataset indicates the level of effort. Because two observations have missing values, 405 observations are considered out of 407 observations.

The Maxwell [30] dataset was gathered by K.D. Maxwell from a Finnish commercial bank. It include the details of Software projects completed between 1985 and 1993. It was first released in the year 2002 [31]. The collection has 62 entities and 28 attributes. 22 of the 28 traits have a direct impact on software development. Function Points is the size attribute.

B. FEATURE SELECTION

Feature selection [32], also known as variable selection, is the process of executing a data preparation step with the goal of reducing data size by selecting the most relevant and important information to provide to a prediction model. Data pre-processing provides several advantages, including improving the effectiveness of a prediction approach, reducing dataset size, reducing training time, minimising complexity, and avoiding the over-fitting problem.

These datasets contains attributes which are related to the digital transformation in the industrial software systems impacted by software effort estimation phase in software companies. These attributes includes; Involvement of customer representative, Performance and availability of the development environment, Availability of IT staff, Number of stakeholders, Pressure on schedule, Impact of standards, Quality requirements of software, Analysis skills of staff, Application knowledge of staff, Tool skills of staff, Experience of project management, Team skills of the project team, Software Logical complexity, and Requirement volatility from both Finnish and Maxwell datasets. All these attributes from both Finnish (31 features out of 46 feature set including target attribute 'Worksup') and Maxwell (21 features out of 28 feature set including target attribute 'Effort') datasets are selected for the prediction purpose [33].

C. MACHINE LEARNING MODELS

The fundamental focus of machine learning, which is commonly referred to as a branch of artificial intelligence, is on the development of techniques and systems that will allow computers to learn and perform out responsibilities on their own. Machine learning techniques, which are analogous

to the human brain in certain ways, allow us to deal with difficulties swiftly. Machine learning is the process of training algorithms on data and then making predictions or acting on the learned patterns. In the last 20 years or so, machine learning methods have been advocated as an alternative method to estimate software labour.

One of the most actively researched areas in the field of machine learning is ensemble learning [34]. Given that no single learning algorithm can achieve optimal results across all datasets, this kind of learning is certain to arise. In ensemble approaches, we combine the results of several separate classifiers into a single consensus. Training many algorithms on the same training set (heterogeneous ensemble technique) and training a single algorithm on many training sets (homogeneous ensemble method) are two approaches to build classifiers. The results from each classifier are then combined using a combining algorithm to get a conclusion.

1) OMNI-ENSEMBLE SELECTION

The paper presents a method for choosing a base classifier in an ensemble system [35] that takes into account both its prior success in the specified domain and the reliability of its latest prediction. We are able to combine both the static and dynamic methods of ensemble selection collectively known as Omni-Ensemble Selection (OES) [36]. Ensemble methods use multiple classifiers to improve a learning model's predictive accuracy; the Static Ensemble Selection (SES) method searches a set of potential ensemble classifiers for the one that best meets a given criterion [34].

Dynamic ensemble selection (DES) [37] automatically selects a subset of ensemble members for prediction. This strategy involves fitting a large number of machine learning models to the training dataset and selecting the models that are projected to perform best when predicting a test case, taking into account the target's unique characteristics. The selection system chooses models that complement each other and improve performance based on accuracy, diversity, and stability.

D. GENETIC ALGORITHM

Genetic Algorithm (GA) [38] is a search algorithm based on Darwin's idea of natural selection. GA focuses on a subset of potential solutions that will converge to a globally optimal solution. GA avoids solution space local optima. GA has gained popularity as a problem-solving tool due to its search abilities.

The search begins with a population of random solutions. Each chromosome represents a solution to the challenge. Binary vectors represent chromosomes in the binary GA. The fitness function evaluates chromosomal candidates. Highly suited chromosomes develop a fresh set of appropriate-sized chromosomes. Crossover and mutation are used to develop new candidates (offspring) from a pair of individuals selected from the population. The process of Selection, crossover, and mutation form new populations, these new generations repeat fitness computation and offspring production and terminate after meeting stopping criteria, the method presents its best solution. GA selects the best classifiers for all test samples in this investigation.

To choose the training and testing sets from the dataset, the holdout approach is utilised. This is done so that it may be used for both the process of selecting classifiers using GA and the individual assessment of the various ways. In this approach, the dataset is separated into the training set and the testing set; hence, we consider 70% of the data to be part of the training set and the remaining 30% to be part of the testing set.

The fitness function of the genetic algorithm is R-square function that returns the square of the Pearson product moment correlation coefficient, which is a dimensionless index ranging from -1 to 1 that represents the strength of a linear relationship between two variables. The Pearson product moment correlation coefficient R_i of a single model i is calculated as:

$$R_i = \frac{n\sum_{j=1}^{n}(A_jP_{ij}) - (\sum_{j=1}^{n}A_j)(\sum_{j=1}^{n}P_{ij})}{\sqrt{\left[n\sum_{j=1}^{n}A_j^2 - (\sum_{j=1}^{n}A_j)^2\right] - \left[n\sum_{j=1}^{n}P_{ij}^2 - (\sum_{j=1}^{n}P_{ij})^2\right]}} \tag{1}$$

where, $P_{(ij)}$ is the value predicted by the individual model i for record j (out of n records/models); and A_j is the actual value for record j.

V. PROPOSED WORK

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 the Figure 3. First phase is data preprocessing, the datasets are imported and normalised 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 Section III. 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 optimisation of accuracy and diversity, from the pool of machine learning models. The pool contains diverse machine-learning regression models which are depicted in Table 3. The best-selected models from

TABLE 3. Regression Algorithms.

Algorithm	Parameter Description	Initials
Support Vector Machine	kernel= linear/rbf/polynomial, degree = 3	SVM
Random Forest	n_estimators=10, min_sample_leaf = 1	RF
Multi-layer Perceptron	hiddenlayer = 1, learningRate = 0.01	MLP
k Nearest Neighbours	k = 3/5/7	kNN
Decision Tree	criterion='squared_error', max_depth = 2	DT
Extra Tree	n_estimators =100, min_samples_leaf = 1	ET
Linear Regression		LR
AdaBoost	n_estimator = 10	ADA
CatBoost	iterations = 10, learning_rate = 1, depth = 2	CAT
XGBoost	n_estimator = 50, max_depth=3, eta = 0.1	XGB
Naive Bayes		NB
Bagging		BG

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 visualisation and analysis through the browser.

VI. RESULTS AND DISCUSSION

The section discusses the results of the experiment conducted with two datasets; Finnish and Maxwell. The evaluations are obtained based upon the predicted values and the performance of models (individual and proposed) is compared based on the evaluation metrics stated in this section.

A. EVALUATION METRICS

The metrics [39], [40] used for comparison of the performance of different machine learning algorithms and proposed Omni-Ensemble Selection are elaborated as follows;

1) **sMAPE**- symmetric Mean Absolute Percentage Error (sMAPE) is an error metric, expressed as a percentage. In other words, it is a metric of accuracy that uses percentage (or relative) error rates as its basis. Since MAPE is asymmetric, it penalises negative errors (where predictions are lower than actuals) more severely than positive errors. This is due to the fact that for too-low predictions, the margin of error cannot exceed 100%. sMAPE effectively makes up limitation in the traditional MAPE by including both a 0% and 200% limit. The equation for sMAPE calculation;

$$sMAPE(y, \hat{y}) = \frac{100\%}{N} \sum_{i=0}^{N-1} \frac{2 * |y_i - \hat{y}_i|}{|y| + |\hat{y}|} \tag{2}$$

2) **MRE**- The absolute error of the measurement is compared to the actual measurement, and the ratio of these two values is the definition of the relative error, is evaluated with equation;

$$MRE(y, \hat{y}) = \frac{1}{N} \sum_{i=0}^{N-1} \frac{|y_i - \hat{y}_i|}{|y_i|} \tag{3}$$

3) **MASE**- Mean Absolute Scaled Error (MASE) is a metric used to evaluate the quality of algorithm-generated forecasts by contrasting them with the results obtained from a naive forecasting method. If it has a value larger than one (1), the algorithm performed poorly. The equation to evaluate MASE is;

$$MASE(y, \hat{y}) = \frac{\frac{1}{N} \sum_{i=0} N - 1|y_i - \hat{y}_i|}{\frac{1}{N-1} \sum_{i=1}^{N-1} |y_i - y_{i-1}|} \tag{4}$$

4) **NSE**- The Nash-Sutcliffe efficiency (NSE) is a normalised statistic that calculates the relative amount of residual variance vs observed data variance. The Nash-Sutcliffe efficiency measures how well the observed vs simulated data plot fits the 1:1. The equation of NSE is given as;

$$NSE(y, \hat{y}) = 1 - \frac{\sum_{i=0}^{N-1} (y_i - \hat{y}_i)^2}{\sum_{i=0}^{N-1} (y_i - mean(y))^2} \tag{5}$$

5) **Coefficient of Determination (COD)**- COD is a statistic for determining how well a model matches your data. In the context of regression, it is a statistical measure of how well the regression model approximates the real data. COD values range between

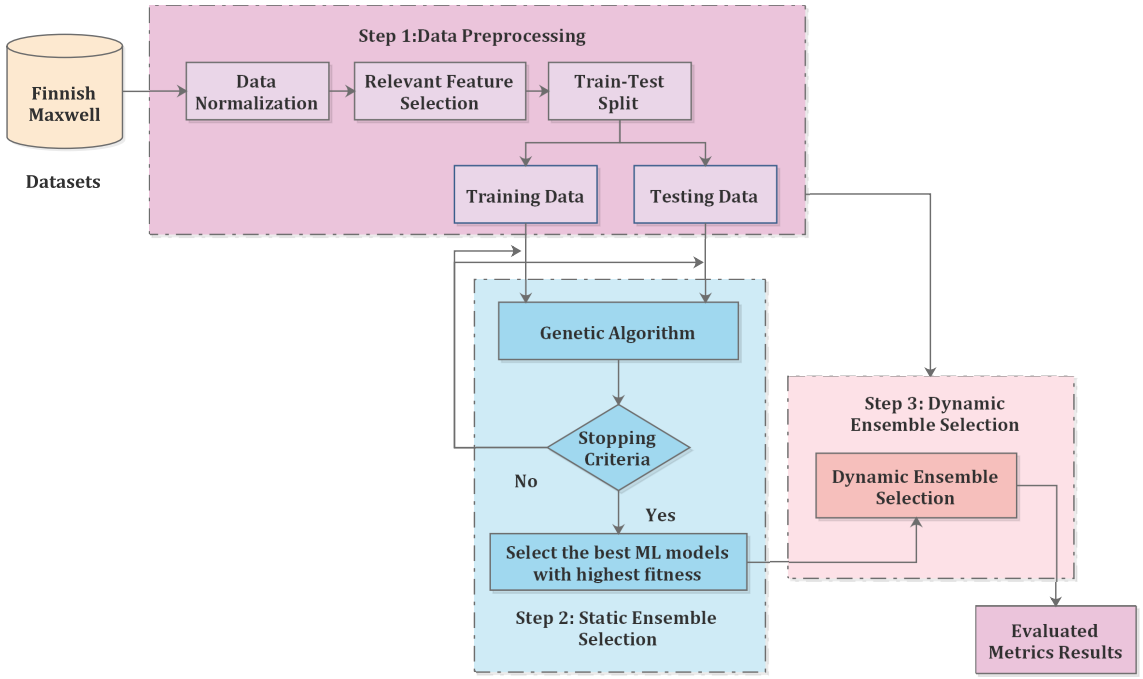

FIGURE 3. Flow Chart of the Proposed Work.

0 and 1. The model is selected if the COD value is near to or equal to 1. If the value is negative, there is no relationship between the data and the model. The equation comprises a formula for calculating COD value.

$$COD(y, \hat{y}) = 1 - \frac{\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{N-1}(y_i - y)^2} \tag{6}$$

where, y_i stands for actual value and $\vec{y_i}$ stands for predicted value and i represents the observations, with total N observations in all above stated equations.

B. RESULTS

Our experiments confirm that the proposed OES can provide better overall performance (in terms of evaluation metrics) on comparing with multiple machine learning models over Finnish and Maxwell datasets. The results shows, the effective effort estimation can impact the performance in the industrial software system, as the error metrics values are reduced to 23.896% and correlation between the attributes is increased to 91.375% in Finnish dataset. Similary the error metrics values are reduced to 15.057% and correlation between the attributes is increased to 98.359% in Maxwell dataset. Figure 4 and 5 illustrates the scatter graphs for the three investigated models; static ensemble selection, dynamic ensemble selection and proposed Omni-Ensemble selection model, which justifies the relationship between the predicted values and actual values. The research identifies on selecting relevant and mandatory attributes for effort estimation which directly impacts the industrial software system and later to digital transformation provides fruitful and effective results.

TABLE 4. Performance Evaluation over Finnish Dataset.

Models	sMAPE(in%)	MRE	MASE	NSE	COD
SVM	48.186	1.87874	0.81134	0.56357	0.80788
RF	29.145	1.04612	0.66909	0.36538	0.74756
MLP	38.703	1.89333	0.69446	0.64448	0.8107
kNN	35.681	1.36927	0.72336	0.56898	0.75544
DT	37.585	2.25293	0.92997	0.28421	0.62113
ET	27.747	0.98134	0.54617	0.68537	0.84353
LR	42.961	1.45984	0.67045	0.6332	0.84546
ADA	37.621	2.46283	0.80211	0.23775	0.72402
CAT	33.534	1.44861	0.69836	0.50544	0.79121
XGB	30.131	1.34004	0.69576	0.2715	0.77509
NB	38.732	1.19796	0.62284	0.67665	0.83747
BG	43.474	1.54597	0.70022	0.62224	0.8411
Static	27.825	1.05944	0.54115	0.71622	0.87652
DES	33.884	0.92209	0.5723	0.66953	0.84379
Proposed	23.896	0.79142	0.45167	0.78121	0.91375

1) COMPARISON WITH OTHER MODELS

In order to compare the performance of the machine learning regression models and the proposed Omni-Ensembled Selection (OES) model, based on Omni-Ensemble Learning (OEL) approach, on combining ensemble selection techniques along with genetic algorithm is represented in the tabular form. Table 4 compares the performance of individual machine learning models, ensemble models and proposed model with Finnish dataset.

Table 5 compares the performance of individual machine learning models, ensemble models and proposed model with maxwell dataset.

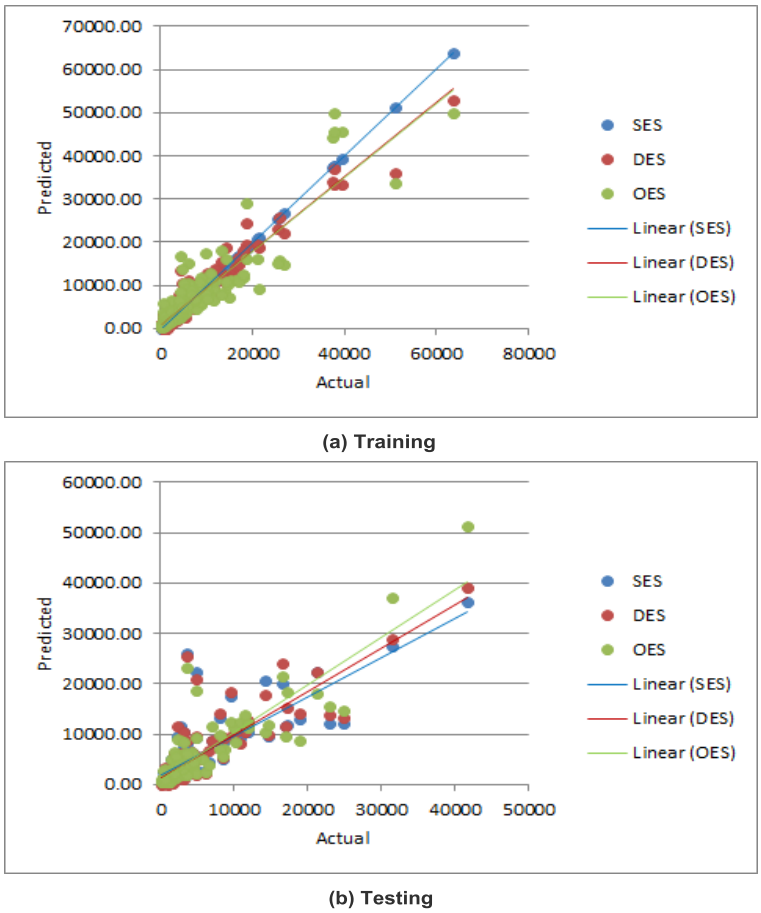

FIGURE 4. Scatter plot of three ensemble models over Finnish: (a) Training, (b) Testing.

TABLE 5. Performance Evaluation over Maxwell Dataset.

Models	sMAPE(in%)	MRE	MASE	NSE	COD
SVM	37.302	1.60133	0.64518	0.64141	0.85525
RF	18.274	0.69642	0.32081	0.88611	0.95035
MLP	45.079	2.61562	0.98424	0.12867	0.2717
kNN	35.082	1.4606	0.78834	0.43263	0.73278
DT	30.156	1.49436	0.6474	0.61278	0.79335
ET	18.59	1.15497	0.37111	0.805	0.89854
LR	65.296	14.70762	1.61663	0.28759	0.63149
ADA	24.89	0.96699	0.57493	0.43299	0.95678
CAT	36.453	1.54738	0.673	0.5657	0.87383
XGB	18.637	0.92803	0.29177	0.93225	0.96763
NB	39.123	1.0559	0.57534	0.66247	0.81607
BG	20.589	1.10704	0.35486	0.88439	0.94103
Static	23.527	0.7629	0.42057	0.8042	0.93988
DES	16.558	0.50287	0.32678	0.90453	0.9522
Proposed	15.057	0.47942	0.26125	0.95146	0.98359

It is evident based on both Tables 4 and 5 and both Figures 4 and 5 that Omni-Ensemble selection models shows best fit with both Finnish and Maxwell datasets. In order to calculate the relative amount of residual variance vs observed data variance, we evaluated Nash-Sutcliffe efficiency (NSE) used to represent the performance of the machine learning models and ensemble models, the highest NSE value obtained by the proposed OES model is 0.781 with the Finnish and 0.951 with the Maxwell dataset.

Based on the evaluated results, on comparing the two metrics; sMAPE and COD, its is observed that both metrics are inversely proportional to each other, as the error rate reduces the correlation between the attributes increases. Figures 6 and 7 shows the spider chart to indicate the inversely proportional relation between sMAPE and COD. The results indicates if the effort estimation is performed effectively, considering all the required attributes, which result in the successful development of the project in software companies which eventually result in the growth of industrial software system and will lead to digital transformation.

In comparison to the proposed work and related work of other researchers in the field of software effort estimation using same datasets; Finnish and Maxwell. Shepperd et al. [41] used analogies for estimating software effort over Finnish dataset by categorizing the software projects in terms of features. The performance is measured

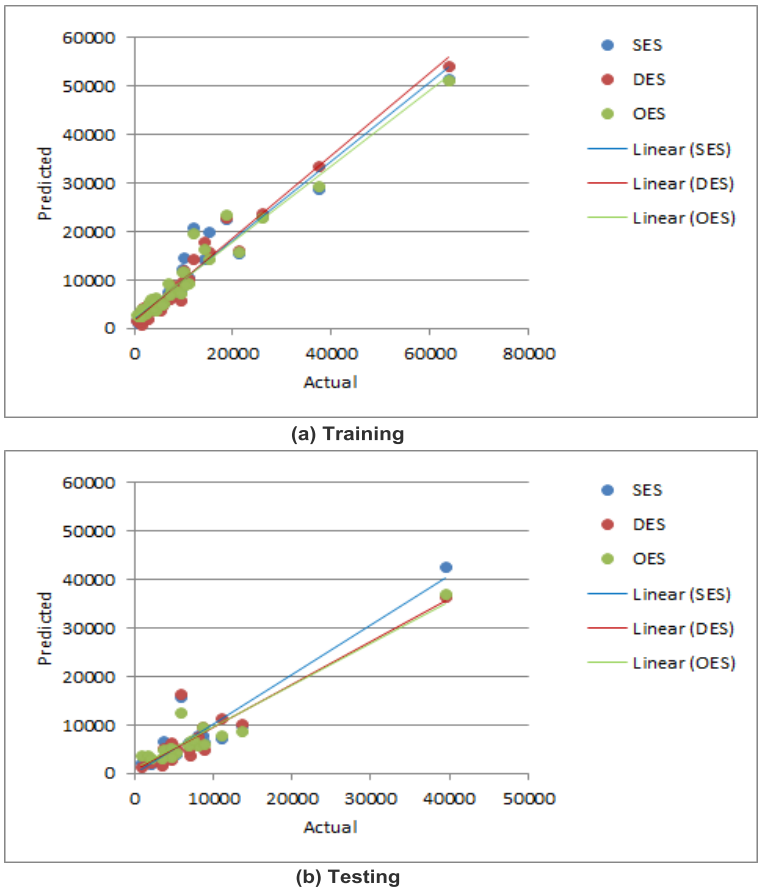

FIGURE 5. Scatter plot of three ensemble models over Maxwell: (a) Training, (b) Testing.

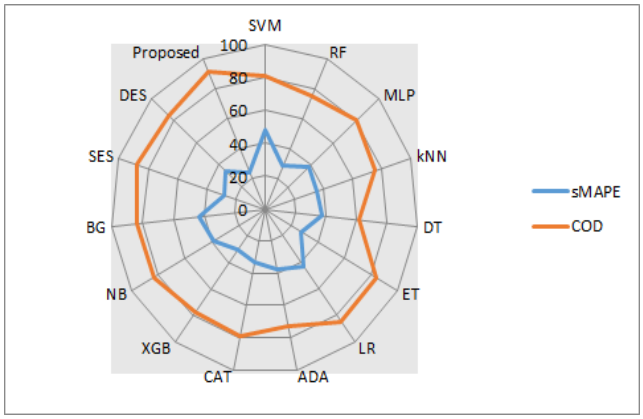

FIGURE 6. Graphical analysis Finnish: sMAPE vs COD.

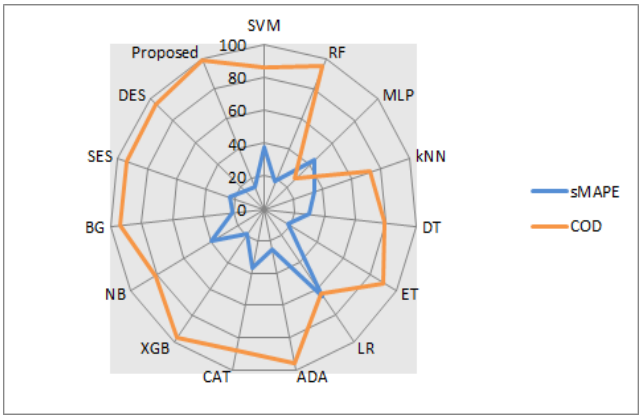

FIGURE 7. Graphical analysis Maxwell: sMAPE vs COD.

in terms of MMRE and Pred(25), while our proposed OES model outperforms the metrics and provides efficient results. The Maxwell dataset has been utilised by some researchers to estimate effort, Sree et al. [42] implemented neuro-fuzzy models for estimating effort, Elish [43] proposed emsemble learning and used voting classifier to extract the results using MMRE and Pred(25) metrics for evaluation. Shahpar et al. [44] used genetic algorithm for feature selection and evaluated MMRE, MdMRE and Pred(25) metrics for effort estimation. Jodpimai et al. [45] utilised correlation feature selection (CFS) for feature selection and GA for computing combined estimation by methods over six datasets. The results of our proposed OES approach outperformed the work by other authors.

TABLE 6. Wilcoxon T-test based p-value on Ensemble Models.

Models	Finnish	Maxwell
Static Ensemble Selection (SES)	0.236	0.112
Dynamic Ensemble Selection (DES)	0.075	0.056
Omni Ensemble Selection (OES)	0.004	0.021

2) STATISTICAL ANALYSIS

The Wilcoxon T-test was performed on both datasets Finnish and Maxwell, comparing the actual value to the predicted value to demonstrate statistical significance between the acquired accuracies from the various used machine learning and ensemble learning models. In non-parametric statistics, the Wilcoxon T-test (or Wilcoxon signed-rank test) compares two different samples. At the alpha = 0.05 significance level, it produces p-values for all models comparing observed and predicted data. If the p-value is less than 0.05, then the null hypothesis can be rejected; if it is between 0.05 and 1, then the samples are not statistically equivalent [46]. In hypothesis testing, the p-value is simply one piece of evidence, and it is crucial to consider the full statistical analysis, the context of the study, and anything else that might be relevant. Table 6 represents the obtained p-value over three ensemble learning models; Static, Dynamic and Omni-Ensemble Selection respectively. It shows the proposed OES approach is robust and statistically significant.

C. DISCUSSIONS

From section VI-B, it is crystal clear that our proposed Omni-Ensemble Selection outperforms in which we first extract the best suitable models from the pool of machine learning models using genetic algorithm-based static ensemble selection (SES-GA) and later the extracted models are fed to dynamic ensemble selection (DES) which estimates the efforts and results are compared on the basis of evaluation metrics. As the Omni-Ensemble Selection strategy is applied, it has enhanced the predictive accuracy. As it combines the estimates of different models, it has reduced bias and variance errors. The proposed OES strategy seems to be more robust and stable than previous approaches, as it combines the properties of both static and dynamic ensemble learning. Also as it combines models with different fusions, it integrates the strength of models and harnesses their collective power. Lastly, a generalised model is created which extracts the suitable models based on the dataset properties. But there are a few limitations as well, with the generalised model creation, additional computational resources are required such as more memory, processing power, and time. Other than computational resources the complexity of the model is increased. The interpretability and explainability of predictions are also challenging. One must ensure the proper training of models to reduce the complexity and time overhead.

Some concerns about the approach applied to this experimental set-up need to be discussed because they reflect on the validity of the analysis. Concerns about the internal construct include experimenter bias and related issues with datasets. The dataset characteristics vary substantially across the different publicly available datasets used in the experiment; the features of the datasets are extracted based on their impact on Industrial software systems. As a result, the analysis bias introduced by data selection can be reduced. Selecting appropriate predictive models requires making use of new evaluation criteria. This work uses principled ML techniques in a pool because the problem is of the regression kind; nevertheless, newer generations of ML may have a different ''mature'' perspective on how to approach addressing problems. Lastly, the situational factor; the proposed approach in the real world may vary with the company's perspective; the companies have different standards, such as CMM level, on which a company focuses in order to fulfil the demands of clients and maintain market value; and so on all pose threats to the external and conclusion validity.

VII. CONCLUSION AND FUTURE WORK

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 organisation in order to enhance operations, procedures, consumer experiences, and overall performance. Industrial software systems are software programmes 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.

ACKNOWLEDGMENT

This research is supported by the British Academy's Researchers at Risk Fellowships Programme.

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/methods-for-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.
[21] R. Marco, S. S. S. Ahmad, and S. Ahmad, "Bayesian hyperparameter optimization and ensemble learning for machine learning models on software effort estimation," *Int. J. Adv. Comput. Sci. Appl.*, vol. 13, no. 3, pp. 1–13, 2022.
[22] R. R. Sinha and R. K. Gora, "Software effort estimation using machine learning techniques," in *Advances in Information Communication Technology and Computing*, V. Goar, M. Kuri, R. Kumar, and T. Senjyu, Eds. Singapore: Springer, 2021, pp. 65–79.
[23] A. Moradbeiky, V. Khatibi, and M. Jafari Shahbazzadeh, "3LEE: A 3-layer effort estimator for software projects," *Int. J. Ind. Electron. Control Optim.*, vol. 5, no. 1, pp. 31–42, 2022.
[24] 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, Art. no. 102621.
[25] M. Hosni, A. Idri, and A. Abran, "On the value of filter feature selection techniques in homogeneous ensembles effort estimation," *J. Softw., Evol. Process*, vol. 33, no. 6, p. e2343, Jun. 2021.
[26] C. S. Yadav, R. Singh, S. Satpathy, S. B. Priya, B. T. Geetha, and V. Goyal, "Energy efficient and optimized genetic algorithm for software effort estimator using double hidden layer bi-directional associative memory," *Sustain. Energy Technol. Assessments*, vol. 56, Mar. 2023, Art. no. 102986.
[27] M. Z. Khan, "Particle swarm optimisation based feature selection for software effort prediction using supervised machine learning and ensemble methods: A comparative study," *Invertis J. Sci. Technol.*, vol. 13, no. 1, pp. 33–50, 2020.
[28] P. Palvia, A. Patula, and J. Nosek, "Problems and issues in application software maintenance management," *J. Inf. Technol. Manage.*, vol. 6, pp. 17–28, Jan. 1995.
[29] B. Sigweni, M. Shepperd, and P. Forselius, "Finnish software effort dataset," Tech. Rep., 2015. [Online]. Available: https://figshare.com/articles/dataset/FinnishEffortEstimationDataset/1334271
[30] Y. Li, "Effort estimation: Maxwell," Tech. Rep., Mar. 2009, doi: 10.5281/zenodo.268461.
[31] K. Maxwell, *Applied Statistics for Software Managers* (Software Quality Institute series). Upper Saddle River, NJ, USA: Prentice-Hall, 2002. [Online]. Available: https://books.google.co.in/books?id=irVQAAAAMAAJ
[32] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Comput. Elect. Eng.*, vol. 40, no. 1, pp. 16–28, Jan. 2014.
[33] J. Brownlee, "How to choose a feature selection method for machine learning," *Mach. Learn. Mastery*, vol. 10, pp. 1–13, Mar. 2019.
[34] T. T. Nguyen, A. V. Luong, M. T. Dang, A. W.-C. Liew, and J. McCall, "Ensemble selection based on classifier prediction confidence," *Pattern Recognit.*, vol. 100, Apr. 2020, Art. no. 107104.
[35] P. Suresh Kumar, H. S. Behera, J. Nayak, and B. Naik, "A pragmatic ensemble learning approach for effective software effort estimation," *Innov. Syst. Softw. Eng.*, vol. 18, no. 2, pp. 283–299, Jun. 2022.
[36] R. Mousavi, M. Eftekhari, and F. Rahdari, "Omni-ensemble learning (OEL): Utilizing over-bagging, static and dynamic ensemble selection approaches for software defect prediction," *Int. J. Artif. Intell. Tools*, vol. 27, no. 6, Sep. 2018, Art. no. 1850024.
[37] A. S. Britto, R. Sabourin, and L. E. S. Oliveira, "Dynamic selection of classifiers—A comprehensive review," *Pattern Recognit.*, vol. 47, no. 11, pp. 3665–3680, Nov. 2014.
[38] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: Past, present, and future," *Multimedia Tools Appl.*, vol. 80, no. 5, pp. 8091–8126, Feb. 2021.
[39] *Welcome to Permetrics's!*, 2021. [Online]. Available: https://permetrics.readthedocs.io/en/latest/
[40] A. Botchkarev, "Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology," 2018, *arXiv:1809.03006*.
[41] M. Shepperd and C. Schofield, "Estimating software project effort using analogies," *Ser. Softw. Eng. Knowl. Eng.*, vol. 16, p. 64, Jan. 2005.
[42] R. P. Sree, P. P. Reddy, and K. Sudha, "Hybrid neuro-fuzzy systems for software development effort estimation," *Int. J. Comput. Sci. Eng.*, vol. 4, no. 12, p. 1924, 2012.
[43] M. O. Elish, "Assessment of voting ensemble for estimating software development effort," in *Proc. IEEE Symp. Comput. Intell. Data Mining (CIDM)*, Apr. 2013, pp. 316–321.
[44] Z. Shahpar, V. Khatibi, A. Tanavar, and R. Sarikhani, "Improvement of effort estimation accuracy in software projects using a feature selection approach," *J. Adv. Comput. Eng. Technol.*, vol. 2, no. 4, pp. 31–38, 2016.

[45] P. Jodpimai, P. Sophatsathit, and C. Lursinsap, "Ensemble effort estimation using selection and genetic algorithms," *Int. J. Comput. Appl. Technol.*, vol. 58, no. 1, pp. 17–28, 2018.
[46] F. S. Nahm, "What the P values really tell us," *Korean J. Pain*, vol. 30, no. 4, pp. 241–242, Oct. 2017.

AKSHAY JADHAV received the B.E. degree in computer science engineering from the VNS Faculty of Engineering, Bhopal, Madhya Pradesh, India, in 2015, and the M.Tech. degree in computer technology and application from the National Institute of Technical Teachers Training and Research (NITTTR), Bhopal, in 2017. He is currently pursuing the Ph.D. degree in computer science engineering with VIT Bhopal University, under the supervision of Prof. Shishir Kumar Shandilya. His research interests include software engineering and health informatics.

SHISHIR KUMAR SHANDILYA (Senior Member, IEEE) is currently the Deputy Director of the SECURE-Centre of Excellence in Cyber Security, VIT Bhopal University. He is also a Visiting Researcher with Liverpool Hope University, U.K., a Cambridge University Certified Professional Teacher and a Trainer, a TEDx Speaker, and a ACM Distinguished Speaker. He is a NASSCOM Certified Master Trainer for Security Analyst SOC (SSC/Q0909: NVEQF Level 7) and an Academic Advisor to National Cyber Safety and Security Standards, New Delhi. He has ten books published by Springer Nature-Singapore, IGI-USA, River-Denmark, and Prentice Hall of India. His recently published book is *Advances in Nature-Inspired Cyber Security and Resilience* (Springer). His research paper on Cybertwin is recently published in IEEE TRANSACTIONS. He has international and national patents and copyrights granted on a NICS and an adaptive cyber defense methods. He has received the IDA Teaching Excellence Award for distinctive use of technology in teaching by the Indian Didactics Association, Bengaluru, in 2016, and the Young Scientist Award for two consecutive years, in 2005 and 2006, by the Indian Science Congress, and a MP Council of Science and Technology.

IVAN IZONIN (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in computer science from Lviv Polytechnic National University, Ukraine, in 2011, the B.Sc. and M.Sc. degrees in economic cybernetics from the Ivan Franko National University of Lviv, Ukraine, in 2012, and the Ph.D. degree in artificial intelligence from Lviv Polytechnic National University, in 2016. He is currently an Associate Professor with the Department of Artificial Intelligence, Lviv Polytechnic National University. His research interests include artificial neural networks, ensemble learning, noniterative training algorithms, and small data approach.

MICHAL GREGUS (Member, IEEE) received the Ph.D. degree (summa cum laude) in mathematical analysis from the Faculty of Mathematics and Physics, Comenius University in Bratislava. He has been working previously in the field of functional analysis and its applications. His current research interests include management information systems, in the modeling of economic processes and in business analytics.