Review of Existing Datasets Used for Software Effort Estimation

Mizanur Rahman[1], Teresa Gonçalves[2], Hasan Sarwar[3]
Faculty of Computing, Universiti Malaysia Pahang, 26600, Pekan, Pahang, Malaysia[1]
Associate Professor, Department of Computer Science, University of Évora, Portugal[2]
Professor, Department of Computer Science and Engineering
United International University, Satarkul, Badda, Dhaka, Bangladesh[3]

***Abstract*—The Software Effort Estimation (SEE) tool calculates an estimate of the amount of work that will be necessary to effectively finish the project. Managers usually want to know how hard a new project will be ahead of time so they can divide their limited resources in a fair way. In fact, it is common to use effort datasets to train a prediction model that can predict how much work a project will take. To train a good estimator, you need enough data, but most data owners don't want to share their closed source project effort data because they are worried about privacy. This means that we can only get a small amount of effort data. The purpose of this research was to evaluate the quality of 15 datasets that have been widely utilized in studies of software project estimation. The analysis shows that most of the chosen studies use artificial neural networks (ANN) as ML models, NASA as datasets, and the mean magnitude of relative error (MMRE) as a measure of accuracy. In more cases, ANN and support vector machine (SVM) have done better than other ML techniques.**

Keywords*—Software effort estimation; software effort prediction; software effort estimation datasets

I. Introduction

Estimating how much work has to be put into creating software is a hot topic of study because of its significance in any software development process. If the amount of work involved is underestimated, not enough money or manpower will be put into the project, leaving no room for error in terms of the final product's quality. On the other side, if you overestimate the amount of work that has to be done, you'll likely end up with a large budget, which will drive up the cost of the program and reduce its competitive edge. Therefore, it is crucial to have a reliable estimate of the time and effort required to complete a project. Bosu et al. [1] emphasized the significance of data, defining software metrics as the gathering of quantitative measures as an intrinsic element of software quality control and assurance operations (particularly, the monitoring and recording of errors during development and testing). This way of thinking has won out. SEE research has expanded to cover a wide variety of problems since its inception, although the largest collections of work in the discipline have either suggested or evaluated models designed largely for effort/cost estimation. It has been argued that the use of metrics in SEE is invaluable because it allows for more informed decision-making during software development and maintenance, which in turn improves development productivity, decreases deployment cycle time, and boosts software quality [2]. Although there is no debate about the benefits of metrics in theory for software engineering, in recent years there have been growing concerns about the quality of the data being gathered and used in the creation of models to predict factors like software size and development effort.

Several recent publications [3] [4] [5] [6] detail the difficulties in assembling and analyzing empirical software engineering datasets. While the SEE research community has acknowledged and prioritized the detection and resolution of issues like noise, outliers, and missingness (or incompleteness), they have largely ignored issues like poor provenance, inconsistency, and commercial sensitivity [1]. Several "classic" SEE datasets have been used extensively in previous research on software effort estimation, and we detail such datasets here. These datasets may be found mostly in the PROMISE repository. These datasets were chosen because of their convenience and widespread application in SEE modeling. Our goal is to perform a comparative analysis of the content of these datasets by using them as a baseline. This will serve to draw attention to any problems related to SEE data collection in general. As a bonus, we'll see how well it works as a comparative tool. By analyzing these established datasets, a standard model could be formulated by academics and practitioners would benefit greatly. Decisions on whether or not to employ a specific dataset in SEE modeling will be improved as a result of this.

Numerous estimation strategies are described in the SEE literature and are grouped into the three primary categories of algorithmic, non-algorithmic, and machine learning (see Fig. 1). For software work estimation, algorithmic strategies use statistical and mathematical formulation. Non-algorithmic models are based on evaluative and interpretive analyses. These models analyze historical data from previously completed projects. Techniques based on machine learning offer an alternative to algorithmic modeling [7]. There is no clear "front-runner" technique for any of the data quality issues in the considered dataset, despite the fact that several methods have been presented to identify or analyze the various quality aspects of SEE datasets. Therefore, we apply the best-in-class method(s) for evaluating the quality of these widely-referenced datasets in Table I. The ultimate goal of this benchmarking exercise is to promote a comprehensive evaluation of data quality before modeling by showing how appropriate techniques, such as algorithmic, non-algorithmic, and machine learning (Fig. 1) can be used by researchers and practitioners to evaluate the quality of their own datasets and inspire them to create or adopt new and improved methods of data collection. The following are the article's main contributions:

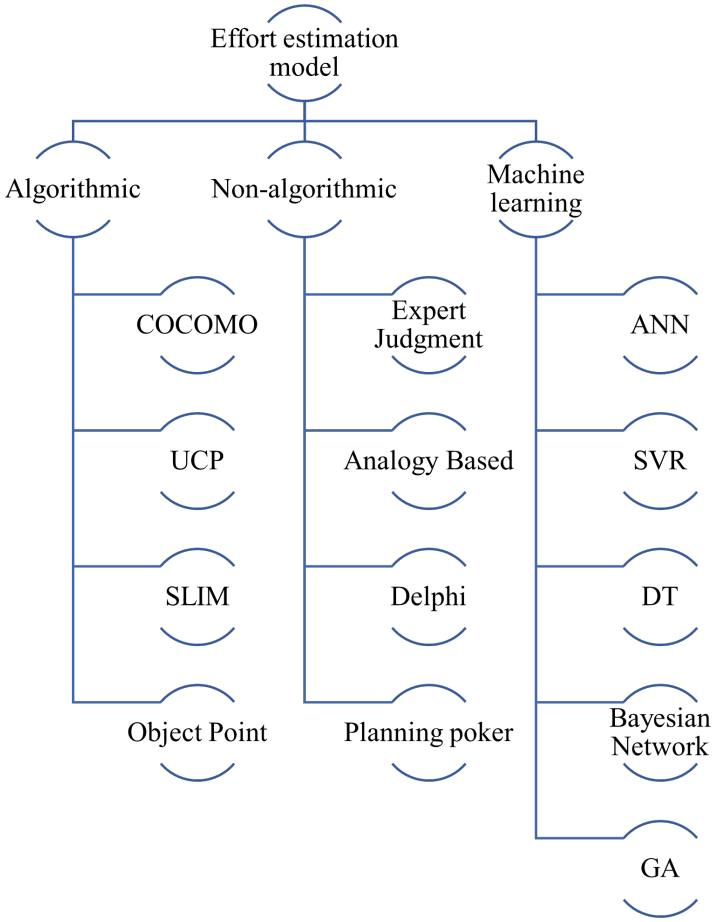

Fig. 1. SEE model description.

- We provide insights into some of the most popular datasets used for estimating software development efforts.
- We find out which is the most used technique for the SEE datasets.
- Which is the most used performance matrix, we also find out this for future researchers.

The rest of the article is divided into the following sections. Section II contains research findings related to the SEE dataset. Section III explains the methodology of our study. The details of the dataset are described in Section IV. The discussion of our work is covered in Section V. Section VI concludes the paper.

II. Related Work

It is impossible to exaggerate the significance of data to the study of SEE because they are at the center of the field's practical application. It is crucial that personnel in charge of gathering data be well-trained and aware of the various issues that could exist in datasets. It is seen that the majority of researchers use secondary data in SEE modeling [24]. Secondary data is research information that has already been obtained and is available to researchers. Secondary users are individuals who only infrequently make use of the product or those who do so via a third party. This is vital to make sure that the best procedures are used to produce and make use of the accessible data that is most trustworthy. In order to provide secondary users with an understanding of how the data were gathered, the processes that were used should, at the very least, be documented. In recent years, there has been a growing awareness of the difficulties associated with the collection and utilization of empirical software engineering datasets [25] [26], despite the fact that the overall body of literature on SEE data quality remains quite limited [27]. In this part, we examine previous assessment studies and provide a cursory mention of a few of the steps that others have done to enhance the quality of SEE datasets and repositories. First, We present a sample of studies that have assessed the state of SEE datasets from a particular angle or aspect of data quality. The SEE community mostly employs this approach to address problems that have an impact on software engineering datasets. The studies we present assess the state of SEE datasets from several dimensions of data quality. Additionally, we provide examples of studies in this chapter that built SEE prediction models using metrics from open-source projects. These metrics were gathered from the projects themselves. After this, a discussion of the few research works that have analyzed the current state of SEE datasets from numerous perspectives that have considered multiple data quality dimensions is discussed later. Table I shows the summary of the datasets used in this paper.

TABLE I. SUMMARY OF THE DATASET USED IN THIS STUDY

SL NO	Dataset name	Source	No of records	No of attributes	Output attribute-effort	Size(Unit measurement)	Ref
1	Desharnais	GitHub	81	12	Person-hours	Function point	[8]
2	COCOMO81	Promise	63	18	Person-months	LOC	[9]
3	China	Promise	499	16	Person-hours	Function points	[10]
4	Maxwell	Promise	62	27	Person-hours	Funciton points	[11]
5	Miyazaki94		48	8/9	Person months	‘KSLOC	[12]
6	Tukutuku		53	9	Person-months		[13]
7	ISBSG	ISBSG	1192	13	Man-hours	Multiple	[14]
8	Albrecht	Promise	24	8	Person-Months	Function point	[15]
9	Kemerer	Zenodo	15	7	Person-months	KSLOC	[16]
10	Kitchenham	SEACRAFT	145	9	Person-hours	Function Points	[17]
11	Nasa93	Promise	93	17	Person-months	LOC	[18]
12	UCP	Promise	70	17	Person-months	LOC	[19]
13	Edusoft	Github			Person-months		[20] [21]
14	Telecom		18	4	Person-months	Files	[22]
15	Finnish		38	9	Person-Hours	Function Points	[23]

TABLE II. LITERATURE REVIEW

SL No	Ref	Year	Datasets	Used Techniques	Evaluation matrix
1	[28]	2023	Albrecht, Kemerer, Miyazaki, China, COCOMO, Maxwell, NASA	GWO-FC, FCNN	MSE, RAE, MAE, RMSE, RRSE, MdMRE, R2
2	[21]	2023	Edusoft	KNN, SVR, DT	MAE, MSE, R-Square
3	[29]	2023	Albrecht, Maxwell, NASA, Telecom, Kemerer, China, Desharnais	CBR-GA	MAE, MBRE, MIBRE
4	[30]	2022	Deasharnais, COCOCMO81, China, Maxwell and Miyazaki94	ANN, SVR	MAR, MMRE
5	[31]	2022	China, Maxwell, and COCOMO81	DTR, RF, LR, LassoR, Ridge R,	MMRE, PRED(25)
6	[32]	2022	Desharnais, Cocomonasav1, COCOMONASA2	LR, RF, MP, SVM, Bagging, Stacking, Vote, CART_R, FRSBM_R, GA-HSBA, BHO-HSBA, FFA-HSBA	MAE, RMSE
7	[33]	2022	ISBSG	M5P, GBRegr, LinearSVR, and RFR	MAE, RMSE
8	[34]	2022	Cocomo81, NASA93, Maxwell and China	analogy-based software effort estimation	MMRE, MdMRE, PRED, SA
9	[5]	2022	ISBSG Release 2021, UCP, NASA93 and China	SVR, RF, Ridge Regression, KNN, and Gradient Boosting Machines	MAE, MSE, MdAE
10	[35]	2022	COCOMO81, CHINA	SGD, KNN, DT, bagging regressor, RFR, Ada-boost regressor, and gradient boosting regressor	MAE, MSE, RMSE, and R2
11	[36]	2021	Albrecht, China, Desharnais, Maxwell	Deepnet,NN, RF, SVM	MAE, RMSE, MSE and R-Squared
12	[37]	2021	NASA, COCOMO-81, China	Deep-MNN, GWDNNSB	MRE, MMRE, MBRE, MIBRE, PRED, MAE, SA, MR
13	[38]	2021	Albrecht, China, Desharnais, Kemerer, Kitchenham, Maxwell, and Cocomo8	RF, SVM, DT, NN, Ridge, LASSO, ElisticNet, DeepNet, Averaging, Bagging, Boosting, Stacking using RF	MAE, RMSE, and R-squared
14	[39]	2021	Desharnais	KNN, LR, SVM, ML	RMSE, MSE, MAE
15	[40]	2020	Desharnais, China, Albrecht	GP (genetic programming)	MMRE, Pred(25)
16	[41]	2020	NASA(93,63,60)	FPA	MMRE
17	[42]	2020	Desharnais	LR, RF, Multi-layer perceptron	CC, MAE, RMSE, RRAE, RSE
18	[43]	2020	COCOMO 81, Desharnais	KNN, SVM, NN, RF and backpropagation	MMRE
19	[44]	2020	NASA 93	COCOMO-II	MMRE, MRE

We conducted a metareview of the study and survey papers, and using that information, we were able to determine the SEE methodologies, datasets, and accuracy measurements that are most commonly employed. Table II provides an overview of the SEE methods, datasets, and accuracy measurements that are most frequently utilized.

We discuss the strengths and disadvantages of the most commonly used SEE approaches after doing a review of the relevant research and gaining an understanding of SEE techniques. Following are the SEE approaches which are most used by the researchers.

Linear regression: Linear regression is a statistical modeling technique used to examine the relationship between one or more independent variables and a dependent variable. It assumes that the variables are linearly related, meaning that changes in the dependent variable are proportional to changes in the independent variable. The benefits include being simple to grasp, apply, interpret, and clarify. Additionally, it is computationally cheap and works well with tiny datasets. Here are some of the flaws: it presupposes a linear connection and is therefore inappropriate for data with a nonlinear connection [45].

Artificial neural network: The following are its strengths: Feature engineering is not necessary, and it can learn complex correlations in the data. The shortcomings are as follows: training requires a lot of data, therefore it is computationally expensive. Additionally, it is challenging to evaluate the results since it is difficult to comprehend the assumptions that underlie them. It might have an overfitting issue, can’t handle missing values, and needs to convert categorical values to numeric values [45].

Analogy-based approaches: An analogy-based strategy is a way of thinking or addressing a problem that depends on discovering connections or analogies between various circumstances, things, or ideas. Analogies, which are similarities or correspondences between two or more items, are used to gather knowledge, draw conclusions, or resolve issues [34]. The following are its advantages: It can handle outliers and the justification for the result is simple to understand. The following are the flaws: computationally demanding, susceptible to the similarity function, requiring the conversion of categorical variables to numeric types, unable to handle missing values, and challenging to find a solution if similar work has not been done previously.

Fuzzy logic: Fuzzy logic is a mathematical foundation for reasoning and making decisions in ambiguous, imprecise, and uncertain situations. The following are the virtues: It is based on the theory of classes with flexible boundaries so that it can accommodate data uncertainty caused by measurement errors in data collection. It can also cope with model uncertainty. It enhances the performance of ML and non-ML models and resembles human reasoning. Its only limitation is that when combined with ML or non-ML models, it becomes computationally intensive.

Machine Learning (ML) techniques: The most popular ML methods for SEE are tree-based models and Support vector Machine(SVM) [46]. Each ML method has advantages and disadvantages of its own. The following are the strengths of the tree-based ML models that use the decision tree (DT) [47], CART, and random forest(RF) techniques: intuitive, making it simple to comprehend and analyze the model's findings. It is appropriate when there are nonlinear relationships in the data and can handle both category and numerical data. It can be said that it is capable of handling the outliers or that it is robust with them. The following are its flaws: If the dataset is small, DT is prone to overfitting. It is unable to handle missing values. A huge dataset has a high time complexity, and even a minor change in the data can significantly alter the model [48]. The following are some of SVM's advantages: It learns nonlinear relationships in the data and is appropriate for large dimensional data [48]. the following are the flaws: memory-intensive and possibly not scalable for large datasets.

Optimization techniques: Optimization techniques are mathematical methods and algorithms that are used to discover the optimal solution to a given problem given a set of constraints. The purpose is to maximize or reduce an objective function, which represents the quantity to be optimized, while adhering to specified conditions or constraints. The following are its advantages: it may be used to pick and weight features, and when combined with ML or non-ML techniques, it can increase estimation accuracy. the following are the flaws: Due to its nondeterministic nature and high processing cost, outcomes may differ from one attempt to the next [28].

III. Review Methodology

we downloaded all relevant papers and analyzed the number of SEE papers published in recent years (2020-2023(June)), categorizing them by the journals in which they appeared. Our searches began by retrieving all SEE files, which were then sorted by the publisher. For the search, we used reputable online resources such as Google, Springer, Elsevier, and similar databases. We compiled a research database of abstracts, keywords, and titles and carefully analyzed both the content and algorithms of each publication. Based on the inclusion/exclusion criteria we have selected the related paper. In this way, we were able to classify the publications on the topic of SEE.

A. Research Questions

We answer the following study questions:

- What are the most commonly utilized datasets in SEE research?
- What accuracy metrics are utilized most frequently in SEE studies?

B. Inclusion/Exclusion Criteria

Inclusion Criteria.

- Studies that used both ML and non-ML techniques for software effort estimation.
- Papers that are written in English.
- Papers that are published in a conference or journal.
- Paper that used publicly available datasets.

Exclusion Criteria.

- The title, abstract, or even their content was not closely related to our search string, however without any semantic interplay.
- No similarity with the research theme, or even the focal aim was completely contrary to the purpose of the issues addressed in the RQs.
- Not based on publicly available datasets.

When searching for information in the literature, we utilized the following search strings: "software effort estimation" OR "software cost estimation." The aims of the study as well as its research questions were taken into consideration when developing the search string.

C. Dataset Description

For accurate effort forecasts, the dataset's quality that supports an estimating model's central premise is crucial. According to this theory, a characteristic is only considered a trustworthy predictor of the outcome if it has a strong correlation with the effort; else, it is unimportant. Datasets considered for this review paper were Desharnais, Cocomo81, China, Maxwell, Miyazaki94, Tukutuku, ISBSG, Albrecht, Kemerer, Kitchenham, Nasa93, and Edusoft dataset. The mean, also known as the average, is a measure of central tendency that is calculated by dividing the sum of all values in a dataset by the total number of data points. The standard deviation measures the spread or dispersion of data points around the mean. It expresses how far the data deviates from the average. Min is the dataset's minimum value, and Max is the dataset's maximum value.

CHINA dataset: The CHINA dataset for predicting software effort consists of 19 attributes. There are 499 different project instances in total. Table III provides the CHINA dataset's descriptive statistics.

1) Maxwell dataset: The Maxwell dataset, which was compiled from one of the largest commercial banks in Finland, has 62 projects that are each described by 23 attributes. Table IV provides a comprehensive summary of the Maxwell dataset. Project Size in Function Points serves as the only numerical attribute.

2) COCOMO81 dataset: The COCOMO81 dataset was often used to verify various effort estimation techniques. It consists of 63 software projects, each of which is described by 18 qualities along with a concrete effort. The COCOMO81 dataset measures real effort in terms of person-months, which are the number of months required for one person to develop a certain project. Table V provides the full description of the COCOMO81 dataset.

3) Albrecht dataset: The Albrecht dataset includes 24 software programs created with third-generation languages like COBOL, PL1, etc. Six independent number attributes and one dependent numeric attribute, "work hours," which indicates the appropriate effort in 1000 hours, are used to define the dataset. Projects were developed in COBOL, PL1, and database management languages for the remainder. Table VI gives a detailed explanation of the Albrecht dataset.

4) Desharnais dataset: The Desharnais dataset began with 81 software projects gathered from Canadian software businesses. This dataset is described by ten attributes: two dependent factors (time and effort in 'person-hours') and eight independent attributes. Unfortunately, four projects out of 81 had missing values, therefore we eliminated them because they could have influenced the estimation procedure. This data preprocessing stage yielded 77 finished software projects. Table VII provides the full description of the Desharnais dataset.

5) Kemerer dataset: In the Kemerer dataset, which consists of 15 software projects, six qualities and one predictable attribute with a "man-month" unit of measurement are used to define the projects. Two categories and four numerical qualities each represent one of the six attributes. Table VIII provides the full description of the Kemerer dataset.

6) Miyazaki94 dataset: Miyazaki provided the Miyazaki94 dataset. 48 software projects are represented in this collection. In total, there are nine qualities. Seven of the nine attributes are conditional attributes, one is a decision attribute, and one is an identifier. Table IX provides the full description of the Miyazaki94 dataset.

7) Tukutuku dataset: In the Tukutuku dataset, 53 online projects are present. Nine numerical attributes are used to describe each online application, including the quantity of media assets, HTML or SHTML files used, and team experience. Table X provides the full description of the Tukutuku dataset.

8) UCP dataset: The UCP dataset consists of 70 instances with 17 attributes for software effort estimation. Table XI provides the full description of this dataset.

9) NASA dataset: NASA datasets have been used to assess how well evolutionary algorithms function. Bailey and Basili provided this dataset in 1981. Shin and Goel utilized it for the first time in 2000, followed by Oliveira in 2006. There are 18 project instances in the dataset. M (methodology utilized) and DL (number of developed lines of source code with comments) are two independent qualities. The dependent characteristic of effort is the number of man-months needed to complete the project. Table XII provides the full description of the NASA dataset.

10) ISBSG dataset: ISBSG21 dataset version 2021 includes 10,531 cross-company projects from various nations, organizations kinds, and development kinds. We chose the dataset instances in accordance with ISBSG rules and the procedure described in [49]. As a result, 1179 novel project category IFPUG version 4+ projects with quality A and B have been created. The feature selection was carried out in accordance with the technique proposed by Dejaeger et al. [50]. The features include project sequencing-related features, features that are not expected to be accessible at the time of first estimation highly correlated features, and features with only one value. Table XIII provides the full description of the ISBSG dataset.

11) Telecom dataset: The Telecom dataset contains information on 18 software development initiatives for a UK telecommunication product. The dataset version utilized in this investigation has four attributes. However, since the other three variables are not available at the time of the work estimation, just the number of files attribute is employed. Table XIV provides the full description of the Telecom dataset.

12) Finnish dataset: The TIEKE organization obtained the Finnish dataset from nine Finnish businesses. Initially, 40 records were obtained, however, because of missing values in some of the attributes of two projects (Kitchenham and Kansala 1993), their data were eliminated, leaving 38 records for analysis. This dataset has nine attributes, each of which has a size measured in function points. Table XV provides the full description of the Finish Dataset.

13) Edusoft dataset: For estimating the time and effort needed for software development using a real-world dataset compiled by Edusoft Consultant Ltd. Noteworthy features of this dataset include task history ID, project ID, client id, task types, task priority, task overall state, total working time in hours, etc. 2000 samples of real-time data make up our dataset.

IV. Discussion

A. What ML Approaches are Employed in SEE Studies?

The process of estimating the time, resources, and cost needed to accomplish a software development project is known as software effort estimation. A particular kind of regression issue is software effort estimation. Regression attempts to forecast a continuous numerical number, in this case, the amount of work needed, using information from the input (such as project size, complexity measures, and historical data). Various project parameters and variables that affect the amount of work necessary for development could be included in the input features.

To estimate the SEE, the following ML techniques were utilized either alone or in combination with other (ML and nonML) estimation techniques. Artificial neural network

TABLE III. CHINA DATASET DESCRIPTION

SL no	Features	Details about the features	Data Type	Feature Selection	Mean	Std Dev	Min	Max
1	ID		Numerical		250	144	1	499
2	AFP	Adjusted function points	Continuous integer	AFP	487	1059	9	17518
3	Input	Function points of input	Continuous integer	Output	167	486	0	9404
4	Output	Function points of external output	Continuous integer	File	114	221	0	2455
5	Enquiry	Function points of external output enquiry	Continuous integer	Interface	62	105	0	952
6	File	Function points of internal logical files	Continuous integer	Added	91	210	0	2955
7	Interface	Function points of external interface added	Continuous integer	PDR_AFP	24	85	0	1572
8	Added	Function points of added functions	Continuous integer	NPDR_AFP	260	830	0	13580
9	Changed	Function points of changed functions	Continuous integer	NPDU_UFP	85	291	0	5193
10	Deleted		Continuous integer	N-Effort	12	124	0	2657
11	PDR_AFP	Productivity delivery rate(adjusted function points)	Continuous Double	Effort	12	12	0.3	83.8
12	PDR_UFP	Productivity delivery rate(Unadjusted function points)	Continuous Double		13	14	0.4	101
13	NPDR_AFP	Normalized productivity delivery rate(adjusted function points)	Continuous Double		14	15	0.4	108
14	NPDU_UFP	Productivity delivery rate(Unadjusted function points)	Continuous Double		1	1	1	4
15	Resource	Team type	Discrete		12	12	0.3	83.8
16	Dev.Type		Numerical Only {0}		0	0	0	0
17	Duration	Total elapsed time for the project	Continuous integer		9	7	1	84
18	N_effort	Normalized effort	Continuous integer		4278	7071	31	54620
19	Effort	Summary work report	Continuous integer		3921	6481	26	54260

TABLE IV. MAXWELL DATASET DESCRIPTION

Sl No	Features	Details about the features	Data Type	Feature Selection	Mean	Standard Dev	Min	Max
1	year	The year in which the project started	Continuous	Year				
2	Har	The hardware platform on which the application is being developed	Discrete		2.61	1	1	5
3	App	The name of the application being developed	Discrete		2.35	0.99	1	5
4	Dba	The database management system being used for the project	Discrete		1.03	0.44	0	4
5	Ifc	The user interface technology being used	Discrete		1.94	0.25	1	2
6	Source	The source code management system being used	Discrete	Source	1.87	0.34	1	2
7	Telonuse	Whether or not the project is using IBM Telon, a legacy mainframe application development tool	Binary		2.55	1.02	1	4
8	Nlan	The number of programming languages being used in the project	Discrete	Nlan	0.24	0.43	0	1
9	T01	Customer participation	Discrete		3.05	1	1	5
10	T02	Development environment adequacy	Discrete		3.05	0.71	1	5
11	T03	Staff availability	Discrete		3.03	0.89	2	5
12	T04	Standards use	Discrete		3.19	0.70	2	5
13	T05	Methods use	Discrete	T05	3.05	0.71	1	5
14	T06	Tools use	Discrete		2.90	0.69	1	4
15	T07	Software's logical complexity	Discrete		3.24	0.90	1	5
16	T08	Requirements volatility	Discrete		3.81	0.96	2	5
17	T09	Quality requirements	Discrete	T09	4.06	0.74	2	5
18	T10	Efficiency requirements	Discrete		3.61	0.89	2	5
19	T11	Installation requirements	Discrete		3.42	0.98	2	5
20	T12	Staff analysis skills	Discrete		3.82	0.69	2	5
21	T13	Staff application knowledge	Discrete		3.06	0.96	1	5
22	T14	Staff tool skills	Discrete		3.26	1.01	1	5
23	T15	Staff team skills	Discrete	T15	3.34	0.75	1	5
24	Duration	The duration of the project in months	Continuous	Duration	17.21	10.65	4	54
25	Size	The size of the project in terms of lines of code	Continuous	Size	673.31	784.08	48	3643
26	Time	The total amount of time spent on the project, in person-months	Discrete	Time	5.58	2.13	1	9
27	Effort	The total amount of effort expended on the project, in person-months	Continuous	Effort	8223.21	10499.90	583	63694

TABLE V. COCOMO81 DATASET DESCRIPTION

SL NO	Features	Details about the features	Data Type	Feature Selection	Mean	Std Dev	Min	Max
1	Rely	Required software reliability	Double	Rely	1.036	0.193	0.75	1.4
2	Data	Database size	Double	Data	1.004	0.073	0.94	1.16
3	Cplx	Product Complexity	Double		1.091	0.203	0.7	1.65
4	Time	Execution time constraint	Double	Time	1.114	0.162	1	1.66
5	Stor	Main storage constraint	Double	Stor	1.144	0.179	1	1.56
6	Virt	Virtual machine volatility	Double		1.008	0.121	0.87	1.3
7	Turn	Required turnabout time	Double		0.972	0.081	0.87	1.15
8	Acap	Analyst capability	Double	Acap	0.905	0.152	0.71	1.46
9	Aexp	Applications experience	Double		0.949	0.119	0.82	1.29
10	Pcap	Programmer capability	Double		0.937	0.167	0.7	1.42
11	Vexp	Virtual machine experience	Double		1.005	0.093	0.9	1.21
12	Lexp	Programming language experience	Double		1.001	0.052	0.95	1.14
13	Modp	Use of modern programming practices	Double	Modp	1.004	0.131	0.82	1.24
14	Tool	Use of software tools	Double		1.017	0.086	0.83	1.24
15	Sced	Required development schedule	Double	Sced	1.049	0.076	1	1.23
16	Loc	Lines of code	Double	Loc	77.21	168.509	1.98	1150
17	Effort	Actual effort expended in person-months	Double	Effort	683.321	1821.582	5.9	11400

TABLE VI. ALBRECHT DATASET DESCRIPTION

SL No	Features	Details about the features	Data Type	Feature Selection	Mean	Std Dev	Min	Max
1	Input	The number of inputs that a program has to process.	Integer	InquiryNumeric	40.25	36.913	7	193
2	Output	Output: the number of outputs produced by a program.	Integer	OutputNumeric	47.25	35.169	12	150
3	Inquiry	The number of inquiries or questions that a program has to answer.	Integer		16.875	19.337	0	75
4	File	The number of files that a program needs to read from or write to.	Integer		17.375	15.522	3	60
5	FPAdj	Function Point Adjustment Factor, which adjusts the raw function points according to specific attributes of the software.	Double		0.989	0.135	0.75	1.2
6	RawFPcounts	The raw function points are calculated based on the Function Point Metrics.	Double	RawFPcounts	638.53	452.653	189.52	1902
7	AdjFP	The adjusted function points are calculated by multiplying the raw function points with the Function Point Adjustment Factor.	Integer	AdjfpNumeric	658.875	492.204	199	1902
8	Effort	The software development effort is measured in person-months.	Double	Effort				

TABLE VII. DESHARNAIS DATASET DESCRIPTION

SL No	Features	Details about the features	Data Type	Feature Selection	mean	Std dev	min	Max
1	Project	Project Number	Discrete					
2	TeamExp	Team experience in years	Discrete		2.185	1.415	-1.00	4.00
3	ManagerExp	Project managers experience in years	Discrete		2.531	1.644	-1.00	7.00
4	YearEnd	Year of completion	Discrete		85.741	1.222	82.00	88.00
5	Length	Length of the project	Continuous		11.667	7.425	1.00	39.00
6	Effort	Measured in person	Continuous	Effort	5046.309	4418.767	546.00	23940.00
7	Transaction	Number of transactions processed	Continuous	Transactions	182.123	144.035	9.00	886.00
8	Entities	Number of entities	Continuous		122.333	84.882	7.00	387.00
9	PointsNonAdjust	Unadjusted function points	Continuous	PointsNonAdjust	304.457	180.210	73.00	1127.00
10	Adjustment	Adjustment factor	Continuous		27.630	10.592	5.00	52.00
11	PointsAjust	Adjustment function points	Continuous	PointsAjust	289.235	185.761	62.00	1116.00
12	Language	Programming language	Categorical					

TABLE VIII. KEMERER DATASET DESCRIPTION

Sl No	Features	Details about the features	Data Type	Feature Selection	Mean	Std Dev	Min	Max
1	ID	Identifier for the project.						
2	Language	The programming language used for the project.	Nominal					
3	Hardware	The type of hardware used for the project.	Nominal		2.333333	1.676163	1	6
4	Duration	The duration of the project in months.	Numerical	Duration	14.26667	7.544787	5	31
5	KSLOC	The estimated size of the project is in the thousands of source lines of code.	Numerical	KSLOC	186.5733	136.8174	39	450
6	AdjFP	Adjusted function points.	Numerical	AdjFP	999.14	589.5921	99.9	2306.8
7	RAWFP	Unadjusted function points.	Numerical	RawFP	993.8667	597.4261	97	2284
8	EffortMM	Effort measured in person-months.		Efforts	219.2479	236.0554	23.2	1107.31

TABLE IX. MIYAZAKI94 DATASET DESCRIPTION

Sl No	Features	Details about the features	Data Type	Feature Selection	Mean	Std Dev	Min	Max
1	ID							
2	KSLOC	COBOL source lines in thousands excluding comment lines	Continuous	KLOC	70.792	87.5678	6.9	417.6
3	SCRN	number of different input or output sceens	Discrete	SCRN	33.39	47.27	0	281
4	FORM	Number of different (report) forms	Discrete	FROM	22.38	20.55	0	91
5	FILE	Number of different record formats	Discrete		34.81	53.36	2	370
6	ESCRN	Total number of data elements in all the screens	Discrete		525.60	626.058	0	3000
7	EFORM	Total number of data elements in all the forms	Discrete		460.67	396.816	0	1566
8	EFILE	Total number of data elements in all the files	Discrete		1854.58	6398.605	57	45000
9	MM	Man-Months form system	Continuous	Man Month	87.475	228.7597	5.6	1586.0

TABLE X. TUKUTUKU DATASET DESCRIPTION

SL no	Features	Details about the features	Data Type	Mean	Standard Dev	Min	Max
1	Teamexp	Average number of years of experience the team has on Web development	Ratio	3.8	2.0	1	10
2	Devteam	Number of people who worked on the software project	Ratio	2.6	2.4	1	23
3	TotWP	Number of web pages in the application	Ratio	69.5	185.7	1	2000
4	Textpages	Number of text pages in the application (text page has 600 words)					
5	TotImg	Number of images in the application	Radio	98.6	218.4	0	1820
6	Anim	Number of animations in the application					
7	Audio/video	Number of audio/video files in the application					
8	Tot-high	Number of high effort features in the application	Radio	14.64	66.59	0	611
9	Tot-nhigh	Number of low effort features in the application	Radio	4.82	4.98	0	35

TABLE XI. UCP DATASET DESCRIPTION

SL	Feature	Mean	Std dev	Min	Max
1	Simple_Actors	0.71	0.90	0	4
2	Average_Actors	0.94	0.88	0	3
3	Complex_Actors	2.62	1.50	0	6
4	Simple_UC	2.7	2.89	0	20
5	Average_UC	15.84	5.37	3	30
6	Complex_UC	14.28	4.45	5	27
7	T1	0.41	1.14	0	5
8	T3	1.35	2.02	0	5
9	T4	2.3	2.37	0	5
10	T5	2.92	2.37	0	5
11	T6	3.14	2.43	0	5
12	T7	3.71	2.06	0	5
13	T9	4	1.95	0	5
14	T10	4.47	1.43	0	5
15	T11	4.9	0.38	3	5
16	ENV2	0.04	0.26	0	2
17	ENV3	0.18	0.72	0	5
18	ENV4	1.21	1.37	0	4
19	ENV5	2.64	1.88	0	5
20	ENV6	4.15	1.01	1	5
21	ENV7	1.57	1.55	0	5
22	ENV8	3.84	1.25	0	5
23	Real_Effort	6558.72	664.23	5775	7970

(ANN), Support vector machine (SVM), Bayesian network (BN), K-nearest neighbors (kNNs), Decision tree (DT), Genetic programming (GP), Classification and regression tree (CART), CBR, Random forest (RF).

B. Which Datasets are most Commonly Utilized in SEE Research? (Research Question 1)

The selected studies made use of around 15 different datasets. We look for datasets that have been used in at least one study. we showed a literature review in Table II. The NASA dataset is one of the most extensively used datasets in the SEE literature, and it has been utilized in several research. Most of the datasets related to SEE are provided by different software companies. A few datasets come from different sources. The sources of these datasets are listed below.

China dataset basically focuses on function points. The Maxwell dataset was obtained from a Finnish commercial bank so anyone can work with bank data if it needs for his/her research. The multinational American company Computer Sciences Corporation provided the Kitchenham dataset. Ten Canadian organizations provided data for the Desharnais dataset. The NASA93 dataset represents 14 distinct application types and was gathered by NASA from five of its development centers. The Albrecht dataset was created using third-generation programming languages such as COBOL, PL1, and database management languages. The ISBSG Repository now houses software projects acquired from various global software development firms. The Telecom dataset was created by software development initiatives on a telecommunication product used in the United Kingdom. The Tukutuku dataset contains audio, video, animation, and webpages, so it might be useful for candidates who work with graphical data. The TIEKE group gathered the Finnish dataset from nine enterprises in Finland. Edusoft dataset can be used to estimate the time and effort required for software development using a real-world dataset provided by Edusoft Consultant Ltd.

C. Performance Evaluation Matrix (Research Question 2)

In a regression problem, the effectiveness of a predictive model that seeks to estimate a continuous target variable is evaluated using evaluation metrics. Several measures have been established and used for assessing the accuracy of a prediction model in the literature on software work estimation. Prediction error (or absolute error), which is the difference between the anticipated value and the actual value, serves as the foundation for these metrics in most cases [45]. Before it was discovered that they are biased towards underestimates and behave differently when comparing different prediction models, the Mean of

TABLE XII. NASA93 DATASET DESCRIPTION

SL No	Features	Details about feature	Mean	Std dev	Min	Max
1	Rely	Requires software reliability	1.11	0.13	0.88	1.40
2	Data	Size of the application database	1.00	0.07	0.94	1.16
3	Cplx	Complexity of the product	1.18	0.15	0.85	1.65
4	Time	Run-time performance constraints	1.13	0.20	1.0	1.66
5	Stor	Memory constraints	1.13	0.19	1.0	1.56
6	Virt	Volatility of the virtual machine environment	0.92	0.09	0.87	1.15
7	Turn	Required turnabout time	0.96	0.09	0.87	1.15
8	Acap	Analyst capability	0.89	0.09	0.71	1.00
9	Aexp	Application Experience	0.93	0.06	0.82	1.13
10	Pcap	Software engineer capability	0.91	0.10	0.70	1.00
11	Vexp	Virtual machine experience	1.00	0.08	0.90	1.21
12	Lexp	Programming language experience	0.97	0.05	0.95	1.14
13	Modp	Application of software engineering methods	0.98	0.09	0.82	1.24
14	Tool	Use of software tools	1.00	0.09	0.83	1.24
15	Sced	Required development schedule	1.04	0.04	1.00	1.08
16	Loc		94.02	133.6	0.90	980.0
17	actual		624.41	1135.93	8.40	8211.00

TABLE XIII. ISBSG2021 DATASET DESCRIPTION

SL No	Features	Mean	Std dev	Min	Max
1	Input count	147.32	219.99	0	
2	Output count	123.99	171.12	0	1337.0
3	Enquiry count	90.74	136.59	0	952.0
4	File count	129.61	166.42	0	1252.0
5	Interface count	49.10	90.59	0	977.0
6	Developer	2182.82	2097.72	70.0	6610.0
7	Functional size	620.67	947.46	6.0	16148.0
8	Value adjustment Factor	1.01	0.08	0.65	1.29
9	Normalised Work Effort Level 1	6679.57	13336.10	40.0	230514.0

TABLE XIV. TELECOM DATASET DESCRIPTION

SL No	Features	Mean	Std Dev	Min	Max
1	Files	110.33	91.33	3	284.00
2	Effort	284.34	264.71	23.54	1,115.54

Magnitude of Relative Error (MMRE), Mean of Magnitude of Relative Error Relative to Estimate (MEMRE), and Prediction at Level l (Pred(l)) were the three most widely used metrics. As a result, their use is discouraged, and future studies should rely on standardized measurements such as the Sum of Absolute Errors (SAE), Mean Absolute Error (MAE), and Standard Accuracy (SA) that are not skewed towards underestimates or overestimates. A technique is regarded as better ML if it has more accuracy, for example, a lower mean magnitude of relative error (MMRE) number. For example, if model A is compared to model B, and A has a lower MMRE value in the majority of investigations, we can argue that model A outperforms model B. We may conclude that MSE, MRE, and MMRE are the most often used performance evaluation matrices based on the literature review Table II mentioned in Section 2.

TABLE XV. FINISH DATASET DESCRIPTION

SL No	Features	Mean	Std Dev	Min	Max
1	The type of hardware	1.26	0.64	1	3
2	AR	2.24	1.50	1	5
3	Function Points	763.58	510.83	65	1814
4	CO	6.26	2.73	2	10
5	Effort	7678.29	7135.28	460	26670

V. CONCLUSION

In this study, we examined the datasets used for software effort assessment and discovered 15 widely used SEE datasets. We've talked in depth about these datasets. Based on our study of the most recent article, we discover that the NASA dataset is the most often used dataset for software effect estimation. The majority of these datasets also lacked timing details. Since software engineering is an ever-evolving field, it is crucial that SEE records include dates for important events like project launches and wrap-ups. This would allow researchers and practitioners to build models over time, allowing them to examine the impact of new development strategies. It was unclear in numerous cases whether datasets were obtained from a single company or from several. Given the continuous discussion over which datasets produce the best prediction accuracy results, it seems only right that this information be disclosed alongside datasets made available for modeling purposes. We also discover that the most popular machine learning techniques are Decision trees (DT), K-nearest neighbors (kNNs), Bayesian networks (BN), Support vector machines (SVM), and Artificial neural networks (ANN). On the other hand, Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) are the most commonly utilized performance evaluation matrices for software effort estimation.

One limitation of our paper is that we did not build those datasets using alternative machine learning algorithms, which would have allowed us to provide a more detailed description of the dataset. In the future, we will experiment with these datasets using machine learning methods in order to better visualize the results of the evaluation matrix and other pertinent information.

ACKNOWLEDGMENT

The work reported in this paper is funded by the Institute for Advanced Research (IAR), United International University (UIU), Bangladesh titled: Implementation of 3D model to

assess the performance improvement of a software company UIU/IAR/02/2019-20/SE/06.

REFERENCES

[1] M. F. Bosu and S. G. Macdonell, "Experience: Quality benchmarking of datasets used in software effort estimation," *Journal of Data and Information Quality (JDIQ)*, vol. 11, no. 4, pp. 1–38, 2019.

[2] I. Noorwali, "A requirements measurement program for systems engineering projects: Metrics, indicators, models, and tools for internal stakeholders," Ph.D. dissertation, The University of Western Ontario (Canada), 2020.

[3] A. Baghel, M. Rathod, and P. Singh, "Software effort estimation using parameter tuned models," *arXiv preprint arXiv:2009.01660*, 2020.

[4] M. Azzeh, "Dataset quality assessment: An extension for analogy based effort estimation," *arXiv preprint arXiv:1703.04575*, 2017.

[5] S. S. Gautam and V. Singh, "Adaptive discretization using golden section to aid outlier detection for software development effort estimation," *IEEE Access*, vol. 10, pp. 90 369–90 387, 2022.

[6] M. F. Bosu, S. G. MacDonell, and P. A. Whigham, "Analyzing the stationarity process in software effort estimation datasets," *International Journal of Software Engineering and Knowledge Engineering*, vol. 30, no. 11n12, pp. 1607–1640, 2020.

[7] Y. Mahmood, N. Kama, A. Azmi, A. S. Khan, and M. Ali, "Software effort estimation accuracy prediction of machine learning techniques: A systematic performance evaluation," *Software: Practice and experience*, vol. 52, no. 1, pp. 39–65, 2022.

[8] J. Desharnais, "Desharnais dataset," https://www.kaggle.com/datasets/toniesteves/desharnais-dataset/code?datasetId=50782&sortBy=dateRun&tab=collaboration, 2018.

[9] A. Kaushik, A. Chauhan, D. Mittal, and S. Gupta, "Cocomo estimates using neural networks," *International Journal of Intelligent Systems and Applications (IJISA)*, vol. 4, no. 9, pp. 22–28, 2012.

[10] F. H. Yun, "China: Effort estimation dataset," in *Zenodo, Switzerland, Tech.*, 2010.

[11] Y.-F. Li, M. Xie, and T. Goh, "A study of mutual information based feature selection for case based reasoning in software cost estimation," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5921–5931, 2009.

[12] S. Amasaki, "miyazaki94," Feb. 2016. [Online]. Available: https://doi.org/10.5281/zenodo.268473

[13] B. Kitchenham and E. Mendes, "A comparison of crosscompany and within-company effort estimation models for web applications," 01 2004.

[14] 2021, "The international software benchmarking standards group," in *Available: http://www.isbsg.org*, 2021.

[15] J. W. Li, Yanfy; Keung, "Effort estimation: Albrecht," Apr. 2010. [Online]. Available: https://doi.org/10.5281/zenodo.268467

[16] J. W. Keung, "kemerer," Apr. 2010. [Online]. Available: https://doi.org/10.5281/zenodo.268464

[17] M. Tsunoda, "kitchenham," Feb. 2017. [Online]. Available: https://doi.org/10.5281/zenodo.268457

[18] T. Menzies, "nasa93," Feb. 2008, Instances: 93 Attributes: 24 -15 standard COCOMO-I discrete attributes in the range Very_Low to Extra_High -7 others describing the project -one lines of code measure -one goal field being the actual effort in person months. [Online]. Available: https://doi.org/10.5281/zenodo.268419

[19] R. Silhavy, "Use case points benchmark dataset," https://zenodo.org/record/344959, 2017.

[20] E. C. Ltd, "Software effort estimation," https://github.com/edusoftresearch/SEE_Data, 2023.

[21] M. Rahman, P. P. Roy, M. Ali, T. Gonc¸alves, and H. Sarwar, "Software effort estimation using machine learning technique," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 4, 2023. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2023.0140491

[22] M. Azzeh, "Software effort estimation based on optimized model tree," in *Proceedings of the 7th International Conference on Predictive Models in Software Engineering*, 2011, pp. 1–8.

[23] B. Sigweni, M. Shepperd, and P. Forselius, "Finnish Software Effort Dataset," 3 2015. [Online]. Available: https://figshare.com/articles/dataset/Finnish_Effort_Estimation_Dataset/1334271

[24] M. Shepperd, D. Bowes, and T. Hall, "Researcher bias: The use of machine learning in software defect prediction," *IEEE Transactions on Software Engineering*, vol. 40, no. 6, pp. 603–616, 2014.

[25] M. F. Bosu and S. G. MacDonell, "A taxonomy of data quality challenges in empirical software engineering," in *2013 22nd Australian Software Engineering Conference*. IEEE, 2013, pp. 97–106.

[26] M. Shepperd, Q. Song, Z. Sun, and C. Mair, "Data quality: Some comments on the nasa software defect datasets," *IEEE Transactions on Software Engineering*, vol. 39, no. 9, pp. 1208–1215, 2013.

[27] M. F. Bosu and S. G. MacDonell, "Data quality in empirical software engineering: a targeted review," in *Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering*, 2013, pp. 171–176.

[28] S. Kassaymeh, M. Alweshah, M. A. Al-Betar, A. I. Hammouri, and M. A. Al-Ma'aitah, "Software effort estimation modeling and fully connected artificial neural network optimization using soft computing techniques," *Cluster Computing*, pp. 1–24, 2023.

[29] S. Hameed, Y. Elsheikh, and M. Azzeh, "An optimized case-based software project effort estimation using genetic algorithm," *Information and Software Technology*, vol. 153, p. 107088, 2023.

[30] S. Goyal, "Effective software effort estimation using heterogenous stacked ensemble," in *2022 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES)*, vol. 1. IEEE, 2022, pp. 584–588.

[31] M. Jawa and S. Meena, "Software effort estimation using synthetic minority over-sampling technique for regression (smoter)," in *2022 3rd International Conference for Emerging Technology (INCET)*. IEEE, 2022, pp. 1–6.

[32] W. Rhmann, B. Pandey, and G. A. Ansari, "Software effort estimation using ensemble of hybrid search-based algorithms based on metaheuristic algorithms," *Innovations in Systems and Software Engineering*, pp. 1–11, 2021.

[33] Z. Sakhrawi, A. Sellami, and N. Bouassida, "Software enhancement effort estimation using correlation-based feature selection and stacking ensemble method," *Cluster Computing*, vol. 25, no. 4, pp. 2779–2792, 2022.

[34] A. Kaushik, P. Kaur, and N. Choudhary, "Stacking regularization in analogy-based software effort estimation," *Soft Computing*, pp. 1–20, 2022.

[35] P. Suresh Kumar, H. Behera, J. Nayak, and B. Naik, "A pragmatic ensemble learning approach for effective software effort estimation," *Innovations in Systems and Software Engineering*, vol. 18, no. 2, pp. 283–299, 2022.

[36] A. P. Varshini, K. A. Kumari, D. Janani, and S. Soundariya, "Comparative analysis of machine learning and deep learning algorithms for software effort estimation," in *Journal of Physics: Conference Series*, vol. 1767, no. 1. IOP Publishing, 2021, p. 012019.

[37] M. S. Khan, F. Jabeen, S. Ghouzali, Z. Rehman, S. Naz, and W. Abdul, "Metaheuristic algorithms in optimizing deep neural network model for software effort estimation," *IEEE Access*, vol. 9, pp. 60 309–60 327, 2021.

[38] K. K. Anitha, V. Varadarajan *et al.*, "Estimating software development efforts using a random forest-based stacked ensemble approach," *Electronics*, vol. 10, no. 10, p. 1195, 2021.

[39] H. D. P. De Carvalho, R. Fagundes, and W. Santos, "Extreme learning machine applied to software development effort estimation," *IEEE Access*, vol. 9, pp. 92 676–92 687, 2021.

[40] K. Mahadev and G. Gowrishankar, "Estimation of effort in software projects using genetic programming," *Int. J. Eng. Res. Technol.(IJERT)*, vol. 9, no. 07, pp. 1321–1325, 2020.

[41] B. Khan, R. Naseem, M. Binsawad, M. Khan, and A. Ahmad, "Software cost estimation using flower pollination algorithm," *Journal of Internet Technology*, vol. 21, no. 5, pp. 1243–1251, 2020.

[42] A. Singh and M. Kumar, "Comparative analysis on prediction of software effort estimation using machine learning techniques," in *Proceedings of the International Conference on Innovative Computing & Communications (ICICC)*, 2020.

[43] P. Suresh Kumar and H. Behera, "Estimating software effort using neural network: an experimental investigation," in *Computational Intelligence in Pattern Recognition: Proceedings of CIPR 2020*. Springer, 2020, pp. 165–180.

[44] A. A. Fadhil and R. G. Alsarraj, "Exploring the whale optimization algorithm to enhance software project effort estimation," in *2020 6th International Engineering Conference "Sustainable Technology and Development"(IEC)*. IEEE, 2020, pp. 146–151.

[45] R. K. Gora and R. R. Sinha, "A study of evaluation measures for software effort estimation using machine learning," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 11, no. 6s, pp. 267–275, 2023.

[46] I. H. Sarker, "Machine learning: Algorithms, real-world applications and research directions," *SN computer science*, vol. 2, no. 3, p. 160, 2021.

[47] A. Najm, A. Zakrani, and A. Marzak, "Systematic review study of decision trees based software development effort estimation," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 7, 2020.

[48] T. Mahboob, S. Gull, S. Ehsan, and B. Sikandar, "Predictive approach towards software effort estimation using evolutionary support vector machine," *International journal of advanced computer science and applications*, vol. 8, no. 5, 2017.

[49] L. Song, L. L. Minku, and X. Yao, "Software effort interval prediction via bayesian inference and synthetic bootstrap resampling," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 28, no. 1, pp. 1–46, 2019.

[50] K. Dejaeger, W. Verbeke, D. Martens, and B. Baesens, "Data mining techniques for software effort estimation: a comparative study," *IEEE transactions on software engineering*, vol. 38, no. 2, pp. 375–397, 2011.