Received: 20 November 2018 | Revised: 16 May 2019 | Accepted: 12 June 2019

DOI: 10.1002/smr.2211

RESEARCH ARTICLE

WILEY Journal of Software: Evolution and Process

A systematic literature review of software effort prediction using machine learning methods

Asad Ali | Carmine Gravino

Department of Computer Science, University of Salerno, Fisciano, Italy

Correspondence
Asad Ali, Department of Computer Science, University of Salerno, Fisciano, Italy.
Email: abasyn.asad@gmail.com

Abstract

Machine learning (ML) techniques have been widely investigated for building prediction models, able to estimate software development effort as well as to improve the accuracy of other estimation techniques. The objective of this paper is to systematically review the recent studies which used and discussed the software effort estimation models built using ML techniques. The performed literature review is based on the empirical studies published in the time period of January 1991 to December 2017, by employing widely used guidelines. The review has selected a total of 75 primary studies after the careful filtering of inclusion/exclusion and quality assessment criteria. The performed analysis reveals that artificial neural network (ANN) as ML model, NASA as dataset, and mean magnitude of relative error (MMRE) as accuracy measure are widely used in the selected studies. ANN and support vector machine (SVM) are the two techniques which have outperformed other ML techniques in more studies. Regression techniques are the mostly used among the non-ML techniques, which outperformed other ML techniques in about 19 studies. Moreover, SVM and regression techniques in combination are characterized by better predictions when compared with other ML and non-ML techniques.

KEYWORDS

accuracy measure, effort estimation, machine learning, SLR, software engineering

1 | INTRODUCTION

Although a lot of work has been done in the area of software effort estimation during the past three decades, a more accurate estimation of software projects is still a big concern of industries. Accurate prediction of software development effort is very important because inaccurate estimations can lead to the wastage of both time and financial resources of organizations. Therefore, predicting the effort of software development is the primary activity of all organizations. The main issue of the software cost estimation is first obtaining an accurate size estimate of the software to be developed[1] Software effort estimation involves the prediction of efforts (in person-months), the cost of the project (in dollars), and the duration need to complete the project (calendar time). Most of the competitive software organizations invest a lot to accurately estimate the development of software projects, because both an overestimation or underestimation of a software project can affect both the project development and the customer satisfaction.[2] According to Corazza et al,[3] an effective software project cost estimation is the one which is monitored by the development team and project manager, accepted to all stakeholders, and is based on a database of relevant past project experience.

In the past few decades, researchers have proposed various methods for what is called software development effort estimation (SDEE). The early three widely used software effort estimation approaches are constructive cost model (COCOMO), function point–based model, and Putnam's software lifecycle model (SLIM) which describe the effort estimation using a formula, parameterized from historical data.[4,5] These methods need some input attributes like experience of the development team, the programming language of the software to be built, the delivered source line of code (SLOC), and the required reliability of the software.

Jorgensen[6] highlighted 11 estimation methods for SDEE, among them regression analysis, and the use of expert judgment is dominant. In expert judgment, the effort estimation is carried out by taking advices from experts who previously (successfully) completed similar projects. Similar in the case of estimation by analogy,[7] the current projects are compared with the similar projects completed in the past and determine the expected estimation.

Software effort estimation using machine learning (ML)[8,9] based methods got attention of researchers and has been extensively used since 1991. ML allows to perform the estimation using information of previously finished projects. By applying this learning mechanism, the experts spend less time on the estimation of the proposed project and more time on other functionality of the software system which will satisfy the customer. However, only in the last decade they have been widely investigated with the aim of comparing their prediction accuracy with the one of other techniques (algorithmic models, expert judgment, etc). Indeed, in the 1990s, the attention of researchers was focused on the use of non-ML techniques.[10] As examples of studies that investigated and compared these learning mechanisms (also against non-ML approaches), Chavoya et al[11] evaluated the potential of genetic programming (GP) and provided comparisons with linear support regression (LSR) and artificial neural networks (ANNs) based on the projects of the Desharnais dataset. Prabhakar and Maitreyee[12] described and compared three ML models ANN, case-based reasoning (CBR), and rule induction (RI) for software effort estimation. The estimation models were evaluated based on the accuracy of estimation, explanatory value, and configurability. Aljahdali et al[13] evaluated the use of support vector regression (SVR) for web development effort estimation and performed comparisons with other techniques such as manual stepwise regression, CBR, and Bayesian networks. Idri and Elyassami[14] investigated the use of fuzzy decision tree which helps in handling uncertain and imprecise data for software effort estimation based on Tukutuku dataset.

Jorgensen et al[10] performed a systematic literature review (SLR), and the results revealed that the use of ML techniques has been increased from the early 2000 and the algorithmic models (non-ML techniques) became scarce with the passage of time. From the provided analysis and interpretation of the results, we believe that the main reason is the possibility of obtaining better effort predictions. Furthermore, it is worth noting that two recent SLRs about the use of ML techniques for software development effort estimation[15] and software fault prediction,[16] respectively, have analyzed the studies that compared the accuracy of these techniques with the one of non-ML techniques. The analysis provided in the first SLR suggests that ML techniques performed better in 66% of the identified experiments, while the second SLR highlighted that they outperformed non-ML techniques in 65% of the identified experiments. As a further consideration, both the studies recommend to investigate more ML techniques, also focusing on those not employed in a sufficient number of studies.

All the above considerations have motivated us to provide a summary as well as findings of the work done on the application of ML approaches for effort estimation since 1991. To this aim, we have performed a systematic literature review of the studies published from January 1991 till December 2017, following the guidelines suggested by Kitchenham.[17] Covering this time period allows us to update the results already provided by a previous SLR by Wen et al,[15] which included the studies from 1991 to 2010, and obtain further considerations on the usefulness of ML techniques taking into account the passage of time. Furthermore, we believe that a substantial amount of work has been done in the last 7 to 8 years which are based on the software effort estimation using ML methods. Thus, the results and conclusions derived by Wen et al might have changed in these last 7 years, and the purpose of our SLR is to figure out what has been changed, if there are any. Moreover, in the previous SLR,[15] five research questions were considered, while we covered these five questions plus some additional ones. The new analysis can also contribute to confirm the positive trend of applying ML techniques for effort estimation previously highlighted by Jorgensen et al[10] in their SLR. Studies have been selected from five major databases, ie, Scopus, Springer, ScienceDirect, ACM Digital Library, and IEEE Xplore based on inclusion/exclusion and quality assessment criteria.

The organization of this paper is as follows: In Section 2, we discuss the various steps we followed to conduct the SLR. Results and discussions are reported in Section 3 as well as answers to our research questions. Section 4 presents the summary of findings while in Section 5, we present the conclusion.

2 | THE METHOD USED TO CONDUCT THE SLR

The planning, the execution as well as the description of the result analysis of our systematic literature review (SLR) follow the guidelines suggested by Kitchenham,[17] which are shown in Figure 1. In the planning phase, we apply the review protocol including the following six stages: (1) research question identification, (2) design of search strategy, (3) study selection, (4) study quality assessment, (5) data extraction, and (6) data synthesis.[18]

According to this protocol, in the first phase, we setup a few research questions that are related to the objectives of the SLR. While considering these research questions, in the second phase of the SLR, we designed a search strategy to identify studies that will help in answering our research questions. In this stage, we define a search string as well as the literature resources to identify our iterative and unbiased search strategy. The third phase is about finding all the relevant studies from selected literature resources which are based on our research questions. We define the

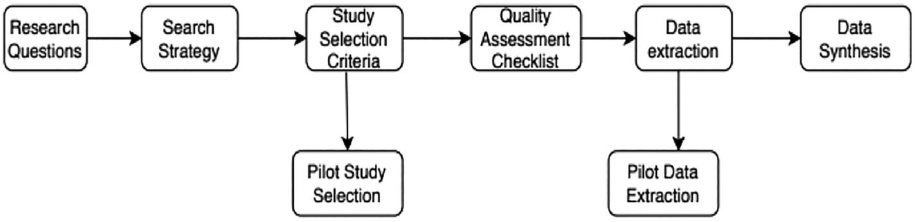

FIGURE 1 Stepwise description of systematic literature review (SLR) protocol

inclusion and exclusion criteria to determine what studies to include and what to discard. Then, we define the quality assessment criteria to gauge the strength and quality of individual studies. To assess the studies, we decided to use the values of fuzzy linguistic variables instead of crisp values. In the fifth phase, we collect all the important information which adhere our research questions and store them in a table (called the data extraction form). In the final phase of the SLR, we analyze and synthesize the extracted data based on the research questions.

In any SLR, the development of review protocol is of immense importance in order to minimize the risk of research biases. In the following subsections, we describe in detail all the steps of our SLR.

2.1 | Research questions

The goal of our SLR is to clarify and assess the empirical evidence resulted from various studies using the ML-based techniques for software development effort estimation. To this aim, we have formulated and analyzed the following research questions:

RQ1. Which are the ML techniques used in SDEE studies?

RQ2. Which is the ML technique that outperforms other ML techniques in terms of effort estimation accuracy?

RQ3. Do ML techniques provide better results in terms of effort prediction accuracy than non-ML techniques?

RQ4. Which combinations of ML and non-ML techniques provide better results in terms effort estimation accuracy than single ML or non-ML techniques?

RQ5. Which are the datasets most frequently used in SDEE studies?

RQ6. Which are the accuracy measures widely used in SDEE studies assessing ML techniques?

RQ7. Which are the dominant journals for papers analyzing ML techniques for SDEE?

We have selected the above research questions with the aim of covering all the aspects investigated by Wen et al in their previous SLR[15] as well as to investigate further aspects. For this reason, we explicitly focused on datasets and accuracy measures employed by the studies investigating ML techniques for effort estimation as well as publication venues. These aspects were also partially discussed in the previous SLR but without formulating specific research questions. Furthermore, apart from the comparisons of ML with other ML techniques, we have also designed a research question which focuses on the performance comparison of ML techniques with non-ML techniques. Similarly, we have considered a research question to investigate which combinations of ML and non-ML techniques provide better results in terms of effort estimation accuracy than single ML or non-ML techniques. Indeed, the previous SLR[15] by Wen et al has highlighted that this aspect should be further and deeper investigated since combining two or more ML/non-ML techniques may enhance the accuracy of the obtained estimation models.

In particular, the aim of RQ1 is to identify the list of ML techniques used in SDEE and analyzed their accuracy. RQ2 aims to identify the list of ML techniques which perform better than other ML techniques in terms of prediction accuracy in the number of studies used in the SLR. In this case, if a technique has higher accuracy, eg, lower mean magnitude of relative error[19] (MMRE) value, it is considered a better ML technique. For instance, if model A is compared with another model B and A provides a lower MMRE value in the majority of the studies, then we can say that model A outperforms model B. The aim of the third research question is to focus on the ML techniques which are characterized by better accuracy measure values with respect to non-ML techniques. RQ4 refers to the combinations of ML and non-ML techniques used for estimating software development effort. In particular, the combination of ML and non-ML techniques which allows to obtain an MMRE value lower than other estimation ML/non-ML techniques (or combination of techniques) is considered better than others. RQ5 aims to illustrate and analyze all the datasets used for software effort estimation, while RQ6 is defined to highlight the list of accuracy measures used in the studies analyzed by the SLR. Accuracy measures are an important component in SDEE studies since they allow to show how reliable a particular estimation model is when it is used to predict the effort. RQ7 has been considered to highlight the dominant journals for papers analyzing ML techniques for SDEE. The idea behind the RQ7 is to give an idea to the users which are the widely used sources for articles related to the software effort estimation using ML techniques.

2.2 | Search strategy

Our main search strategy to identify and download the studies consist of two phases:

a. Primary search
b. Secondary search

Regarding the primary search phase, we used the following steps:

1. Identify major terms from the research questions.
2. Consider synonyms and alternative terms used in step 1.
3. Search term combinations, ie, Boolean OR for synonyms and alternative spellings and Boolean AND to combine major terms.

As for the secondary search phase, we review the references to select primary studies that are missed/ignored in primary search.
The search string used is

> *Software AND (effort OR cost OR costs) AND (estimat* OR predict*) AND (machine AND learning OR "data mining" OR "artificial intelligence" or AI OR "pattern recognition" OR "case based reasoning" OR "decision tree" OR "regression tree" OR "classification tree" OR "neural net*" OR "genetic algorithm" OR "genetic program*" OR "Bayesian belief network" OR "Bayesian net*" OR "association rule*" OR "support vector machine" OR SVR OR "support vector regression" OR "support vector*."*

As done by similar systematic literature reviews,[10,15] sophisticated search terms have been included by employing alternative terms and synonyms of related terms using the Boolean operator OR and combining the main terms via the Boolean operator AND. In previous SLRs, the use of Boolean strings has allowed to got almost all the studies available in the databases, and for any missing studies, the references included in the selected studies can be exploited.[10,15] Taking into account these considerations, and suggestions of previous papers, we have also employed a search string based on the Boolean operators OR and AND including all possible synonyms. For instance, we have combined the synonyms of cost, namely effort, using the OR operator because some studies use software cost estimation and some others use software effort estimation. Same considerations were done for estimation and prediction terms. We used predict* and estimate*, which means if the words used in the individual studies are predict, prediction, predictions, and predicted, they are picked and selected by our search string. Similarly, we have combined different estimation techniques using the OR operator and at the end broken the string into substrings in order to search the studies based on the individual estimation techniques (support vector regression, neural networks, decision tree, etc). As for possible missing studies, we have used the references of the identified studies and downloaded further 25 different studies. We did not hesitate to apply this strategy since it has been successfully exploited in previous SLRs (eg, [10,15]).

The literature sources we focused on for searching and selecting our primary studies are the following:

- IEEE Xplore
- ScienceDirect
- Scopus
- Springer
- ACM Digital Library

We have decided to select these databases for retrieving studies because these are the digital libraries that are widely used in the community of software engineering.[20]

The studies we focused on are between 1991 and 2017, although some studies published in the first quarter of 2018 are also included. As mentioned earlier, the main idea to select the studies in this period of time is because in our opinion, a lot of new research has been carried out in recent years, and thus, we need to determine if it has changed the results claimed by the previous SLR[15] which analyzed the studies published from January 1991 to December 2010.

With our primary search phase, we identified about 241 relevant studies. Then, we used secondary search phase criteria, ie, taking into account the references of the selected studies. This allowed us to identify 26 additional relevant studies (missed in the initial search). Thus, based on our primary and secondary search phases, the total studies we selected for our SLR are 267. Figure 2 shows the steps used to select our studies. All these studies are searched and selected by just looking the study titles and abstracts.

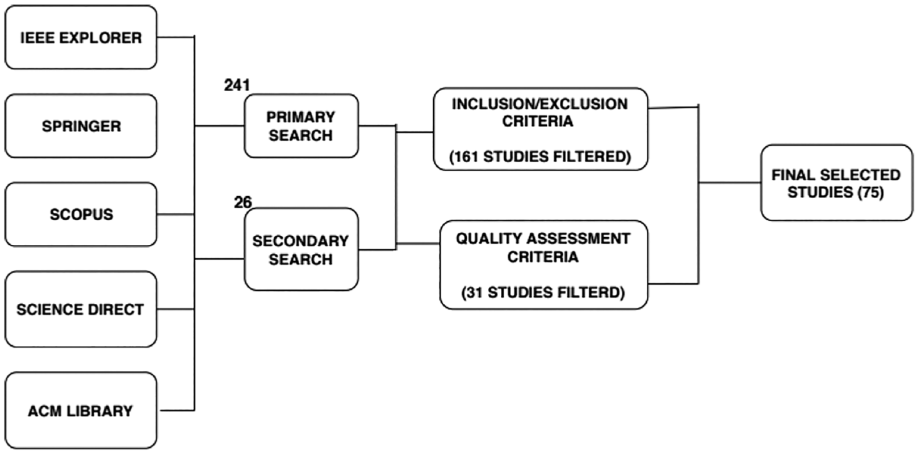

FIGURE 2 Stepwise studies selection and filtering

2.3 | Study selection

After the search and selection of our studies based on their titles and abstracts, we used two main phases to filter them and get more relevant and reliable literature. In particular, we first used our inclusion and exclusion criteria to decide which studies to include or exclude in the systematic literature review. Then, we used quality assessment criteria to further filter the selected studies. The inclusion/exclusion criteria we used are the following:

Inclusion:

- Studies that used ML techniques for software effort estimation.
- Studies that used both ML and non-ML techniques for software effort estimation.
- Studies that compare different ML techniques or studies that compare ML techniques with non-ML techniques.
- For studies that are published both in journals and conferences, only the journal version of the study was included.

Exclusion:

- Studies which focus on software effort estimation but do not use ML techniques.
- Studies employing a dependent variable different from effort/cost/budget estimation.
- Studies which focus on the estimation of software maintenance/quality/testing.
- Review studies (ie, without empirical investigations).

After carefully studying other related SLRs and several rounds of meetings, both the authors have finalized the inclusion/exclusion criteria with mutual understanding.

2.4 | Quality assessment criteria

We performed a quality assessment for each study to determine its credibility and relevance.[21] It can be considered a supplementary criterion used to select the studies in our SLR. In particular, after applying inclusion/exclusion criteria, we have exploited some quality criteria questions to weigh the obtained candidate studies. Studies having a low quality (ie, with weights less than certain thresholds) have been excluded.

To determine our quality assessment criteria, we have used the following questions:

Q1. Are the aims of the research clearly stated?

Q2. Are the estimation methods well defined and deliberated?

Q3. Is the experiment applied on sufficient dataset(s)?

Q4. Is the accuracy of estimations measured?

Q5. Is the proposed estimation method compared with other methods?

Q6. Are the limitations of the study analyzed explicitly?

Q7. How latest the publication is?

Q8. Does the study have sufficient number of average citation count?

We used fuzzy linguistic variables to score different quality assessment questions. The idea behind using fuzzy linguistic variables is that mostly systematic literature reviews have used a score of 0 to 1 for each quality assessment question. This means that a question could have 0 or 1 as value. We think it is not a fair way to weigh them. We could have a study with a fair answer (but not excellent) for a particular quality question, and so assigning it a score of 0 is not the best way to gauge. For instance, we have a quality question: "Is the experiments applied on a sufficient number of datasets?." So, if we have five datasets, we will assign a crisp value of 1 for this question but what happens if we have three datasets or even two datasets? Do we need to assign value of 0 to this question? Since the use of the fuzzy linguistic variables is a novel idea and no previous SLR has used it, we have discussed it with other researchers of similar research fields (eg, PhD students and researchers), and upon their full agreement after several meetings and discussions, we have decided to employ them. In particular, all the involved researchers agreed upon that marking the individual studies using fuzzy linguistic variables can be considered a fair way of assessment compared with the used of crisp values. Thus, for this reason, we avoided crisp values and used fuzzy linguistic variables.

In particular, we organized the scores for each question as

0	(No)
0.1-0.3	(Rarely)
0.4-0.6	(Partly)
0.7-0.9	(Mostly)
1.0	(Yes)

Since we have eight quality assessment questions, the total score for a single study can be as

0	(No)
0.1-3.1	(Rarely)
3.2-5.5	(Partly)
5.6-7.2	(Mostly)
7.3 or more	(Yes)

Again, after several rounds of discussion and careful review of the previous work about the quality assessment criteria, both the authors agreed on the use of fuzzy linguistic variable to assess the quality of the different studies selected. The studies with a score of no and rarely (as linguistic variable values) are excluded from the final list of studies because of their low scores against the quality assessment questions. As an example, suppose that a study receives the following quality assessment scores for the eight questions: Q1 (0.3), Q2 (0.1), Q3 (0.5), Q4 (0.2), Q5 (0.2), Q6 (0.5), Q7 (0.4), and Q8 (0.7). The total score is 2.9, which is less than 3.2. This means that the study cannot be included in the selected list. Differently, if we have the following scores for a particular study, it can be selected and included in our list since the total score is 3.6 that is greater than 3.2: Q1 (0.2), Q2 (0.5), Q3 (0.4), Q4 (0.7), Q5 (0.3), Q6 (0.5), Q7 (0.1), and Q8 (0.9).

As result of the application of this quality assessment, we have further discarded 31 studies. In other words, all the studies with the linguistic variable scores "partly, mostly, and yes" are included in the SLR, where the studies yes scored are regarded as high-quality studies based on our quality assessment questions. Table 1 shows all the studies with their scores.

One of our assessment questions is the average citation count for a particular study (Q8). In that case, we can imagine that the publications in the year of 2017 to 2018 have low citations as they are published more recently; thus, we have excluded the "citation count" question for the studies which are published in/and after the year 2016.

The most cited study of our SLR has more than 500 citations, while the mean and the median of citations are 85 and 44, respectively. We have decided to score the papers according to their citations as follows:

- 300 and more citations: marked as 1.0.
- 201-299 citations, marked as 0.9.
- 101-199, marked as 0.7-0.8.
- 51-100, marked as 0.5-0.6.

TABLE 1 Quality assessment scores with the linguistic variables

Study No.	Quality Assessment Score	Study No.	Quality Assessment Score	Study No.	Quality Assessment Score
S1	Partly	S29	Partly	S54	Mostly
S2	Partly	S30	Partly	S55	Mostly
S3	Yes	S31	Mostly	S56	Mostly
S4	Partly	S32	Mostly	S57	Partly
S5	Partly	S33	Mostly	S58	Partly
S6	Mostly	S34	Mostly	S59	Partly
S7	Mostly	S35	Mostly	S60	Partly
S8	Yes	S36	Mostly	S61	Partly
S9	Mostly	S37	Partly	S62	Yes
S10	Partly	S38	Partly	S63	Partly
S11	Partly	S39	Mostly	S64	Partly
S12	Partly	S40	Partly	S65	Mostly
S13	Mostly	S41	Partly	S66	Mostly
S14	Mostly	S42	Partly	S67	Partly
S15	Mostly	S43	Mostly	S68	Mostly
S16	Mostly	S44	Mostly	S69	Mostly
S17	Partly	S45	Partly	S70	Partly
S18	Partly	S46	Mostly	S71	Partly
S19	Mostly	S47	Partly	S72	Partly
S20	Partly	S48	Mostly	S73	Partly
S21	Yes	S49	Partly	S74	Partly
S22	Partly	S50	Mostly	S75	Yes
S23	Mostly	S51	Mostly		
S24	Mostly	S52	Mostly		
S25	Yes	S53	Partly		
S26	Mostly				
S27	Yes				
S28	Mostly				

- 1-50, marked as 0.1-0.4.
- citations, marked as 0.

We have decided the scores to be associated to each range taking into the number of papers included in the range. As an example, we have associated a single score value to papers in the range 201 to 299 since just four papers are included in it, while we have considered a wider range of four values for 1 to 50 since many papers are included in it. In particular, we classified 5 papers having more than 300 citations, 4 papers in between 201 and 299; 12 papers in between 101 and 200, 13 papers in between 51 and 100, and 36 papers in between 1 and 50. Just four papers are marked as 0.

As mentioned above, for the studies published in or after 2016, we have calculated the quality assessment scores considering only seven quality questions (rather than eight), ie, Q1 to Q7.

We can observe that among the 31 papers discarded by applying quality assessment criteria, the majority of them have obtained low score values for questions Q1, Q2, Q4, and Q5, ie, the aims of the research are not clearly stated, the estimation methods are not well defined, the number of datasets employed is low, the accuracy of estimations is not evaluated, or the proposed estimation method is not compared with other methods. Note that no study has been discarded mainly for the score value obtained for Q8 (ie, the number of citations).

We can highlight some of the studies to our readers which have a high impact. For instance, the studies that answer most of the research questions and have high-quality assessment values can be considered as high impact studies.

So, we have selected about 75 most relevant and rigorous studies after the filter of inclusion/exclusion criteria, quality assessment questions, and removing some duplications. References to these studies are reported in Appendix A at the end of the paper (ie, S1, S2, …, S75).

2.5 | Data extraction

One of the issues we noticed in the data extraction phase is that some studies used different synonyms for some ML models. For instance, some of the studies used the term reasoning by analogy whereas other used it as CBR. Similarly, some studies used “random forest,” some “regression tree,” and some used “M5P,” while all these can be considered as the categories of decision trees.

Table 2 shows the list of research questions addressed by the corresponding studies selected in our SLR. It is worth noting that not every selected study addresses all of our research questions.

We analyzed and assigned a score/value to each of studies shown in Table 2 in order to describe their overall credibility (quality). We scored each study based on how many research questions it addresses as well as its quality assessment score. The score ranges from 0 to 10 where 0 is the minimum value and 10 is the maximum value, a study may have. Clearly, the higher value a study has, the more credible it is. Each study gets

TABLE 2 Research questions addressed by the studies selected in our systematic literature review (SLR)

Study	Research Questions						
S1	RQ1	-	--		RQ5	-	-
S2	RQ1	RQ2	-	RQ4	RQ5	-	-
S3	RQ1	RQ2	RQ3	RQ4	-	-	RQ7
S4	RQ1	-	RQ3	-	-	-	-
S5	RQ1	-	RQ3	RQ4		RQ6	-
S6	RQ1	-	RQ3	-	RQ5	-	-
S7	RQ1	RQ2	-	-	RQ5	-	-
S8	RQ1	RQ2	-	-	RQ5	-	-
S9	RQ1	RQ2	-	-	RQ5	-	-
S10	RQ1	RQ2	RQ3	RQ4	RQ5	-	-
S11	RQ1	RQ2	RQ3	RQ4	RQ5	-	-
S12	RQ1	RQ2	-	-		RQ6	-
S13	RQ1	RQ2	-	-		-	RQ7
S14	RQ1	RQ2	RQ3	-	RQ5	-	RQ7
S15	RQ1	RQ2	-	-	RQ5	RQ6	-
S16	RQ1	RQ2	-	-	RQ5	RQ6	-
S17	RQ1	RQ3	-	-		RQ6	-
S18	RQ1	RQ2	-	-		RQ6	-
S19	RQ1	RQ2	-	-	RQ5	-	RQ7
S20	RQ1	RQ3	-	-	RQ5	RQ6	-
S21	RQ1	RQ2	-	-		RQ6	-
S22	RQ1	RQ2	-	-		-	-
S23	RQ1	RQ2	-	-		RQ6	-
S24	RQ1	RQ2	RQ3	-	RQ5	RQ6	-
S25	RQ1	RQ2	RQ3	-	RQ5	RQ6	RQ7
S26	RQ1	-	RQ3	-		RQ6	-
S27	RQ1	RQ2	RQ3	RQ4	RQ5	RQ6	RQ7
S28	RQ1	RQ2	RQ3	RQ4	RQ5	-	-
S29	RQ1	RQ2	-	-	RQ5	-	-
S30	RQ1	-	RQ3	-		-	-
S31	RQ1	RQ2	-	-	RQ5	RQ6	RQ7
S32	RQ1	RQ2	RQ3	RQ4	RQ5	RQ6	-
S33	RQ1	-	-	RQ4	RQ5	-	-

(Continues)

TABLE 2 (Continued)

Study	Research Questions						
S34	RQ1	RQ2	-	-	-	-	RQ7
S35	RQ1	RQ2	RQ3	-	RQ5	-	-
S36	RQ1	RQ2	RQ3	-	-	-	-
S37	RQ1	RQ2	RQ3	-	-	RQ6	-
S38	RQ1	RQ2	-	-	-	-	-
S39	RQ1	-	-	-	-	-	-
S40	RQ1	RQ2	RQ3	-	RQ5	-	-
S41	RQ1	RQ2	RQ3	-	-	-	-
S42	RQ1	RQ2	RQ3	-	-	-	-
S43	RQ1	RQ2	RQ3	-	-	-	-
S44	RQ1	RQ2	RQ3	-	RQ5	RQ6	RQ7
S45	RQ1	RQ2	RQ3	-	-	-	RQ7
S46	RQ1	RQ2	RQ3	-	RQ5	-	RQ7
S47	RQ1	RQ2	RQ3	-	RQ5	-	-
S48	RQ1	RQ2	RQ3	-	-	RQ6	RQ7
S49	RQ1	RQ2	RQ3	-	-	-	-
S50	RQ1	RQ2	-	-	-	-	RQ7
S51	RQ1	-	-	-	-	-	-
S52	RQ1	-	-	-	-	-	RQ7
S53	RQ1	-	-	-	-	RQ6	RQ7
S54	RQ1	RQ2	-	-	-	-	RQ7
S55	RQ1	RQ2	-	RQ4	RQ5	RQ6	RQ7
S56	RQ1	-	RQ3	-	-	-	RQ7
S57	RQ1	-	RQ3	-	-	RQ6	-
S58	RQ1	RQ2	-	RQ4	RQ5	RQ6	RQ7
S59	RQ1	RQ2	-	-	RQ5	RQ6	-
S60	RQ1	-	RQ3	-	-	-	RQ7
S61	RQ1	-	RQ3	-	RQ5	-	RQ7
S62	RQ1	RQ2	RQ3	-	-	RQ6	RQ7
S63	RQ1	-	RQ3	RQ4	-	RQ6	RQ7
S64	RQ1	-	RQ3	-	RQ5	RQ6	RQ7
S65	RQ1	-	RQ3	RQ4	RQ5	-	RQ7
S66	RQ1	-	RQ3	-	RQ5	RQ6	-
S67	RQ1	-	RQ3	RQ4	-	RQ6	-
S68	RQ1	-	RQ3	-	-	RQ6	-
S69	RQ1	RQ2	RQ3	RQ4	-	RQ6	RQ7
S70	RQ1	RQ2	RQ3	-	RQ5	RQ6	-
S71	RQ1	-	RQ3	-	-	-	-
S72	RQ1	-	RQ3	-	-	RQ6	-
S73	RQ1	-	RQ3	-	RQ5	-	-
S74	RQ1	-	RQ3	-	RQ5	-	RQ7
S75	RQ1	RQ2	RQ3	RQ4	-	RQ6	-

one point for every research question it addresses, and since we have seven research questions, there are seven points a study can get at maximum. Similarly, about quality assessment scores, if a study has the linguistic variable of partly, its score is 1, if the variable value is mostly, its score is 2, and if the value is yes, the score is 3. So, a study can get a maximum of 7 score from the research questions and 3 from the quality assessment scores. The idea behind assigning scores to each study is to recommend some quality studies to our readers, based on their overall score.

Figure 3 shows all the studies with their associated credibility scores. As the figure shows, the highly scored studies are S25, S27, S32, S44, S54, S61, and S67, whereas the lowest scored studies are S22, S30, S38, and S71, respectively.

2.6 | Data synthesis

The primary purpose of the data synthesis is to identify and group all the evidence from the selected studies that are related to our research questions. A single piece of information might have small evidence force, but the collection of many of them can make a point stronger.[22] We have mainly adopted two methods to synthesize our result data:

- Narrative: We analyze our data and use the results in the form of tables and various charts, mostly bar charts. RQ1, RQ5, RQ6, and RQ7 are analyzed in this manner.
- Vote counting: For RQ2, RQ3, and RQ4, we adopted the vote counting method in which some comparisons are made. In this case, we analyzed the accuracy measures to determine which models outperform the other models.

2.7 | Threat to validity

2.7.1 | Study selection bias

We used the search string to select the relevant studies of our SLR and have tried our best to phrase the search string according to our research questions. However, there is a possibility that we may have missed some relevant studies due to the (rare) fact that some studies did not include the important keywords in their study title, abstract, and keywords. Although we tried our best to avoid this possibility by referring to the bibliography of individual studies to select all the relevant studies, still, there is a chance we may have missed some important studies which can be considered as a threat.

2.7.2 | Subjective quality assessment

Quality assessment sets a number of criteria, and we exclude the studies that are below those criteria. In this SLR also, there is a possibility that some good-quality studies may have excluded which could be a potential threat, but again, to minimize this possibility, we have used the fuzzy linguistic variables in the quality assessment criteria to gauge the studies in membership function and exclude only those studies which are below the criteria we have set. A further threat could be related to the fact that only the two authors of the paper discussed and scored the papers selected by the SLR. However, the use of fuzzy linguistic variables and the strategy involving other researchers for their choices were also considered to mitigate this threat.

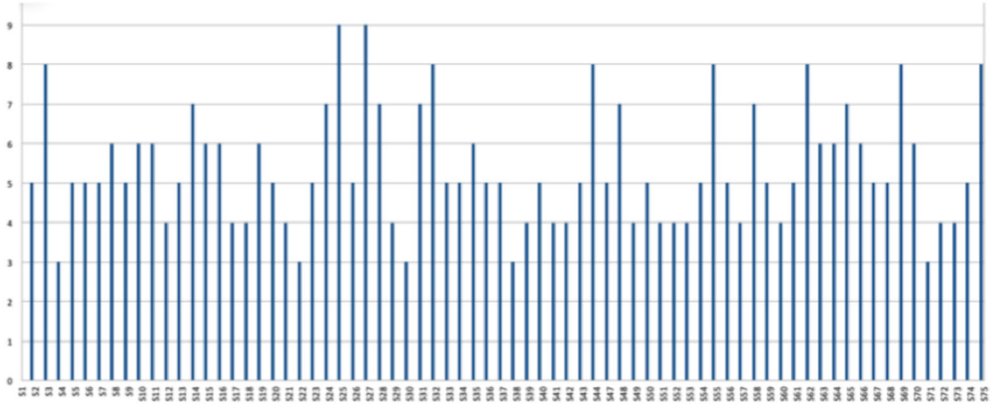

FIGURE 3 Overall credibility of the selected studies

3 | RESULTS AND DISCUSSIONS

In this section, we describe the results obtained from our primary studies for each research question.

3.1 | RQ1: Which are the ML techniques used in SDEE studies?

We analyze the ML techniques used in all the studies found by our SLR, which are listed in the following:

- Artificial neural network (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)

Figure 4 shows for each technique the number of studies that applied it. We can note that the most frequently used ML technique is ANN, which has been employed in about 45 (60%) studies. Different types of neural networks (NNs) have been used like feedforward NN, radial basis function NN, fuzzy NN, etc. The second mostly used estimation technique is SVM, investigated in about 19 (25%) different studies. CBR has been investigated in about 13 (17%) studies, CART in 9 (12%), while BN has been employed in 8 (11%) different studies. GP has been used in five (7%), and kNN has been investigated in four (5%) different studies. Finally, RF has been employed in two (3%) studies of the SLR. We consider RF and DT as one category, although some researchers categorize CART also in this category.

These ML techniques have been used either alone or in combination with other (ML and non-ML) estimation techniques to estimate the SDEE. In about 89% of studies, ML techniques have been used and compared (in terms of prediction accuracy) with the combination of other ML and non-ML techniques, whereas in 11% of studies, a particular ML approach has been compared only with non-ML approaches. ANN is the mostly used ML technique in combination with non-ML techniques, which have been used in 22 (29%) different studies. As for the combinations of techniques, GA has been mostly used for feature weighting or feature selection.

3.2 | RQ2: Which is the ML technique that outperforms other ML techniques in terms of effort estimation accuracy?

About 42% of studies used ML techniques which were also compared with other ML approaches. The ML techniques which have outperformed other ML approaches in accurately predicting the development effort (ie, characterized by lower MMRE value) are listed in Table 3. The techniques in the first column (bold) outperformed the others in the rows. As an example, ANN in the first column of the first row outperformed the other ML techniques in a greater number of experiments. For instance, it outperformed CBR in the studies S7, S56, and S69, CART in the studies S7, S21, and S31, SVM in the studies S8, S15, S23, and S58, BN in the study S29, and DT in the study S28.

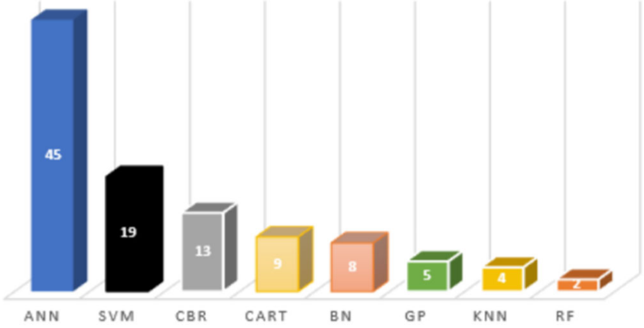

FIGURE 4 Machine learning (ML) techniques used by studies in our systematic literature review (SLR)

TABLE 3 The machine learning (ML) techniques that outperformed other ML techniques (studies between brackets)

Best ML Techniques	Outperformed ML Techniques				
ANN	CBR (S7, S56, S69)	CART (S7, S21, S31)	SVM (S8, S15, S23, S58),	BN (S29)	DT (S28)
SVM	CBR (S3, S75)	CART (S21)	ANN (S12, S15, S16, S21, S22)	BN (S3, S75)	DT (S13)
DT	CBR (S36)	SVM (S9, S13, S6)	ANN (S9, S14, S36)	BN (S39)	
CART	CBR(S7)				
GA	ANN (S15, S53)	SVM (S15)			
CBR	ANN (S7, S21)	CART (S7, S21, S62, S70)	SVM (S21)		
GP	ANN (S55)	–	–	–	--

Abbreviations: ML, machine learning; ANN, artificial neural network; SVM, support vector machine; BN, Bayesian network; CBR, case-based reasoning; GP, genetic programming; CART, classification and regression tree; DT, decision tree.

Note that a single study may include more than one experiment based on different datasets. The second technique which performed better is SVM which outperformed other techniques in 11 different experiments. DT and CBR are the next well-performed approaches, which had better performance in eight and five different experiments, respectively.

It is worth to note that some studies used the same ML techniques in more than four experiments based on different datasets. In that case, a technique is considered providing better results if its performance was better than the other technique(s) at least in two experiments. For instance, if ANN and SVM were used in five different experiments by a particular study, and ANN performed better in two experiments and SVM in three experiments, we consider both as better techniques in that study, but if, for instance, SVM performed better in four experiments and ANN in just one, we only consider SVM as performing better.

3.3 | RQ3: Do ML techniques provide better results in terms of effort prediction accuracy than non-ML techniques?

As for non-ML approaches, about 58% of studies used non-ML techniques. The commonly used non-ML methods are COCOMO, stepwise multiple regression (SWMR), multivariate adaptive regression splines (MAR splines), simple linear regression, multiple linear regression (MLR), and stochastic gradient boosting (SGB). We have included only the non-ML techniques which have been used in more than one study. MLR and COCOMO are the widely used non-ML techniques, investigated in 15 (20%) and 14 (19%) different studies, respectively. SWMR has been used in 10 (13%), linear regression in 6 (8%), and MAR splines and ordinary least regression have been both used in 5 (7%) studies, respectively. Figure 5 shows the distribution of various non-ML approaches.

In 43 (57%) different studies, combinations of both ML and non-ML approaches have been applied and then compared in terms of accuracy measures. ANN is the mostly used ML technique which has been compared with (and outperformed) 18 different non-ML techniques as shown in Table 4. Indeed, the effort predictions obtained with ANN were better in 13 different studies when compared with different regression-based models; in five studies, it outperformed the COCOMO-based model, and in two different studies, it had better performances than function point–based model. The second-best technique is CBR which achieved better effort predictions in five studies with respect to non-ML techniques; SVM also outperformed non-ML techniques in five studies; CART, BN, and GP performed better in two studies each while DT had better performances than non-ML techniques in only one study.

In 21% of studies, non-ML techniques have performed better than ML techniques. In particular, the effort estimations obtained with different regression-based models were more accurate than those provided by ML techniques. Table 5 shows that linear regression-based models outperformed ANN in five different studies, CBR in eight, BN in three, GP in two, and SVM, CART, and k-nearest in one study each.

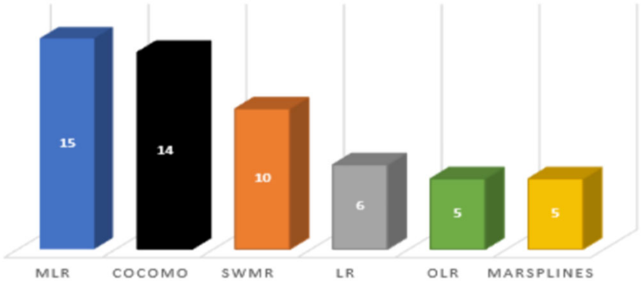

FIGURE 5 Nonmachine learning (ML) techniques used by studies in our systematic literature review (SLR)

TABLE 4 Machine learning (ML) techniques that outperformed non-ML techniques

Best ML Techniques	Outperformed Non-ML Techniques		
ANN	Linear regression (S15, S19, S23, S24, S26, S28, S31, S46, S56, S58, S63, S67, S69)	COCOMO (S4, S17, S43, S71, S73)	Function points (S61, S73)
SVM	Linear regression (S3, S15, S28, S75)	COCOMO (S30)	
GP	Linear regression (S52)	COCOMO (S30)	
CART	Linear regression (S31, S68)	-	
Decision tree	Linear regression S38	-	
BN	Mean model, median model (S1)	COCOMO (S74)	
CBR	Linear regression (S51, S54, S56, S57, S62, S66, S70)		

Abbreviations: ML, machine learning; ANN, artificial neural network; SVM, support vector machine; BN, Bayesian network; CBR, case-based reasoning; GP, genetic programming; CART, classification and regression tree; COCOMO, constructive cost model.

TABLE 5 Nonmachine learning (ML) techniques that outperformed ML techniques

Best Non-ML Techniques	Outperformed ML Techniques						
Regression-based models	SVM (S8)	ANN (S10, S24, S44, S45, S63)	BN (S39, S53, S75)	CBR (S3, S48, S49, S54, S57, S60, S64, S75)	K-nearest (S44)	GP (S45, S48)	CART (S70)

Abbreviations: ML, machine learning; ANN, artificial neural network; SVM, support vector machine; BN, Bayesian network; CBR, case-based reasoning; GP, genetic programming; CART, classification and regression tree.

3.4 | RQ4: Which combinations of ML and non-ML techniques provide better results in terms of effort estimation accuracy than single ML or non-ML techniques?

To identify which combination of ML and non-ML techniques performed better than others, we consider MMRE as it is a widely used accuracy measure. We know that 54% of studies used both combinations of ML and non-ML techniques, but only 30% of them used MMRE as accuracy measure. Also, most of these 30% of studies have used ANN-regression models or SVM-regression-based models, and most of the experiments considered Desharnais,[23] NASA,[24] and COCOMO[25] as datasets. Since we do not have sufficient experiments employing other (ML and non-ML) techniques, we consider including them in our results a potential threat to the reliability of our arguments.

Each of the combinations, ie, ANN and regression analysis and SVM and regression analysis, were performed on 10 different experiments (one study may have more than one experiments), and we have calculated the mean of their MMRE values to determine which combination performs better than the others. Of course, there may be other combinations which had better results than these, but due to the limited number of experiments, we have to exclude them. The combinations characterized by the best accuracy measures are those involving SVM and regression techniques, with an average MMRE value of 0.37. The other combinations that provided better results than others are those exploiting ANN and regression techniques, characterized by an average MMRE value of 1.73. Table 6 shows all the experiments of the combination of ANN and regression analysis and SVM and regression analysis.

3.5 | RQ5: Which are the datasets most frequently used in SDEE studies?

About 10 different datasets have been used in the selected studies. We consider here the datasets that have been used in more than one study. NASA dataset is the most widely used dataset in the SLR, which has been employed in 17 (23%) different studies. COCOMO and ISBSG datasets

TABLE 6 Combinations of machine learning (ML) and non-ML techniques that provided better results

Datasets	ANN-Regression (MMRE)	SVM-Regression (MMRE)
Desharnais	0.40-0.61, 0.31-0.59	0.16-0.18, 0.36-0.59, 0.47-0.61
NASA	0.19-0.18, 0.20-0.17, 0.31-0.61, 0.32-0.79, 0.16-0.30, 0.23-0.21, 0.18-0.23	0.16-0.17, 0.40-0.59, 0.16-0.18, 0.20-0.21, 0.19-0.23
COCOMO	0.16-0.31, 25.19-47.92	0.17-0.21, 0.22-0.26

Abbreviations: MMRE, mean magnitude of relative error; MdMRE, median of magnitude of relative error; ANN, artificial neural network; SVM, support vector machine; BN, Bayesian network; CBR, case-based reasoning; COCOMO, constructive cost model.

are the second widely investigated datasets, which have both recorded equally in 16 (21%) studies, followed by the dataset Desharnais, which has been used in 14 (19%) studies. Kemerer has been used in eight (11%) studies while Maxwell and Albrecht datasets have been both used in seven (9%) studies. Tukutuku dataset has been used in four (5%), Finnish datasets has been used in three (4%) different studies, while Miyazaki has been investigated only in two studies. Figure 6 summarizes their use.

3.6 | RQ6: Which are the accuracy measures widely used in SDEE studies assessing ML techniques?

Accuracy measures are important components of our SLR. They are used to highlight how reliable a particular effort estimation model (built with ML and/or non-ML techniques) is when it comes to predicting the software development effort. Several accuracy measures have been used in the studies selected by our SLR. Among these measures, MMRE and Pred(25) have been widely used, which are based on magnitude of relative error (MRE). Another measure based on MRE is median of MRE (MdMRE). Of course, the lower the MMRE and MdMRE values are, the more accurate a particular estimation model is. Differently, measuring the percentage of predictions that are within 25% of the actual value, in the case of Pred(25), the higher the percentage, more accurate the predicted values of the model are. Other accuracy measures employed in the selected studies are mean absolute error (MAE), root mean squared error (RMSE), mean of magnitude of error relative to the estimate (MMER), and mean square error (MSE). The definitions of all these accuracy measures are reported in Appendix B.

As for the use of these accuracy measures, from Figure 7, we can note that MMRE is the mostly used one since 52 (69% of all) different studies employed it, followed by Pred(25) which was used in 46 (61% of all) studies. MdMRE results is used in 24 (32% of all) studies, while MAE[26] in 12 (16% of all) studies, and RMSE[27] in 7 (9% of all) of studies. The remaining measure, ie, MMER[28] and MSE,[27] result is employed in six (8% of all) and four (5% of all) studies, respectively.

Table 9 in Appendix B shows all the MMRE, MdMRE, and Pred(25) values of the ML models. We have selected MMRE, MdMRE, and Pred(25) because these three accuracy measures are widely recognized and used by the scientific community of software effort estimation. In this case, we have included the values recorded in all the selected studies of our SLR.

3.7 | RQ7: Which are the dominant journals for papers analyzing ML techniques for SDEE?

Table 7 shows the dominant journal/conferences where the studies selected in our SLR were published. The idea behind this is to let users know about the main sources of the articles related to software effort estimation using ML techniques.

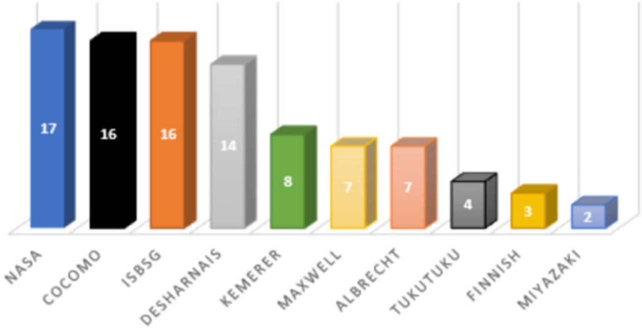

FIGURE 6 Datasets employed by studies selected in our systematic literature review (SLR)

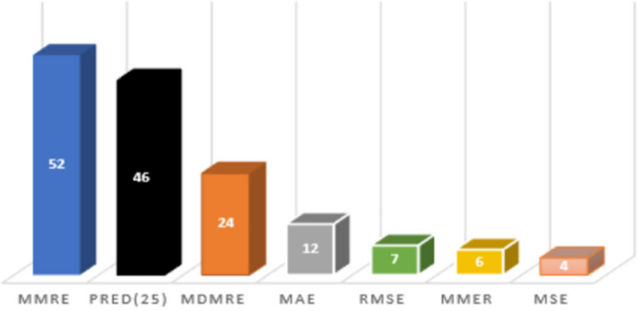

FIGURE 7 List of accuracy measures used in the studies of our systematic literature review (SLR)

TABLE 7 Dominant journals/conferences about software development effort estimation (SDEE) using machine learning (ML) techniques

Publication Name	Type	Number	Percent
Journal of Systems and Software	Journal	9	12%
Information and Software Technology	Journal	9	12%
IEEE Transactions on Software Engineering	Journal	4	5%
Empirical Software Engineering	Journal	3	4%
International Conference on Education Technology and Computer	Conference	3	4%
International Conference on Swarm, Evolutionary, and Memetic Computing	Conference	2	3%
International Conference on Big Data, Cloud Computing, Data Science & Engineering	Conference	2	3%
Procedia Computer Science	Journal	2	3%
International Conference on Predictive Models in Software Engineering	Conference	2	3%

We can observe that about 57% of the studies have been published in journals while the remaining in conference proceedings (ie, 43%). Journal of System and Software and Information and Software Technology are the two leading journals that have published the great number of studies selected in our SLR (ie, nine each). These two journals have published about 24% of all the selected publications that investigated ML techniques. The next two major (journal) publication sources are IEEE Transactions on Software Engineering and Empirical Software Engineering journals that have published four and three studies selected by our SLR, respectively. Among the conferences, the International Conference on Education Technology and Computer is the leading one, which has published three studies, ie, about 4%, while International Conference on Swarm, Evolutionary, & Memetic Computing, International Conference on Big Data, Cloud Computing, Data Science & Engineering, and International Conference on Predictive Models in Software Engineering have published two studies each. We consider here only those journals/conferences which have published more than one studies.

4 | SUMMARY OF FINDINGS

We have presented the results of an SLR related to software effort estimation using ML techniques. We selected a total of 75 studies, extracted from IEEE Xplore, Springer, ScienceDirect, ACM Digital Library, and Scopus databases between the period of January 1991 and December 2017, although some studies are also included from the first quarter of 2018. The last detailed SLR about software effort estimation using ML techniques has been published in 2012,[15] which included the studies in between January 1991 and December 2010, and we do believe that there is a need of a new analysis which covers this gap of 7 to 8 years. Indeed, the idea behind conducting this further SLR is to identify if there is a substantial amount of work done after the last SLR was published, and we found that 34 out of 75 selected studies were conducted and presented after the last SLR publication. Figure 8 shows the distribution of different studies from 1991 to 2018.

To highlight the contributions of our SLR, we summarize the achieved findings taking into account also those provided by the previous SLR presented by Wen et al.[15] This allows also to perform a detailed comparison with the previous SLR, which is summarized in Table 8.

The selected studies of the previous SLR are 84 while our SLR has selected 75 studies. The main reason of this difference is because Wen et al have used seven digital libraries while we have considered only five libraries to select the studies. However, many studies included in the SLR of Wen et al come from Google Scholar and an online bibliographic library (BESTweb), which we did not consider because we have identified some studies of very low quality from these digital libraries. We believe that, if the number of studies about a particular domain is already big in numbers (from well-known and good-quality sources, like ScienceDirect, IEEE Explore, etc), we can refrain using sources like Google Scholar and/or private online libraries.

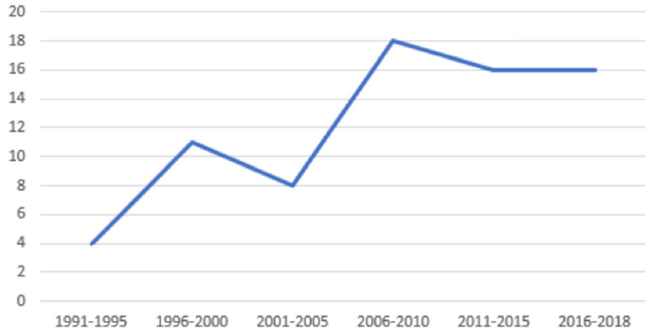

FIGURE 8 Distribution of different studies of our systematic literature review (SLR)

TABLE 8 Summary of the comparison with the systematic literature review (SLR) results presented by Wen et al[15]

Object of the Comparison	Our SLR	Previous SLR
Top 3 widely used ML techniques	ANN (60%), SVM (25%), CBR (17%)	CBR (37%), ANN (26%), DT (17%)
Widely used non-ML techniques	Regression models	Regression models
Top 3 ML techniques that performed better than non-ML techniques	ANN, CBR, and SVM	CBR, ANN, and DT
Non-ML technique that performed better than ML techniques	Regression approach (26%)	Regression approach (36%)
Top 3 widely used datasets	NASA (22%), COCOMO (21%), ISBSG (21%)	Desharnais (28%), COCOMO (22%), ISBSG (20)
Top 3 widely used accuracy measures	MMRE (69%), Pred(25) (61%), MdMRE (31%)	MMRE (89%), Pred(25) 65%, MdMRE (31%)

Abbreviations: MMRE, mean magnitude of relative error; MdMRE, median of magnitude of relative error; ANN, artificial neural network; SVM, support vector machine; BN, Bayesian network; CBR, case-based reasoning; COCOMO, constructive cost model.

As in the case of the considered research questions (see Section 2.1), there is not a big difference between the quality assessment questions of the two SLRs. However, the way the individual studies are scored and evaluated based on those questions has a significant difference. Unlike the SLR of Wen et al, we have preferred to use fuzzy linguistic variables to gauge and assess the studies, which helps in excluding only those studies which are very low in the quality. In particular, we have excluded 31 studies based on our quality assessment questions.

Analyzing the results in Table 8, we can observe that Wen et al found that the three widely used ML models in their SLR are CBR, ANN, and DT, employed in 37%, 26%, and 17% of the selected studies, respectively. The three mostly used ML techniques in our SLR are ANN, SVM, and CBR, used in 60%, 25%, and 17% of the selected studies, respectively. It means that in recent studies (published after 2010), ANN has been used widely, which is reflected in its increase from 26% to 60%. Also, CBR which resulted to be the most frequently used ML technique in the previous SLR is investigated only in 17% of studies in our SLR, reflecting its limited use in the recent studies. On the other hand, SVM is the ML technique which is used mostly in recent studies and hence results to be in the top three widely used ML techniques of our SLR.

The percentage of usage of ANN is almost twice than the second widely investigated estimation technique (SVM), and the reasons could be the variety of flavors ANN can make available. For instance, in our SLR, ANN is used as multilayer perceptron (13 studies), radial basis function neural network (7 studies), backpropagation neural network (3 studies), fuzzy neural network (2 studies), and wavelet neural network (2 studies). Furthermore, ensemble of neural networks and general regression neural networks has also been considered. Although, in most of the studies different types of ANN are used, like MLP and RBFNN, researchers have discussed motivations and advantages of applying general ANN, such as it is able to model complex relationship between the dependent and independent variables and it is less sensitive to noise and generalizable, thus allowing to infer relationships on unseen data. Furthermore, studies have shown that when some preprocessing steps are performed before ANN is applied, such as removing the noise and filtering the irrelevant features, prediction accuracy of ANN has been recorded as best. In particular, ANN has been employed with techniques like fuzzy systems and feature selection algorithms (eg, genetic algorithms).

As for the analysis performed to identify which combination of ML and non-ML techniques performed better than others, we have considered just the results in terms of MMRE in order to perform a fair comparison among different approaches. So, we have carried out the comparison taking into account a few experiments (10) based on COCOMO, Desharnais, and NASA datasets, where SVM combined with regression analysis has just one competitor, ie, ANN combined with regression analysis. SVM and regression resulted to be the best combination. However, we want to highlight that there is just one experiment, with the COCOMO dataset, in which the ANN & regression analysis obtained higher MMRE values (25.19-47.92), and due to this result, the overall MMRE value considering all the experiments become higher (and so worse) than the one obtained with SVM and regression analysis. So, we think we cannot provide a general result from this point of view and further analysis should be performed. We can just highlight that since ANN and SVM are the most widely used ML techniques, they have also been exploited in combination with simple regression analysis and allowed to obtain better effort predictions in the few cases considered.

About the datasets used in all the studies, the top three widely used datasets in Wen et al's SLR are Desharnais (28%), COCOMO (22%), and ISBSG (20%), while the three mostly used datasets in our SLR are NASA, employed in 23% of the studies, and COCOMO and ISBSG, both used in 21% of the selected studies. Thus, COCOMO and ISBSG are common in mostly used datasets of the both SLRs, although there is a difference in their percentage of usage. Differently, Desharnais is the mostly used dataset in the previous SLR, while in our case, it is not in the list of the three widely used datasets. In particular, it is replaced by the NASA dataset (which was not in the top 3 of the previous SLR), used frequently in our case (23%). This means that it has been employed in many studies recently. About these widely used datasets, there is no detail given, even by the authors of the primary studies, about the reasons COCOMO or NASA datasets are selected by them. Surprisingly, the authors just mention the statistical details about the employed variables, such as their maximum, minimum, mean, median, and standard deviation, and no justifications about their choice are provided. Probably, the only reason is that they have employed datasets publicly available (eg, in the PROMISE repository) and used in the past by other researchers, so that possible comparisons can be accomplished.

TABLE 9 Mean magnitude of relative error (MMRE), median of magnitude of relative error (MdMRE), and Pred(25) of machine learning (ML) models in all studies

Techniques	Studies ID	MMRE	Pred(25)	MdMRE	Studies ID	MMRE	Pred(25)	MdMRE
SVM	S3	1.45	0.08	0.81	S15	0.405	72.22	-
	S3	1.25	0.17	0.73	S15	0.36	66.67	-
	S3	0.81	0.17	0.85	S15	0.177	94.44	
	S3	2.08	0.08	0.85	S15	0.16	94.44	-
	S8	0.473	55.56	-	S15	0.16	88.89	-
	S9	23.11	77	7.53	S16	0.13	76/91	-
	S9	23.43	75.75	8.71	S21	0.45	0.25	0.43
	S9	33.28	13.77	82	S21	0.40	0.37	0.37
	S9	33.39	73.81	15.66	S21	0.09	0.98	0.077
	S9	19.63	80.50	5.17	S22	435.69	-	-
	S9	22.59	79.25	7.02	S34	-	41.67	-
	S12	0.93	-	-	S50	1.15	0.42	-
	S12	0.948	-	-	S50	1.79	0.52	-
					S50	0.66	0.40	-
					S58	0.18	-	83.33
					S75	0.590	0.391	0.339
					S75	0.856	0.40	0.455
ANN	S4	0.413	40	-	S28	47.66	38	33.17
	S4	0.46	50	-	S28	54.37	32.67	36.91
	S7	1.0	0.56		S28	85.44	24	61.41
	S7	1.43	0.11		S28	39.88	55	29.10
	S7	0.18	0.81		S28	49.82	30.8	44.13
	S7	0.14	1.0		S31	0.32	0.31	0.24
	S7	0.10	0.83		S31	0.38	0.43	0.33
	S7	0.57	0.22		S31	0.42	0.44	0.38
	S7	0.42	0.33		S43	0.22	0.75	0.12
	S8	0.40	66.67		S44	0.025	1	0.0015
	S9	109.96	34	65.75	S44	0.011	1	0.00058
	S9	20.19	75.25	11.78	S44	0.000026	1	0.000086
	S9	91.73	14.62	75.37	S44	0.0064	1	0.011
	S9	12.15	88.33	7.44	S44	0.00016	1	0.0011
	S9	82.82	32.0	66.60	S44	0.00035	1	0.00093
	S9	18.40	79.50	8.31	S44	0.000090	1	0.00010
	S12	1.22	-	-	S44	0.0053	1	0.0042
	S12	0.95	-	-	S44	0.0013	1	0.0023
	S12	1.312	-	-	S44	0.011	1	0.00067
	S12	0.96	-	-	S44	0.000026	1	0.000086
	S14	0.686	38.25	-	S44	0.0079	1	0.011
	S15	0.31	72.22	-	S47	0.32	0.83	0.31
	S15	0.19	94.44	-	S47	0.28	0.86	0.30
	S16	0.21	64.65	-	S47	0.33	0.83	0.37
	S21	0.49	0.25	0.51	S47	0.30	0.72	0.26
	S21	0.57	0.22	0.43	S48	0.52	-	-
	S21	0.088	0.99	0.077	S48	0.51	-	-
	S22	71.58	-	-	S48	0.52	-	-
	S23	0.19	0.66	0.16	S49	68.85	26.67	-
	S23	0.23	0.56	0.17	S50	4.44	0.12	-
	S23	0.70	0.10	0.36	S50	2.12	0.32	-
	S23	0.70	0.14	0.58	S50	2,03	0.08	-
	S23	0.75	0.33	0.61	S56	60.63	-	56
	S24	-	45.7	-	S58	0.20	-	72
	S24	-	60.5	-	S59	0.1870	-	72.22
	S24	-	28.1	-	S59	0.1881	-	72.22
	S25	-	0.80	-	S63	1.32	-	0.50
	S25	-	0.61	-	S67	0.44	-	0.63
	S26	36.47	-	-	S69	0.70	0.36	0.10
	S26	46.52	-	-	S69			
	S26	37.95	-	-				
	S26	25.19	-	-				

(Continues)

TABLE 9 (Continued)

Techniques	Studies ID	MMRE	Pred(25)	MdMRE	Studies ID	MMRE	Pred(25)	MdMRE
BN	S1	34.26	33.33	27.42	S75	7.65	7.69	1.67
	S3	7.65	0.08	1.67	S75	4.09	0.02	0.96
	S3	1.90	0.15	0.86	S75	1.90	0.15	0.86
	S39	68.94	39.5	35.49	S75	27.95	0.09	5.31
	S39	132.69	22.58	64.44				
	S39	163.53	19.35	67.74				
	S41	7.65	0.08	1.67				
	S41	1.90	0.15	0.86				
CBR	S35	49.0	29.1	42.7	S66	62	-	33
	S35	26.6	50.0	22.6	S66	64	-	36
	S35	37.4	73.3	15.8	S69	1.25	0.58	0.19
	S35	32.3	45.5	26.8	S70	0.30	0.20	0.58
	S41	5.27	0.08	0.97	S70	1.14	0.70	0.17
	S41	5.06	0.11	0.87	S75	5.27	0.08	0.97
	S52	0.72	0.35	0.42	S75	4.46	0.08	0.92
	S57	14.65	11.9	88.10	S75	5.06	0.11	0.87
	S60	198.8	-	-	S75	6.73	0.15	0.89
	S62	13.3	-	87.57	S75	5.63	0.09	0.97
	S64	1.91	-	0.18	S75	6.0	0.09	0.84

Abbreviations: MMRE, mean magnitude of relative error; MdMRE, median of magnitude of relative error; ANN, artificial neural network; SVM, support vector machine; BN, Bayesian network; CBR, case-based reasoning.

The widely used accuracy measures in both SLRs are MMRE (89% in the Wen et al's SLR and 69% in our SLR), Pred(25) (65% in the Wen et al's SLR and 61% in our SLR), and MdMRE (31% in the Wen et al's SLR and 32% in our SLR). It means that both the SLRs have common (frequently used) accuracy metrics, but their percentages of usage in our SLR are slightly less than the one of previous SLR, except for MdMRE. It is worth noting that, even if the use of MMRE and related measures has been criticized by several researchers since 2001[29] (since they wrongly prefer a model that consistently underestimates[30]), these measures continue to be used to evaluate the accuracy of effort predictions. Probably, the reason is that they represent the first employed measures and they have been widely used in the past, and researchers continue to apply them since this allows to also accomplish comparisons with previous studies.

Regarding the comparison of the performances of the different ML techniques, CBR, ANN, and DT are the ones that performed better in the Wen et al's SLR, while in our SLR, the best performed ML techniques are ANN, SVM, and DT. In particular, the performances of CBR in the Wen et al's SLR, when compared with the other ML techniques, were better in 26 different studies, while the best prediction models recorded in our SLR were obtained with ANN since it outperformed other techniques in 12 different studies. We can also note that ANN is the second-best performed technique in the previous SLR, which performed better in 22 studies in the Wen et al's SLR, while in our SLR, SVM is the second-best performed technique which has better performance in 11 studies compared with the other ML techniques. Similarly, DT is the third better performed ML models in both SLRs, obtaining better performances in nine studies in the previous SLR and in eight studies in our SLR.

In both the SLRs, among the non-ML techniques employed, regression methods are the widely used and better performed ones. About the comparison between ML and non-ML techniques, CBR resulted to be the best performed approach in the previous SLR since it outperformed the others in 25 studies, while ANN (which is the better performed ML technique in our SLR) outperformed the other non-ML techniques in 20 studies. In previous SLR, ANN is the second-best technique, outperforming non-ML techniques in 22 studies, while in our case, the second-best technique is CBR that provided better predictions in 7 studies.

On the other hand, in both SLRs, regression-based models are the only ones among those obtained employing non-ML techniques which provided predictions better than those achieved using ML techniques. Indeed, regression-based models provided better results than ML techniques in 30 studies in the previous SLR, while in our SLR in 17 studies.

Furthermore, we want to highlight that in our SLR, we formulated further research questions with respect to the previous SLR presented by Wen et al,[15] with the aim of investigating which combination of ML and non-ML techniques performed better, in terms of prediction accuracy measured with MMRE. We selected only the combinations of ANN and regression-based models and SVM and regression-based models because we have sufficient number (10) of experiments in these combinations on three different datasets, ie, Desharnais, NASA, and COCOMO. We analyzed all the 10 experiments and found that the combinations characterized by the best accuracy measures are those involving SVM and regression techniques, with an average MMRE value of 0.37. The combinations of ANN and regression technique obtained on average an MMRE value of 1.73. The idea behind including this information (it is missing in the previous SLR) is that both the SLRs considered research questions focusing on ML techniques with the aim of highlighting the ones performing better than others and also verifying whether they provide better predictions with respect to non-ML techniques. So, to provide a more complete view about the use and comparison of ML and non-ML techniques, we considered an additional research question with respect to the previous SLR presented by Wen et al[15] in order to analyze their combinations for building prediction models and assess their prediction accuracy.

The three dominant journals (which published more studies) of the previous SLR are Information and Software Technology (IST), IEEE Transactions on Software Engineering (TSE), and the Journal of System and Software (JSS) which together published about 41% of studies. Also in our SLR, the three dominant journals are JSS, IST, and TSE, which together cover about 29% of selected studies. Thus, we can observe that JSS and IST are the two journals which published a wide range of studies in both SLRs. For instance, JSS has published about 11% of studies in the previous SLR and 12% of studies in our SLR, so it means there was a 1% increase in the number of studies in the past few years. Similarly, IST has published about 16% of studies in the previous SLR and again about 12% of studies in our SLR, so we can infer that a fewer number of studies in the last 7 to 8 years have been published.

Apart from the major similarities and differences between the two SLRs, we want also to discuss some minor differences. For instance, kNN is the ML technique which has been employed in four studies selected by our SLR, while no study selected by previous SLR investigated it, and hence, we can conclude that kNN is used in studies published recently. Also, in our analysis, we clearly mention that in some studies, non-ML techniques outperformed ML techniques providing details. In particular, CBR (in seven studies), ANN (in five studies), BN (in two studies), SVM (in one study), GP (in one study), and CART (in one study) were outperformed by regression techniques. Differently, previous SLR only discussed it at abstract level by just mentioning that regression techniques are the leading non-ML techniques which outperformed ML techniques in some studies. Furthermore, our SLR has revealed that the well-known feature selection algorithms figured out are genetic algorithms, particle swarm optimizations, and Tabu search, while previous SLR by Wen et al just mentioned the name of genetic algorithms. One of the reasons of mentioning only genetic algorithm is that the bio-inspired feature selection algorithms have started to be investigated in the domain of SDEE in late 2000, ie, 2008 to 2010.

The quality of individual studies investigated in both SLRs are also important and need to be highlighted. For instance, in the studies published in the 1990s, the authors just used to select a dataset and an estimation technique (ML/non-ML) and provided the resulted accuracy values, but the majority of recent studies prefer to use some preprocessing steps, remove noise, deal with the uncertain and imprecise information, and select the subset of features that are most relevant in the datasets. The qualities of individual studies in an SLR are assessed and analyzed by exploiting quality assessment criteria, and low-quality studies are excluded from the SLR, but still we believe that the quality level of individual studies selected in the SLR need to be pointed out. In our SLR, we have made an attempt to achieve this by marking the selected studies from 1 to 10, where 1 means the lowest quality and 10 means the top quality. The marking is based on the quality assessment score an individual study received and the number of research questions a particular study managed to address. The primary objective for providing these details is to give an idea and recommend our readers about high-quality studies in the domain of software effort estimation. This is the sort of information which missed in the previous SLR.

Finally, in a nutshell, we want to highlight that the primary purpose of previous SLR was to analyze all the possible studies discussing ML-based software development effort estimation models published between 1991 and 2010, while our SLR has analyzed the studies published in the time period selected by the previous SLR as well as the studies published after with the aim of determining

- What are the recent priorities of researchers, taking into account the estimation techniques/datasets/accuracy metrics they employed in their studies.
- The work done in the past 7 years and how it changed the results claimed by the previous SLR, in terms of estimation techniques/datasets/accuracy metrics.
- If the prediction accuracy of the employed estimation techniques has improved in the last 7 years, taking into account the recent use of bio-inspired feature selection algorithms, such as particle swarm optimizations and Tabu search that has increased the performance of various estimation techniques (not considered in the previous SLR).

4.1 | Suggestions for researchers

Taking into account the above considerations, we want to highlight some findings that could be exploited in the future by researchers to carry out further investigations in the domain of SDEE using ML methods.

Studies about software effort estimation should use a more number of different ML techniques. Estimation techniques like Naïve Bayes, association rules, random forest, classification, and regression tree are either never used or investigated only in a limited number of studies. Thus, we encourage researchers to employ and further evaluate these techniques rarely used in the domain of SDEE.

Also, there is a variety of different ANN-based strategies which are used in several studies in the context of SDEE. However, it is still not clear which of these alternatives has to be applied taking into account the type and size of the datasets employed as well as the accuracy measures applied for evaluating the prediction accuracy. Among the different ANN-based strategies, MLP is widely used, but when compared with other ANN strategies, like feed forward neural network, the performances of MLP are not recorded as best. For instance, in study S46 of our SLR, the authors compared multilayer perceptron, general regression neural network, radial basis function neural network, and cascade correlation neural network and the best performed technique (in terms of MAE criterion) is cascade correlation neural network, but surprisingly, it is one of the least used among ANN strategies. Therefore, we encourage researchers to investigate cascade correlation neural network further in the future.

The accuracy of the software development effort estimations increases when the most relevant features of the datasets are selected or the features that may negatively affect the predictions are excluded. For this reason, recently, the use of bio-inspired feature selection algorithms has gained much attention in various domains, but still, it is scarce in the context of SDEE. Only genetic algorithm, particle swarm optimizations, and Tabu search have been used in some studies selected in our SLR. However, we believe that further investigations are needed to assess the effectiveness of these techniques, and other algorithms should be considered. Some of these are ant colony optimizations, wolf search, cuckoo search, harmony search, firefly search, and bee search which have already provided interesting results in a variety of domains.[31,32]

High-quality and detailed project datasets in the domain of SDEE play a crucial role in evaluating and validating the various estimation techniques. After performed our SLR, we realized that many of the considered datasets do not have great number of projects/observations (eg, Albrecht, COCOMO, Finnish, Kemerer, Maxwell, NASA). Moreover, some other datasets are not easily or publicly available. To overcome this problem, researchers should share and openly provide their project datasets, for example, through the PROMISE repository which is widely known in the SDEE community. There are also some project datasets like Canadian Financial (CF) organization dataset, SoftLab Data Repository dataset (provided by a Turkish software company), IBM Data Processing Services (DPS) dataset, and a dataset from a Taiwan software company, used in one or two studies of our SLR, that should be further investigated in the future.

Even if ML techniques have been used and compared with non-ML techniques in about 54% studies of our SLR, the majority of these studies have exploited as non-ML technique regression-based techniques, ie, SWMR, MAR splines, simple linear regression, and MLR.[33] We believe that there are some other non-ML techniques which are rarely used by researchers and should be investigated also in combination with ML techniques, eg, COCOMO-based models.

5 | CONCLUSION AND FUTURE WORK

We have presented the results of an SLR about software effort estimation using ML techniques covering the past one decade. The number of selected studies is 75, which we evaluated and analyzed to fetch important information for users. Indeed, even if a recent SLR[15] was published in 2012, which included the studies from 1991 to 2010, we believe that a substantial amount of work has been done in the last 8 years about software effort estimation using ML methods. Furthermore, there are differences between the designs of the performed SLRs. In particular, in the previous SLR by Wen et al,[15] five research questions were considered while we addressed these five questions plus some additional ones. As for the results and comparison, we observed that the most frequently used ML approaches highlighted in the previous SLR[15] are CBR, ANN, and DT, while in our SLR, we have found that the mostly used ML methods are ANN and SVM. Regression techniques (as non-ML approaches) result to be widely used in both SLRs. As for the comparison among ML-based models (in terms of efforts estimation accuracy), CBR and ANN provided better results in the previous SLR by Wen et al,[15] while in our SLR, ANN and SVM performed better than the other ML approaches. With regard to the comparison among ML and non-ML approaches, again Wen et al found that CBR and ANN performed better when compared with the investigated non-ML approaches, while in our study, also ANN and CBR are the two notable ML approaches characterized by better performances than other estimation techniques. In both SLRs, MMRE and Pred(25) are the frequently used accuracy measures. As for the datasets employed in the analyzed studies, the widely used datasets found in our SLR are COCOMO, NASA, and ISBSG while the mostly used datasets highlighted in the previous SLR are Desharnais, COCOMO, and ISBSG.

As for the future work, we intend to perform a further SLR about the bio-inspired feature selection algorithms which are used in the area of software effort estimation, and in particular when combined with the use of ML techniques. Indeed, from the literature, we have found that genetic algorithm (GA) is also used in some studies as well as particle swarm optimization (PSO) and Tabu search. But, we do believe that there are a variety of other (bio-inspired) feature selection algorithms like ant colony optimization (ACO), artificial bee colony (ABC), and firefly algorithms that have been used recently in a few studies addressing the software effort estimation problem.

ORCID

Asad Ali https://orcid.org/0000-0001-7465-1090
Carmine Gravino https://orcid.org/0000-0002-4394-9035

REFERENCES

1. Rastogi H, Dhankhar S, Kakkar M. A survey on software effort estimation techniques. In Confluence the Next Generation Information Technology Summit, 2014; 826-830.

2. Braga PL, Oliveira AL, Meira SR. Software effort estimation using machine learning techniques with robust confidence intervals. In Hybrid Intelligent Systems, 2007; 352-357.
3. Mendes E. Improving software effort estimation using an expert-centered approach. In International Conference on Human-Centred Software Engineering Springer, Berlin, 2012; 18-33.
4. Fenton N, Bieman J. *Software metrics: a rigorous and practical approach*. CRC press; 2014.
5. Popli R, Chauhan N. Cost and effort estimation in agile software development. In Optimization, International Conference on Reliabilty, and Information Technology (ICROIT), 2014; 57-61
6. Jørgensen M. A review of studies on expert estimation of software development effort. *J Syst Softw*. 2004;70(1-2):37-60.
7. Jorgensen M. Practical guidelines for expert-judgment-based software effort estimation. *IEEE Software*. 2005;22(3):57-63.
8. Mitchell TM. *Machine Learning*. Maidenhead, UK: McGraw Hill; 1997.
9. Witten IH, Frank E. *Data Mining: Practical Machine Learning Tools and Techniques*. USA: Elsevier; 2005; 1-525.
10. Jorgensen M, Sheperd M. A systematic review of software development cost estimation studies. *IEEE Trans Softw Eng*. 2007;33(1):33-53.
11. Chavoya A, Cuauhtemoc LM, Meda-Campaña ME. Software development effort estimation by means of genetic programming. *Int J Adv Comput Sci Appl*. 2013;4(11):1-414.
12. Prabhakar MD. Prediction of software effort using artificial neural network and support vector machine. *Int J Adv Res Comput Sci Softw Eng*. 2013;3(3):1-525.
13. Aljahdali S, Alaa FS, Narayan CD. Estimating software effort and function point using regression, support vector machine and artificial neural networks models. In AICCSA, 2015;1-8.
14. Idri A, Elyassami S. A fuzzy decision tree to estimate development effort for web applications. *Int J Adv Comput Sci Appl Spec Issue Artif Intell*.
15. Wen J, Shixian L, Zhiyong L, Yong H, Changqin H. Systematic literature review of machine learning based software development effort estimation models. *Inf Softw Technol*. 2012;54(1):41-59.
16. Malhotra R. A systematic review of machine learning techniques for software fault prediction. *Appl Soft Comput*. 2015;27:505-518.
17. Kitchenham, B. Procedures for performing systematic reviews. Keele, UK, Keele University, 2004; 33: 1-26.
18. Kitchenham B. Charters. Guidelines for performing systematic literature review in software engineering. Keele, UK, Keele University Version 2.3; 2007
19. Conte SD, Dunsmore HE, Shen VY. *Software engineering metrics and model*. Redwood City, CA: Benjamin-Cummins Pub Co, Inc; 1986.
20. Turner M. Digital libraries and search engines for software engineering research: An overview. Keele University, UK, 2010.
21. Higgins JPT, Green S. Handbook for systematic reviews of interventions. version 5.1. 0, The Cochrane Collaboration; 2011.
22. Lawrence S. Soup or art? The role of evidential force in empirical software engineering. IEEE Software 2005; 66-73.
23. Papatheocharous E. Software cost modelling and estimation using artificial neural networks enhanced by input sensitivity analysis. *J Univ Comput Sci*. 2012;18(14):2041-2070.
24. Sheta AF, Al-Afeef A. Software effort estimation for NASA projects using genetic programming. *J Intell Comput*. 2010;1(3):147.
25. Boehm BW. *Software Engineering Economics*. Englewood Cliffs, NJ: Prentice-Hall; 1981.
26. Langdon WB, Dolado J, Sarro F, Harman M. Exact mean absolute error of baseline predictor, MARP0. *Inf Softw Technol*. 2016;73:16-18.
27. Kantor JM, Richmond CD, Bliss DW, Correll B. Mean-squared-error prediction for Bayesian direction-of-arrival estimation. *IEEE Trans Signal Process*. 2013;61(19):4729-4739.
28. Kumar TM, Jayaram MA. Comparison of hard limiting and soft computing methods for predicting software effort estimation: in reference to small scale visualization projects. *Int J Eng Technol*. 2018;7(4.6):291-295.
29. Kitchenham B, Pickard L, MacDonell S, Shepperd M. What accuracy statistics really measure. *IEE Proc Softw*. 2001;148(3):81-85.
30. Myrtveit I, Stensrud E. Validity and reliability of evaluation procedures in comparative studies of effort prediction models. *Empir Softw Eng*. 2012;17(1-2):23-33.
31. Darwish A. Bio-inspired computing: algorithms review, deep analysis, and the scope of applications. *Future Comput Inf J*. 2018;3(2):231-246.
32. Madić M, Marković D, Radovanović M. Comparison of meta-heuristic algorithms for solving machining optimization problems. Facta universitatis-series. *Mech Eng*. 2013;11(1):29-44.
33. İlhan U, Güvenir A. An overview of regression techniques for knowledge discovery. *Knowl Eng Rev*. 1999;14(4):319-340.

How to cite this article: Ali A, Gravino C. A systematic literature review of software effort prediction using machine learning methods. *J Softw Evol Proc*. 2019;e2211. https://doi.org/10.1002/smr.2211

APPENDIX A

All the studies selected in the performed SLR are listed below:

[S1] Mendes E. The use of a Bayesian network for web effort estimation. In International Conference on Web Engineering, Springer, Berlin, Heidelberg, 2007; 90-104.

[S2] Murillo-Morera J, Quesada-López C, Castro-Herrera C, Jenkins M. A genetic algorithm based framework for software effort prediction. Journal of Software Engineering Research and Development. 5(1):4, 2017.

[S3] Corazza A, Di Martino S, Ferrucci F, Gravino C, Mendes E. Using support vector regression for web development effort estimation. In International Workshop on Software Measurement Springer, Berlin, Heidelberg, 2009; 255-271.

[S4] Attarzadeh I, Ow SH. Proposing a new software cost estimation model based on artificial neural networks. In 2nInternational Conference on Computer Engineering and Technology (ICCET), 2010; Vol. 3, V3-487.

[S5] Sikka G, Kaur A, Uddin M. Estimating function points: using machine learning and regression models. In Education Technology and Computer (ICETC), 2010; vol. 3, V3-52.

[S6] Gharehchopogh FS. Neural networks application in software cost estimation: a case study. In Innovations in Intelligent Systems and Applications (INISTA), 2011; 69-73.

[S7] Jodpimai P, Sophatsathit P, Lursinsap C. Estimating software effort with minimum features using neural functional approximation. In Computational Science and Its Applications (ICCSA), 2010; 266-273.

[S8] Braga PL, Oliveira AL, Meira SR. Software effort estimation using machine learning techniques with robust confidence intervals. In Hybrid Intelligent Systems, 2007; 352-357.

[S9] Baskeles B, Turhan B, Bener A. Software effort estimation using machine learning methods. In Computer and information sciences, iscis 2007; 1-6.

[S10] Dejaeger K, Verbeke W, Martens D, Baesens B. Data mining techniques for software effort estimation: a comparative study. IEEE transactions on software engineering. 2012; 38(2):375-97.

[S11] Pillai SK, Jeyakumar MK. Extreme learning machine for software development effort estimation of small programs. In Circuit, Power and Computing Technologies (ICCPCT), 2014: 1698-1703.

[S12] Iwata K, Nakashima T, Anan Y, Ishii N. Effort estimation for embedded software development projects by combining machine learning with classification. In Intl Conf on Applied Computing and Information Technology/3rd Intl Conf on Computational Science/Intelligence and Applied Informatics/Intl Conf on Big Data, Cloud Computing, Data Science & Engineering (ACIT-CSII-BCD), 2016; 265-270.

[S13] Satapathy SM, Rath SK. Effort estimation of web-based applications using machine learning techniques. In International Conference Advances in Computing, Communications and Informatics (ICACCI), 2016; 973-979

[S14] Satapathy SM, Acharya BP, Rath SK. Early stage software effort estimation using random forest technique based on use case points. IET Software. 2016;10(1):10-7.

[S15] Oliveira AL, Braga PL, Lima RM, Cornélio ML. GA-based method for feature selection and parameters optimization for machine learning regression applied to software effort estimation. Information and Software Technology. 2010;52(11):1155-66.

[S16] Pospieszny P, Czarnacka-Chrobot B, Kobylinski A. An effective approach for software project effort and duration estimation with machine learning algorithms. Journal of Systems and Software, 2018; 137:184-96.

[S17] Rijwani P, Jain S. Enhanced software effort estimation using multi layered feed forward artificial neural network technique. Procedia Computer Science. 2016; 1; 89:307-12.

[S18] Dragicevic S, Celar S, Turic M. Bayesian network model for task effort estimation in agile software development. Journal of Systems and Software. 2017; 1; 127:109-19.

[S19] López-Martín C. Predictive accuracy comparison between neural networks and statistical regression for development effort of software projects. Applied Soft Computing. 2015; 27:434-49.

[S20] Sachan RK, Nigam A, Singh A, Singh S, Choudhary M, Tiwari A, Kushwaha DS. Optimizing basic COCOMO model using simplified genetic algorithm. Procedia Computer Science. 2016; 89:492-8.

[S21] Li YF, Xie M, Goh TN. A study of project selection and feature weighting for analogy based software cost estimation. Journal of Systems and Software. 2009; 82(2):241-52.

[S22] Chou JS, Cheng MY, Wu YW, Wu CC. Forecasting enterprise resource planning software effort using evolutionary support vector machine inference model. International Journal of Project Management. 2012; 30(8):967-77.

[S23] Kumar KV, Ravi V, Carr M, Kiran NR. Software development cost estimation using wavelet neural networks. Journal of Systems and Software. 2008; 81(11):1853-67.

[S24] Nassif AB, Ho D, Capretz LF. Towards an early software estimation using log-linear regression and a multilayer perceptron model. Journal of Systems and Software. 2013; 86(1):144-60.

[S25] López-Martín C, Abran A. Neural networks for predicting the duration of new software projects. Journal of Systems and Software. 2015; 101:127-35.

[S26] de Barcelos Tronto IF, da Silva JD, Sant'Anna N. An investigation of artificial neural networks based prediction systems in software project management. Journal of Systems and Software. 2008; 81(3):356-67.

[27] García-Floriano A, López-Martín C, Yáñez-Márquez C, Abran A. Support vector regression for predicting software enhancement effort. Information and Software Technology. 2018; 97:99-109.

[S28] Kultur Y, Turhan B, Bener A. Ensemble of neural networks with associative memory (ENNA) for estimating software development costs. Knowledge-Based Systems. 2009;22(6):395-402.

[S29] Zhang W, Yang Y, and Qing W. On the predictability of software efforts using machine learning techniques., 2011.

[S30] Lin, Chang, Genetic Algorithm and Support Vector Regression for Software Effort Estimation, Advanced Research on Material Engineering, 2011.

[S31] Benala TR, Dehuri S, Satapathy SC, Raghavi CS. Genetic algorithm for optimizing neural network based software cost estimation. In International Conference on Swarm, Evolutionary, and Memetic Computing, 2011; 233-239.

[S32] Aljahdali S, Sheta AF, Debnath NC. Estimating software effort and function point using regression, support vector machine and artificial neural networks models. InAICCSA 2015; 1-8.

[S33] Gabrani G, Saini N. Effort estimation models using evolutionary learning algorithms for software development. In Colossal Data Analysis and Networking (CDAN), 2016; 1-6.

[S34] Hosni M, Idri A, Nassif AB, Abran A. Heterogeneous ensembles for software development effort estimation. In Soft Computing & Machine Intelligence (ISCMI), 2016;174-178.

[S35] Azzeh M. Software effort estimation based on optimized model tree. In Proceedings of the International Conference on Predictive Models in Software Engineering 2011; 6.

[S36] Shivhare J, Rath SK. Software effort estimation using machine learning techniques. In Proceedings of the 7th India Software Engineering Conference, 2014; 19.

[S37] Han W, Jiang H, Zhang X, Li W. A neural network based algorithms for project duration prediction. In Control and Automation (CA), 2014; 60-63.

[S38] Moharreri K, Sapre AV, Ramanathan J, Ramnath R. Cost-effective supervised learning models for software effort estimation in agile environments. In Computer Software and Applications Conference (COMPSAC), 2016; 2, 135-140.

[S39] Tierno IA, Nunes DJ. An extended assessment of data-driven Bayesian networks in software effort prediction. In Software Engineering (SBES), 2013 27th Brazilian Symposium 2013; 157-166.

[S40] Minku LL, Yao X. Ensembles and locality: insight on improving software effort estimation. Information and Software Technology. 2013; 55(8): 1512-28.

[S41] Bhatia S, Attri VK. Machine learning techniques in software effort estimation using COCOMO dataset. International Journal of Research and Development organization (IJRD), 2015; 2(6).

[S42] Rao PS, Kumar RK. Software effort estimation through a generalized regression neural network. In Emerging ICT for Bridging the Future-Proceedings of the 49th Annual Convention of the Computer Society of India (CSI), 2015; 19-30.

[S43] Huang SJ, Chiu NH. Applying fuzzy neural network to estimate software development effort. Applied Intelligence.2009; 30(2):73-83.

[S44] Benala TR, Mall R, Dehuri S. Software effort estimation using functional link neural networks optimized by improved particle swarm optimization. In International Conference on Swarm, Evolutionary, and Memetic Computing, 2013; 205-213.

[S45] López-Martín C, Chavoya A, Meda-Campaña ME. Software development effort estimation in academic environments applying a general regression neural network involving size and people factors. In Mexican Conference on Pattern Recognition, 2011; 269-277.

[S46] Nassif AB, Azzeh M, Capretz LF, Ho D. Neural network models for software development effort estimation: a comparative study. Neural Computing and Applications. 2016; 27(8):2369-81.

[S47] Kaushik A, Soni AK, Soni R. An improved functional link artificial neural networks with intuitionistic fuzzy clustering for software cost estimation. International Journal of System Assurance Engineering and Management. 2016; 7(1):50-61.

[S48] Satapathy SM, Kumar M, Rath SK. Fuzzy-class point approach for software effort estimation using various adaptive regression methods. CSI transactions on ICT. 2013; 1(4):367-80.

[S49] Lefley M, Shepperd MJ. Using genetic programming to improve software effort estimation based on general data sets. In Genetic and Evolutionary Computation Conference, 2477-2487.

[S50] Benala TR, Bandarupalli R. Least square support vector machine in analogy-based software development effort estimation. In Recent Advances and Innovations in Engineering (ICRAIE), 2016; 1-6.

[S51] Mittas N, Athanasiades M, Angelis L. Improving analogy-based software cost estimation by a resampling method. Information and Software Technology. 2008;50(3):221-30.

[S52] Ferrucci F, Gravino C, Oliveto R, Sarro F. Genetic programming for effort estimation: an analysis of the impact of different fitness functions. In Proceedings of 2nd International Symposium on Search Based Software Engineering 2007; 89-98.

[S53] Ferrucci F, Gravino C, Oliveto R, Sarro F, Mendes E. Investigating tabu search for web effort estimation. In Software Engineering and Advanced Applications (SEAA), 2010; 350-357).

[S54] Briand, Lionel C., Langley T, and Wieczorek, I. A replicated assessment and comparison of common software cost modeling techniques. In Proceedings of the 22nd international conference on Software engineering, pp. 377-386. ACM, 2000.

[S55] Burgess, Colin J., and Martin Lefley. Can genetic programming improve software effort estimation? A comparative evaluation. Information and software technology43, no. 14 (2001): 863-873.

[S56] Finnie, Gavin R., Gerhard E. Wittig, and Jean-Marc Desharnais. A comparison of software effort estimation techniques: using function points with neural networks, case-based reasoning and regression models. Journal of systems and software 39, no. 3 (1997): 281-289.

[S57] Mendes, Emilia, and Nile Mosley. Further investigation into the use of CBR and stepwise regression to predict development effort for web hypermedia applications. In Proceedings International Symposium on Empirical Software Engineering, pp. 79-90. IEEE, 2002.

[S58] Oliveira, Adriano LI. Estimation of software project effort with support vector regression. Neurocomputing 69, no. 13-15 (2006): 1749-1753.

[S59] Shin, Miyoung, and Amrit L. Goel. Empirical data modeling in software engineering using radial basis functions. IEEE Transactions on Software Engineering 26, no. 6 (2000): 567-576.

[S60] Stensrud, Erik, and Ingunn Myrtveit. Human performance estimating with analogy and regression models: an empirical validation. In Proceedings Fifth International Software Metrics Symposium. Metrics (Cat. No. 98 TB100262), pp. 205-213. IEEE, 1998.

[S61] Wittig, Gerhard, and Gavin Finnie. Estimating software development effort with connectionist models. Information and Software Technology 39, no. 7 (1991): 469-476.

[S62] Mendes, Emilia, Ian Watson, Chris Triggs, Nile Mosley, and Steve Counsell. A comparative study of cost estimation models for web hypermedia applications. Empirical Software Engineering 8, no. 2 (2003): 163-196.

[S63] Gray, Andrew R., and Stephen G. Macdonell. Software metrics data analysis—exploring the relative performance of some commonly used modeling techniques. Empirical Software Engineering 4, no. 4 (1999): 297-316.

[S64] Jeffery, Ross, Melanie Ruhe, and Isabella Wieczorek. A comparative study of two software development cost modeling techniques using multi-organizational and company-specific data. Information and software technology 42, no. 14 (2000): 1009-1016.

[S65] Heiat, Abbas. Comparison of artificial neural network and regression models for estimating software development effort. Information and software Technology 44, no. 15 (2002): 911-922.

[S66] Shepperd, Martin, and Chris Schofield. Estimating software project effort using analogies. IEEE Transactions on software engineering 23, no. 11 (199): 736-743.

[S67] Gray, Andrew R., and Stephen G. MacDonell. A comparison of techniques for developing predictive models of software metrics. Information and software technology 39, no. 6 (1997): 425-437.

[S68] Briand, Lionel C., Khaled El Emam, Dagmar Surmann, Isabella Wieczorek, and Katrina D. Maxwell. An assessment and comparison of common software cost estimation modeling techniques. In Proceedings of the 1999 International Conference on Software Engineering (IEEE Cat. No. 99CB37002), pp. 313-323. IEEE, 1999.

[S69] Chiu, Nan-Hsing, and Sun-Jen Huang. The adjusted analogy-based software effort estimation based on similarity distances. Journal of Systems and Software 80, no. 4 (2007): 628-640.

[S70] Jeffery, Ross, Melanie Ruhe, and Isabella Wieczorek. Using public domain metrics to estimate software development effort. In Proceedings Seventh International Software Metrics Symposium, pp. 16-27. IEEE, 2001.

[S71] Lee, Anita, Chun Hung Cheng, and Jaydeep Balakrishnan. Software development cost estimation: integrating neural network with cluster analysis. Information & Management 34, no. 1 (1998): 1-9.

[S72] Wittig, Gerhard E., and G. R. Finnic. Using artificial neural networks and function points to estimate 4GL software development effort. Australasian Journal of Information Systems 1, no. 2 (1994).

[S73] Srinivasan, Krishnamoorthy, and Douglas Fisher. Machine learning approaches to estimating software development effort. IEEE Transactions on Software Engineering 21, no. 2 (1995): 126-137.

[S74] Chulani, Sunita, Barry Boehm, and Bert Steece. Bayesian analysis of empirical software engineering cost models. IEEE Transactions on Software Engineering 25, no. 4 (1999): 573-583.

[S75] Corazza, Anna, Sergio Di Martino, Filomena Ferrucci, Carmine Gravino, Federica Sarro, and Emilia Mendes. How effective is Tabu search to configure support vector regression for effort estimation? In Proceedings of the International Conference on Predictive Models in Software Engineering 2010: 4

APPENDIX B

To evaluate the quality of (software effort) estimations obtained with different techniques, different accuracy metrics have been introduced and used. In the following, we reported the definitions of accuracy measures employed in the studies selected by our SLR (see Section B1) and then the results achieved in those studies in terms of these measures (see Section B2).

B1. Definition of evaluation criteria

Mean absolute error

One of the simplest accuracy measures used is the mean of absolute errors (MAE), which describes the difference between the actual and predicted values (how far is the predicted value from the actual value) and then find the average of it. The values of MAE can be obtained using the following formula:

$$\mathrm{MAE} = (\mathrm{AE\_1} + \mathrm{AE\_2} + \ldots + \mathrm{AE\_}n)/n$$

where AE_*i* = |Actual_*i* − Predicted_*i*|) for *i* = 1,2, … *n*.

Mean (and median) of magnitude of relative error

One of the widely used accuracy measures is the mean of magnitude of relative error (MMRE), where the magnitude of relative error (MRE) is defined as

$$\mathrm{MRE\_}i = \left|\mathrm{Actual\_}i - \mathrm{Predicted\_}i\right|/\mathrm{Actual\_}i$$

for *i* = 1,2, … *n* and MMRE is equal to Mean (MRE_1, MRE_2, …, MRE_*n*).

The median of magnitude of relative error (MdMRE) is equal to Median (MRE_1, MRE_2, …, MRE_*n*).

Prediction at level l

The prediction at level l, Pred(l), is widely used in combination with MMRE and measures the percentage of estimates that are within l% of the actual values, and l is usually set at 25. In other words, it can be defined:

$$\mathrm{Pred(l)} = k/n$$

where *k* is the number of observations whose MRE is less than or equal to l, and *n* is the total number of observations.

Mean of magnitude of error relative to the estimate

This measure is based on the notion of magnitude of error relative to the estimate (MER) defined as

$$\mathrm{EMRE\_}i = \left|\mathrm{Actual\_}i - \mathrm{Predicted\_}i\right|/\mathrm{Predicted\_}i$$

for *i* = 1,2, … *n* and mean of EMRE is equals to Mean (EMRE_1, EMRE_2, …, EMRE_*n*).

Mean square error

The mean square error (MSE) is calculated exploiting the square of the difference between the estimated value and the actual value observed for each observation, as shown in the following equation:

$$\mathrm{SE\_}i = (\mathrm{Actual\_}i - \mathrm{Predicted\_}i)\hat{}2$$

for *i* = 1,2, … *n* and MSE is equal to Mean (SE_1, SE_2, …, SE_*n*).

Root mean square error.

The value of root mean squared error (RMSE) is achieved as the square root of the MSE.

B2. Results in terms of evaluation criteria

In Table 9, we present the accuracy metric values obtained in the studies selected by our SLR for the widely used ML techniques. We selected only the values of MMRE, MdMRE, and Pred(25) because these are widely used accuracy measures not only in our SLR but in a wide variety of articles published in the area of software effort estimation. It is worth noting that for a single machine learning technique, we can have different entries labeled with the same study ID since it has been applied on different datasets in the same study. Also, there are some cases in which we have multiple applications of a machine learning technique. For instance, there are some studies in which multiple types of ANN are used like multilayer perceptron (MLP), recursive neural network, fuzz neural network, etc.