Information and Software Technology 54 (2012) 41–59

Contents lists available at SciVerse ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Systematic literature review of machine learning based software development effort estimation models

Jianfeng Wen [a,*], Shixian Li [a], Zhiyong Lin [b], Yong Hu [c], Changqin Huang [d]

[a] Department of Computer Science, Sun Yat-sen University, Guangzhou, China
[b] Department of Computer Science, Guangdong Polytechnic Normal University, Guangzhou, China
[c] Institute of Business Intelligence and Knowledge Discovery, Department of E-commerce, Guangdong University of Foreign Studies, Sun Yat-sen University, Guangzhou, China
[d] Engineering Research Center of Computer Network and Information Systems, South China Normal University, Guangzhou, China

ARTICLE INFO

Article history:
Received 28 October 2010
Received in revised form 8 August 2011
Accepted 8 September 2011
Available online 16 September 2011

Keywords:
Software effort estimation
Machine learning
Systematic literature review

ABSTRACT

Context: Software development effort estimation (SDEE) is the process of predicting the effort required to develop a software system. In order to improve estimation accuracy, many researchers have proposed machine learning (ML) based SDEE models (ML models) since 1990s. However, there has been no attempt to analyze the empirical evidence on ML models in a systematic way.
Objective: This research aims to systematically analyze ML models from four aspects: type of ML technique, estimation accuracy, model comparison, and estimation context.
Method: We performed a systematic literature review of empirical studies on ML model published in the last two decades (1991–2010).
Results: We have identified 84 primary studies relevant to the objective of this research. After investigating these studies, we found that eight types of ML techniques have been employed in SDEE models. Overall speaking, the estimation accuracy of these ML models is close to the acceptable level and is better than that of non-ML models. Furthermore, different ML models have different strengths and weaknesses and thus favor different estimation contexts.
Conclusion: ML models are promising in the field of SDEE. However, the application of ML models in industry is still limited, so that more effort and incentives are needed to facilitate the application of ML models. To this end, based on the findings of this review, we provide recommendations for researchers as well as guidelines for practitioners.

Contents

* Corresponding author. Tel.: +86 20 34022695.
E-mail address: wjfsysu@gmail.com (J. Wen).

0950-5849/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.infsof.2011.09.002

1. Introduction

Software development effort estimation (SDEE) is the process of predicting the effort required to *develop* a software system. Generally speaking, SDEE can be considered as a sub-domain of *software effort estimation*, which includes the predictions of not only software development effort but also software maintenance effort. The terminology *software cost estimation* is often used interchangeably with *software effort estimation* in research community, though *effort* just forms the major part of software *cost*. Estimating development effort accurately in the early stage of software life cycle plays a crucial role in effective project management. Since 1980s, many estimation methods have been proposed in SDEE domain. Jørgensen and Shepperd [1] conducted a review which identifies up to 11 estimation methods used for SDEE. Among them, regression-based method is found to dominate, and the use of expert judgment and analogy method is increasing. In recent years, machine learning (ML) based method has been receiving increasing attention in SDEE research. Some researchers [2–4] even view ML based method as one of the three major categories of estimation methods (the other two are expert judgment and algorithmic model). Boehm and Sullivan [5] considered the learning-oriented technique as one of the six categories of software effort estimation techniques. Zhang and Tsai [6] summarized the applications of various ML techniques in SDEE domain, including case-based reasoning, decision trees, artificial neural networks, and genetic algorithms.

Despite the large number of empirical studies on ML models,[1] inconsistent results have been reported regarding the estimation accuracy of ML models, the comparisons between ML model and non-ML model, and the comparisons between different ML models. For example, it was reported that the estimation accuracy varies under the same ML model when it is constructed with different historical project data sets [7–10] or different experimental designs [11]. As for the comparison between ML model and regression model, studies in [12–14] reported that ML model is superior to regression model, while studies in [2,15,16] concluded that regression model outperforms ML model. As for the comparison between different ML models (e.g., artificial neural networks and case-based reasoning), study in [17] suggested that the former outperforms the latter while study in [18] reported the opposite result.

The divergence in the existing empirical studies on ML models, the causes of which are still not fully understood up to now, may prevent practitioners from adopting ML models in practice. Comparing with other domains in which ML techniques have been applied successfully, SDEE domain poses many challenges such as small training data sets, information uncertainty, qualitative metrics, and human factors. Furthermore, the theory of ML techniques is much complicated than that of conventional estimation techniques. Although the research on ML model is increasing in academia, recent surveys [1,19,20] have revealed that expert judgment (a kind of non-ML method) is still the dominant method for SDEE in industry. To facilitate the applications of ML techniques in SDEE domain, it is crucial to systematically summarize the empirical evidence on ML models in current research and practice.

Existing literature reviews of SDEE can be divided into two categories: traditional literature reviews and systematic literature reviews (SLR). The traditional reviews [5,21–26] mainly cover the state-of-the-art and research trends, whereas the SLRs [1,19,20,27–31] aim to answer various research questions pertaining to SDEE. To the best of authors' knowledge, there is *no* existing SLR that focuses on ML models, which motivates our work in this paper. Specifically, we performed an SLR on ML models published in the period from *1 January 1991* to *31 December 2010*. The purpose of this SLR is to summarize and clarify the available evidence regarding (1) the ML techniques for constructing SDEE models, (2) the estimation accuracy of ML models, (3) the comparisons of estimation accuracy between ML models and non-ML models, (4) the comparisons of estimation accuracy between different ML models, and (5) the favorable estimation contexts of ML models. We further provide practitioners with guidelines on the application of ML models in SDEE practice.

The rest of this paper is organized as follows. Section 2 describes the methodology used in this review. Section 3 presents and discusses the review results. Section 4 provides the implications for research and practice. Section 5 discusses the limitations of this review. Conclusion and future work are presented in Section 6.

2. Method

We planned, conducted, and reported the review by following the SLR process suggested by Kitchenham and Charters [32]. We developed the review protocol at the planning phase of this SLR. The review protocol mainly includes six stages: research questions definition, search strategy design, study selection, quality assessment, data extraction, and data synthesis. Fig. 1 outlines the six stages of the review protocol.

In the first stage, we raised a set of research questions based on the objective of this SLR. Then, in the second stage, aiming at the research questions, we designed search strategy to find out the studies relevant to the research questions. This stage involves both the determination of search terms and the selection of literature resources, which are necessary for the subsequent search process. In the third stage, we defined study selection criteria to identify the relevant studies that can really contribute to addressing the research questions. As part of this stage, pilot study selection was employed to further refine the selection criteria. Next, the relevant studies underwent a quality assessment process in which we devised a number of quality checklists to facilitate the assessment. The remaining two stages involve data extraction and data synthesis, respectively. In the data extraction stage, we initially devised data extraction form and subsequently refined it through pilot data extraction. Finally, in the data synthesis stage, we determined the

[1] We use the term "ML model" in this review to denote "ML based SDEE method". This term may cause confusion since one ML technique identified in this review, i.e., case-based reasoning (CBR), belongs to lazy learning technique and thus has no model. Despite that, we still use this term for descriptive convenience.

proper methodologies for synthesizing the extracted data based on the types of the data and the research questions the data addressed.

A review protocol is of critical importance for an SLR. To ensure the rigorousness and repeatability of this SLR and to reduce researcher bias as well, we elaborately developed the review protocol by frequently holding group discussion meetings on protocol design. In the following Subsection 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, we will present the details of the review protocol. At the end of this section, we will analyze the threats to validity of the review protocol.

2.1. Research questions

This SLR aims to summarize and clarify the empirical evidence on ML based SDEE models. Towards this aim, five research questions (RQs) were raised as follows.

(1) *RQ1: Which ML techniques have been used for SDEE?*
RQ1 aims at identifying the ML techniques that have been used to estimate software development effort. Practitioners can take the identified ML techniques as candidate solutions in their practice. For ML techniques that have not yet been employed in SDEE, researchers can explore the possibility of using them as potential feasible solutions.
(2) *RQ2: What is the overall estimation accuracy of ML models?*
RQ2 is concerned with the estimation accuracy of ML models. Estimation accuracy is the primary performance metric for ML models. This question focuses on the following four aspects of estimation accuracy: accuracy metric, accuracy value, data set for model construction, and model validation method.
(3) *RQ3: Do ML models outperform non-ML models?*
In most of the existing studies, the proposed ML models are compared with conventional non-ML models in terms of estimation accuracy. RQ3 therefore aims to verify whether ML models are superior to non-ML models.
(4) *RQ4: Are there any ML models that distinctly outperform other ML models?*
The evidence of comparisons between different ML models can be synthesized to determine which ML models consistently outperform other ML models. Thus, RQ4 aims to identify the ML models with relatively excellent performance.
(5) *RQ5: What are the favorable estimation contexts of ML models?*
RQ5 aims at identifying the strengths and weaknesses of different ML models. With fully understanding the characteristics of the candidate ML models, practitioners can make a rational decision on choosing the ML models that favor the focused estimation contexts.

2.2. Search strategy

The search strategy comprises search terms, literature resources, and search process, which are detailed one by one as follows.

2.2.1. Search terms

The following steps were used to construct the search terms [30]:

(a) Derive major terms from the research questions.
(b) Identify alternative spellings and synonyms for major terms.
(c) Check the keywords in relevant papers or books.
(d) Use the Boolean OR to incorporate alternative spellings and synonyms.
(e) Use the Boolean AND to link the major terms.

The resulting complete search terms are as follows. Note that the terms of ML techniques mainly come from the textbooks on machine learning [33–35].

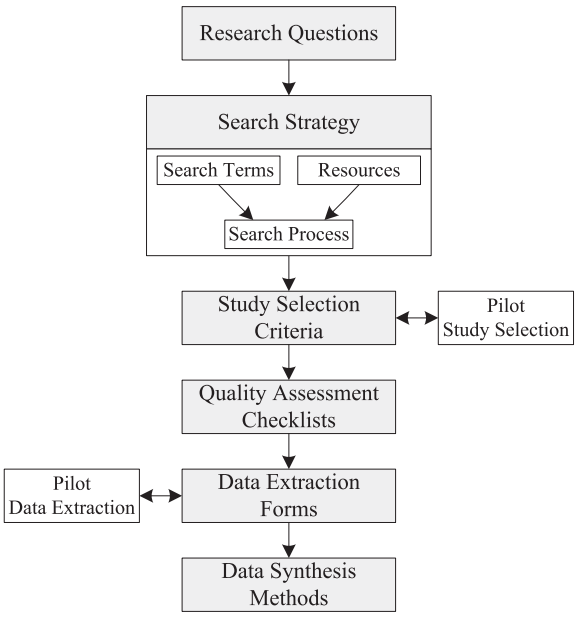

Fig. 1. Stages of review protocol.

software AND (effort OR cost OR costs) AND (estimat* OR predict*) AND (learning OR "data mining" OR "artificial intelligence" OR "pattern recognition" OR analogy OR "case based reasoning" OR "nearest neighbo*" OR "decision tree*" OR "regression tree*" OR "classification tree*" OR "neural net*" OR "genetic programming" OR "genetic algorithm*" OR "bayesian belief network*" OR "bayesian net*" OR "association rule*" OR "support vector machine*" OR "support vector regression").

2.2.2. Literature resources

The literature resources we used to search for primary studies include six electronic databases (*IEEE Xplore, ACM Digital Library, ScienceDirect, Web of Science, EI Compendex,* and *Google Scholar*) and one online bibliographic library (*BESTweb*). Some other important resources such as *DBLP, CiteSeer,* and *The Collection of Computer Science Bibliographies* have not been considered, since they are almost covered by the selected literature resources.

The online bibliographic library BESTweb (http://www.simula.no/BESTweb), which is maintained by Simula Research Laboratory, supplies both journal papers and conference papers on software effort estimation. At the time of conducting this review, the library contained up to 1365 papers that were published in the period 1927–2009. More details about BESTweb can be found in [1].

The search terms constructed previously were used to search for journal papers and conference papers in the six electronic databases. The search terms were adjusted to accommodate different databases, since the search engines of different databases use different syntax of search strings. The search was conducted on the first five databases covering title, abstract, and keywords. For Google Scholar, only title was searched since the full text search will return millions of irrelevant records.

We restricted the search to the period from 1 January 1991 to 31 December 2010, because the application of ML techniques in SDEE domain was launched just in the early 1990s. For example, Mukhopadhyay et al. [36] used case-based reasoning technique, Briand et al. [37] used decision trees technique, both in 1992.

2.2.3. Search process

SLR requires a comprehensive search of all relevant sources. For this reason, we defined the search process and divided it into the

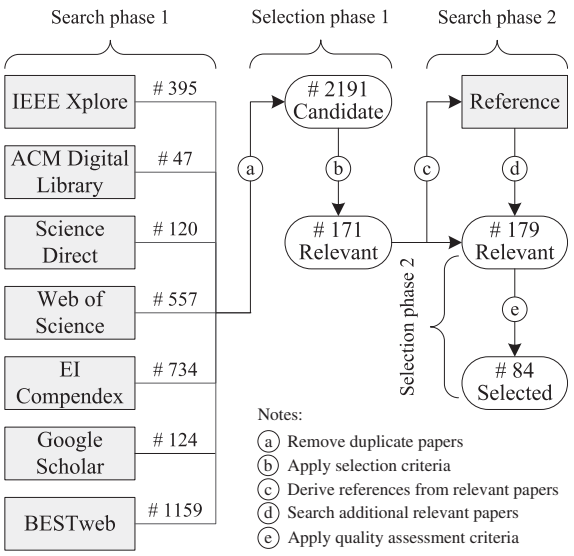

Fig. 2. Search and selection process.

following two phases (note that the relevant papers are those papers that meet the selection criteria defined in the next section).

- Search phase 1: Search the six electronic databases separately and then gather the returned papers together with those from BESTweb to form a set of candidate papers.
- Search phase 2: Scan the reference lists of the relevant papers to find extra relevant papers and then, if any, add them into the set.

Software package Endnote (http://www.endnote.com) was used to store and manage the search results. We identified 179 relevant papers according to the search process. The detailed search process and the number of papers identified at each phase are shown in Fig. 2.

2.3. Study selection

Search phase 1 resulted in 2191 candidate papers (see Fig. 2). Since many of the candidate papers provide no information useful to address the research questions raised by this review, further filtering is needed to identify the relevant papers. This is exactly what study selection aims to do. Specifically, as illustrated in Fig. 2, the study selection process consists of the following two phases. (Note that in each selection phase, two researchers of this review conducted the selection independently. If there were any disagreements between them, a group meeting involving all researchers would be held to resolve the disagreements.)

- Selection phase 1: Apply the inclusion and exclusion criteria (defined below) to the candidate papers so as to identify the relevant papers, which provide potential data for answering the research questions.
- Selection phase 2: Apply the quality assessment criteria (defined in the next section) to the relevant papers so as to select the papers with acceptable quality, which are eventually used for data extraction.

We defined the following inclusion and exclusion criteria, which had been refined through pilot selection. We carried out the study selection by reading the titles, abstracts, or full text of the papers.

Inclusion criteria:

- Using ML technique to estimate development effort.
- Using ML technique to preprocess modeling data.
- Using hybrid model that employs at least two ML techniques or combines ML technique with non-ML technique (e.g., combining with statistics method, fuzzy set, or rough set) to estimate development effort.
- Comparative study that compares different ML models or compares ML model with non-ML model.
- For study that has both conference version and journal version, only the journal version will be included.
- For duplicate publications of the same study, only the most complete and newest one will be included.

Exclusion criteria:

- Estimating software size, schedule, or time only, but without estimating effort.
- Estimating maintenance effort or testing effort.
- Addressing software project control and planning issues (e.g., scheduling, staff allocation, development process).
- Review papers will be excluded.

By applying the selection criteria in selection phase 1, we identified 171 relevant papers. Then, by scanning the references in these relevant papers, we identified 8 extra relevant papers that were missed in the initial search. Therefore, we identified in total 179 relevant papers.[2] After applying the quality assessment criteria in selection phase 2, we ultimately identified 84 papers as the final selected studies, which were then used for data extraction. The details of the quality assessment are described in the next section. The complete list of these 84 selected papers can be found in Table A.8 in Appendix A.

2.4. Study quality assessment

The quality assessment (QA) of selected studies is originally used as the basis for weighting the retrieved quantitative data in meta-analysis [38], which is an important data synthesis strategy. However, since the retrieved data in this review were produced by different experimental designs, and the amount of data is relatively small, meta-analysis is thus unsuitable for such cases. For this reason, we did not use the quality assessment results to weight the retrieved data. Instead, we just used the quality assessment results to guide the interpretation of review findings and to indicate the strength of inferences. Moreover, the quality assessment results will be served as an additional criterion for study selection (see selection phase 2 in Fig. 2).

We devised a number of quality assessment questions to assess the rigorousness, credibility, and relevance of the relevant studies. These questions are presented in Table 1. Some of them (i.e., QA1, QA7, QA8, and QA10) are derived from [39]. Each question has only three optional answers: "Yes", "Partly", or "No". These three answers are scored as follows: "Yes" = 1, "Partly" = 0.5, and "No" = 0. For a given study, its quality score is computed by summing up the scores of the answers to the QA questions.

Two researchers of this review performed the quality assessment of the relevant studies individually. All disagreements on the quality assessment results were discussed among all researchers, and the consensus was reached eventually. To ensure the reliability of the findings of this review, we considered only the relevant studies with acceptable quality, i.e., with quality score

[2] Please contact the authors to obtain the full list of the relevant papers.

Table 1
Quality assessment questions.

No.	Question
QA1	Are the aims of the research clearly defined?
QA2	Is the estimation context adequately described?
QA3	Are the estimation methods well defined and deliberate?
QA4	Is the experimental design appropriate and justifiable?
QA5[a]	Is the experiment applied on sufficient project data sets?
QA6	Is the estimation accuracy measured and reported?
QA7	Is the proposed estimation method compared with other methods?
QA8	Are the findings of study clearly stated and supported by reporting results?
QA9	Are the limitations of study analyzed explicitly?
QA10	Does the study add value to academia or industry community?

[a] "Yes" (Sufficient): two or more data sets; "Partly" (Partly sufficient): only one data set; "No" (Insufficient): no data set.

Table 2
The form of data extraction card.

Data extractor
Data checker
Study identifier
Year of publication
Name of authors
Source
Article title
Type of study (experiment, case study, or survey)
RQ1: Which ML techniques have been used for SDEE?
ML techniques used to estimate software development effort
RQ2: What is the overall estimation accuracy of ML models?
Data sets used in experiments
Validation methods used in experiments
Metrics used to measure estimation accuracy
Estimation accuracy values
RQ3. Do ML models outperform non-ML models?
Non-ML models that this ML model compares with
Rank of the ML and non-ML models regarding estimation accuracy
Degree of improvement in estimation accuracy
RQ4. Are there any ML models that distinctly outperform other ML models?
Other ML models that this ML model compares with
Rank of the ML models regarding estimation accuracy
Degree of improvement in estimation accuracy
RQ5. What are the favorable estimation contexts of ML models?
Strengths of ML technique in terms of SDEE
Weaknesses of ML technique in terms of SDEE

greater than 5 (50% of perfect score), for the subsequent data extraction and data synthesis. Accordingly, we further dropped 95 relevant papers with quality score not more than 5 in selection phase 2 (see Fig. 2). The quality scores of the remaining studies are presented in Table B.9 in Appendix B.

2.5. Data extraction

By data extraction, we exploited the selected studies to collect the data that contribute to addressing the research questions concerned in this review. We devised the cards with the form of Table 2 to facilitate data extraction. This form had been refined through pilot data extraction with several selected studies. For ease of the subsequent data synthesis, the items in Table 2 are grouped according to the associated research questions.

Table 2 shows that the data extracted for addressing RQ2, RQ3, and RQ4 are related to the experiments conducted by the selected studies. We notice that the term "experiment" is used with varying meaning in the software engineering community, and most of the selected studies used this term without clear definition. Hence, to avoid confusion, in this review we explicitly define the term *experiment* as a process in which a technique or an algorithm is evaluated with appropriate performance metrics based on a particular data set. It should be pointed out that the term *experiment* defined above is different from the term *controlled experiment in software engineering* defined by Sjøeberg et al. [40].

We used the data extraction cards to collect data from the selected studies. One researcher of this review extracted the data and filled them into the cards. Another researcher double-checked the extraction results. The checker discussed disagreements (if any) with the extractor. If they failed to reach a consensus, other researchers would be involved to discuss and resolve the disagreements. The verified extracted data were documented into a file, which would be used in the subsequent data synthesis.

During the data extraction, some data could not be extracted directly from the selected studies. Nevertheless, we could obtain them indirectly by processing the available data appropriately. For example, the estimation accuracy values of the same model may vary due to different model configurations or different samplings of data set. For the former case, we chose the accuracy value generated by the optimal configuration; for the latter case, we calculated the mean of the accuracy values. Moreover, to ensure the reliability of the extracted estimation accuracy of ML models, we did not extract the accuracy data from the experiments that were conducted on student projects data sets or simulated project data sets. That is, we extracted only the accuracy data from the experiments conducted on real-world project data sets.

Another issue we encountered during data extraction was that some studies used different terminologies for the same ML technique. For example, some studies used *analogy* as the synonym of *case-based reasoning*. Similarly, some studies used *regression trees*, but other ones used *classification and regression trees*; both techniques belong to the category of *decision trees*. Also, *optimized set reduction* [37] is essentially a form of decision trees. To avoid ambiguity, we adopted the terms *case-based reasoning* and *decision trees* uniformly in data extraction and throughout this review.

Not every selected study provides answers to all the five research questions. For ease of tracing the extracted data, we explicitly labeled each study with the IDs of the research questions to which the study can provide the corresponding answers. See Table A.8 in Appendix A for the details.

2.6. Data synthesis

The goal of data synthesis is to aggregate evidence from the selected studies for answering the research questions. A single piece of evidence might have small evidence force, but the aggregation of many of them can make a point stronger [41]. The data extracted in this review include both quantitative data (e.g., values of estimation accuracy) and qualitative data (e.g., strengths and weaknesses of ML techniques). For the quantitative data, the ideal synthesis methodology is the canonical meta-analysis [38] due to its solid theoretical background. However, the quantitative data in this review were achieved by using varying experimental designs. Moreover, the number of selected studies was relatively small. Therefore, it is unsuitable to employ the standard meta-analysis directly in such a situation [42–44].

We employed different strategies to synthesize the extracted data pertaining to different kinds of research questions. The synthesis strategies are explained in detail as follows.

For the data pertaining to RQ1 and RQ2, we used *narrative synthesis* method. That is, the data were tabulated in a manner consistent with the questions. Some visualization tools, including bar chart, pie chart, and box plot, were also used to enhance the

presentation of the distribution of ML techniques and their estimation accuracy data.

For the data pertaining to RQ3 and RQ4, which focus on the comparison of estimation accuracy between different estimation models, we used *vote counting* method. That is, suppose we want to synthesize the comparisons between model A and model B. We just need to count the number of experiments in which model A is reported to outperform model B and the number of experiments in which model B is reported to outperform model A. The two numbers are then used as the basis to analyze the overall comparison between model A and model B. By this way, we can obtain a general idea about whether an estimation technique outperforms another technique or not.

For the data pertaining to RQ5, we used *reciprocal translation* method [45], which is one of the meta-ethnography techniques for synthesizing qualitative data. As Kitchenham and Charters suggested [32], "when studies are about similar things and researchers are attempting to provide an additive summary, synthesis can be achieved by 'translating' each case into each of the other cases". In this review, reciprocal translation method is applicable when some strengths or weaknesses of an ML technique retrieved from different studies have similar meanings. We will translate these strengths or weaknesses into a uniform description of strength or weakness. For instance, suppose we have identified three different but similar descriptions on the strength of an ML technique from three studies: (1) "can deal with problems where there are complex relationships between inputs and outputs", (2) "can automatically approximate any continuous function", and (3) "has the ability of modeling complex non-linear relationships and are capable of approximating any measurable function". By applying reciprocal translation, we may unify the above three descriptions into the same one "can learn complex (non-linear) functions".

2.7. Threats to validity

The main threats to validity of this review protocol are analyzed from the following three aspects: study selection bias, publication bias, and possible inaccuracy in data extraction.

The selection of studies depends on the search strategy, the literature sources, the selection criteria, and the quality criteria. We have constructed the search terms matching with the research questions and have used them to retrieve the relevant studies in the six electronic databases. However, some desired relevant studies may not use the terms related to the research questions in their titles, abstracts, or keywords. As a result, we may have the high risk of leaving out these studies in the automatic search. To address this threat, we manually searched BESTweb [1], a bibliographic library with abundant papers on software effort estimation, as a supplementary way to find out those relevant studies that were missed in the automatic search. Besides this solution, we also manually scanned the references list of each relevant study to look for the extra relevant studies that were not covered by the search of the six electronic databases and BESTweb library. In addition, we have defined the selection criteria that strictly comply with the research questions to prevent the desired studies from being excluded incorrectly. The final decision on study selection was made through double confirmation, i.e., separate selections by two researchers at first and then disagreements resolution among all researchers. Nevertheless, it is still possible that some relevant studies have been missed. If such studies do exist, we believe that the number of them would be small.

Publication bias is possibly another threat to validity of this review. Given the likelihood of publication bias, positive results on ML models are more likely to be published than negative results, or researchers may tend to claim that their methods outperform others'. As a consequence, this will lead to an overestimation of the performance of ML models. Fortunately, one inclusion criterion (i.e., the fourth inclusion criterion) in this review may mitigate this threat. This inclusion criterion aims to identify the studies that do not propose any new ML model but just conduct comparisons between ML models and other models. These comparative studies have no bias towards any estimation model; they report the comparison results in an objective way. Therefore, in these studies both positive and negative comparison results are likely to be reported, which helps alleviate publication bias to some extent. Nevertheless, since this review did not consider gray literature (i.e., theses, technical reports, white papers, and work in progress) due to the difficulties in obtaining them, it must be admitted that publication bias exists unavoidably.

To reduce the threat of inaccurate data extraction, we have elaborated the specialized cards for data extraction. In addition, all disagreements between extractor and checker have been resolved by discussion among all researchers. However, since the data extraction strategy was designed to extract only the optimal one from the multiple estimation accuracy values produced by different model configurations, an accuracy bias could still occur. Even so, we believe that the accuracy bias is limited, as the optimal configuration of a model is usually preferable in practice.

3. Results and discussion

This section presents and discusses the findings of this review. First, we present an overview of the selected studies. Second, we report and discuss the review findings according to the research questions, one by one in the separate subsections. During the discussion, we interpret the review results not only within the context of the research questions, but also in a broader context closely related to the research questions. For justification, some related works are also provided to support the findings.

3.1. Overview of selected studies

We identified 84 studies (see Table A.8 in Appendix A) in the field of ML based SDEE. These papers were published during the time period 1991–2010. Among them, 59 (70%) papers were published in journals, 24 (29%) papers appeared in conference proceedings, and one paper came from book chapter. The publication venues and distribution of the selected studies are shown in Table 3. The publication venues mainly include *Information and Software Technology* (IST), *IEEE Transactions on Software Engineering* (TSE), *Journal of Systems and Software* (JSS), and *Empirical Software Engineering* (EMSE). These four journals are prestigious in software engineering domain [46], and 44 (52%) studies for this review are retrieved from them.

Regarding the types of the selected studies, all the studies belong to experiment research except for one case study research (S9), and no survey research was found. Although most of the selected studies used at least one project data set from industry to validate ML models, it does not follow that the validation results sufficiently reflect the real situations in industry. In fact, the lack of case study and survey from industry may imply that the application of ML techniques in SDEE practice is still immature.

Regarding the quality of the selected studies, since in the study selection strategy we have required the quality score of study must exceed 5 (notice that the perfect score of quality assessment is 10), we believe that the selected studies are with high quality. In fact, as shown in Table 4, about 70% (58 of 84) of the selected studies are in high or very high quality level.

Table 3
Publication venues and distribution of selected studies.

Publication venue	Type	# Of studies	Percent
Information and Software Technology (IST)	Journal	14	16
IEEE Transactions on Software Engineering (TSE)	Journal	12	14
Journal of Systems and Software (JSS)	Journal	9	11
Empirical Software Engineering (EMSE)	Journal	9	11
Expert Systems with Applications	Journal	4	5
International Conference on Predictive Models in Software Engineering (PROMISE)	Conference	4	5
International Software Metrics Symposium (METRICS)	Conference	4	5
International Symposium on Empirical Software Engineering and Measurement (ESEM)	Conference	3	4
International Conference on Tools with Artificial Intelligence (ICTAI)	Conference	3	4
International Conference on Software Engineering (ICSE)	Conference	2	2
Journal of Software Maintenance and Evolution: Research and Practice	Journal	2	2
Others		18	21
Total		84	100

Table 4
Quality levels of relevant studies.

Quality level	# Of studies	Percent
Very high ($8.5 \leqslant score \leqslant 10$)	12	6.7
High ($7 \leqslant score \leqslant 8$)	46	25.7
Medium ($5.5 \leqslant score \leqslant 6.5$)	26	14.5
Low ($3 \leqslant score \leqslant 5$)	70	39.1
Very low ($0 \leqslant score \leqslant 2.5$)	25	14.0
Total	179	100

3.2. Types of ML techniques (RQ1)

From the selected studies, we identified eight types of ML techniques that had been applied to estimate software development effort. They are listed as follows.

- Case-Based Reasoning (CBR)
- Artificial Neural Networks (ANN)
- Decision Trees (DT)
- Bayesian Networks (BN)
- Support Vector Regression (SVR)
- Genetic Algorithms (GA)
- Genetic Programming (GP)
- Association Rules (AR)

Among the above listed ML techniques, CBR, ANN, and DT are the three most frequently used ones; they together were adopted by 80% of the selected studies, as illustrated in Fig. 3. The detailed information about which ML techniques were used in each selected study can be found in Table C.10 in Appendix C. Fig. 3 presents only the amount of research attention that each type of ML technique has received during the past two decades; as a complement to Fig. 3, Fig. 4 is plotted to further present the distribution of research attention in each publication year. As shown in Fig. 4, on one hand, an obvious publication peak appears around year 2008;

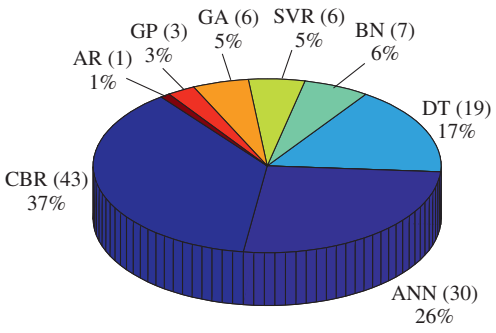

Fig. 3. Distribution of the studies over type of ML technique.

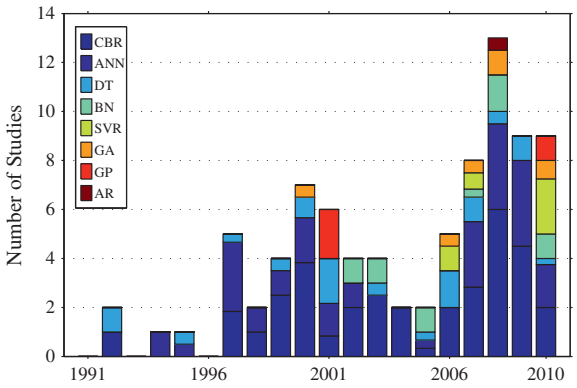

Fig. 4. Distribution of the studies over publication year.

on the other hand, compared to other ML techniques, CBR and ANN seem to have received dominant research attention in many years. Note that some studies contain more than one ML technique.

The identified ML techniques were used to estimate software development effort usually in two forms: in alone or in combination. The combination form may be obtained by combining two or more ML techniques or by combining ML techniques with non-ML techniques. The typical ML technique and non-ML technique that were often used to combine with other ML techniques are *GA* and *fuzzy logic*, respectively. As for GA, according to the selected studies, it was found to be used only in combination form. The studies reporting the use of GA in combination with other ML techniques are, for example, S28 (with CBR), S74 (with ANN), and S65 (with SVR). In these combined models, GA was used just for feature weighting (S28, S74) or feature selection (S65). As for fuzzy logic, it was widely used to deal with uncertainty and imprecision information. In particular, it was advocated to combine fuzzy logic with some ML techniques such as CBR (S5), ANN (S29), and DT (S31) to improve model performance by preprocessing the inputs of the models.

What we found about the ML techniques used in SDEE domain is highly consistent with the findings of several other relevant review works. For instance, Zhang and Tsai [6] identified five ML techniques (CBR, ANN, DT, BN, GA) that are used for software effort estimation, all of which have also been identified in this review. Jørgensen and Shepperd [1] summarized the distribution of studies according to estimation methods. Specifically, they identified up to 11 estimation methods, of which four are ML methods, i.e., CBR (or Analogy), ANN, DT, and BN (ranked in descending order of the number of studies). These four ML methods have also been identified in this review, even their ranking order in terms of the number of studies is the same.

3.3. Estimation accuracy of ML models (RQ2)

Since ML model is data-driven, both model construction and model validation rely heavily on historical project data. Therefore, when evaluating the estimation accuracy of an ML model, we should take into account the historical project data set on which the model is constructed and validated, as well as the employed validation method.

Various historical project data sets were used to construct and validate the ML models identified in this review. The most frequently used data sets together with their relevant information are shown in Table 5. The relevant information includes the type of the data set (within-company or cross-company), the number and percentage of the selected studies that use the data set, the number of projects in the data set, and the source of the data set. With respect to validation methods, *Holdout*, *Leave-One-Out Cross-Validation* (LOOCV), and *n-fold Cross-Validation* ($n > 1$) are the three dominant ones used in the selected studies. Specifically, the numbers (percentages) of the studies that used these three validation methods are 32 (38%) for Holdout, 31 (37%) for LOOCV, and 16 (19%) for *n*-fold Cross-Validation.

Besides data set and validation method, accuracy metric should also be considered in evaluating the ML models. Effort estimation accuracy can be measured with various metrics, and different metrics measure the accuracy from different aspects. Thus, to answer RQ2 properly, it is crucial to adopt appropriate accuracy metrics in evaluating estimation accuracy. It is found in the selected studies that *MMRE* (Mean Magnitude of Relative Error), *Pred (25)* (Percentage of predictions that are within 25% of the actual value), and *MdMRE* (Median Magnitude of Relative Error) are the three most popular accuracy metrics. Specifically, the numbers (percentages) of the studies that used these three metrics are 75 (89%) for MMRE, 55 (65%) for Pred (25), and 31 (37%) for MdMRE. In view of the dominance of MMRE, Pred (25), and MdMRE metrics, we adopted them in this review to evaluate the estimation accuracy of ML models.

The original values of MMRE, Pred (25), and MdMRE extracted from the selected studies are summarized in Table C.10 in Appendix C. Notice that, since GA was found to be used only for feature selection and feature weighting in combined ML models, no estimation accuracy data of GA are provided in Table C.10. To analyze the estimation accuracy graphically, we used box plots to demonstrate the distributions of MMRE, Pred (25), and MdMRE values of ML models in Fig. 5a–c, respectively. Since the observations of MMRE and Pred (25) for AR are few, we did not plot them in the corresponding figures to avoid the meaningless presentation of only one or two data points in a box plot. For the same reason, box plots of MdMRE for SVR, GP, and AR are also not presented in Fig. 5c.

A lower MMRE and MdMRE, or a higher Pred (25) indicates a more accurate estimate. On the performance of ML models measured in MMRE and Pred (25) (see Fig. 5a and b), ANN and SVR are the most accurate ones (with median MMRE around 35% and median Pred (25) around 70%), followed by CBR, DT, and GP (with both median MMRE and median Pred (25) around 50%), whereas BN has the worst accuracy (with median MMRE around 100% and median Pred (25) around 30%). On the performance of ML models measured in MdMRE (see Fig. 5c), both CBR and ANN have their median MdMRE around 30%, whereas those of BN and DT are around 45%. Apart from the above mentioned observations, according to Fig. 5a we can also see that, CBR, ANN, DT, BN, and GP all have their median MMRE nearly in the centers of the boxes, which indicates that the MMRE values of these ML models are symmetrically distributed around the medians; in contrast, the distribution of MMRE values of SVR exhibits a slight negative skewness. Furthermore, as shown in Fig. 5a, the MMRE values of CBR, ANN, SVR, and GP have less variation than those of DT and BN, since they have relatively shorter boxes and narrower ranges of values.

The box plots in Fig. 5 are useful tools to understand the estimation accuracy of ML models intuitively. To more deeply analyze the estimation accuracy of ML models, in Table 6 we also provide the detailed statistics of MMRE, Pred (25), and MdMRE for each type of ML model as a beneficial complement to the box plots. Notice that the outliers identified in Fig. 5a were removed before calculating the statistics. As can be seen in Table 6, except for BN, the rest of ML models have their arithmetic means of MMRE approximately ranging from 35% to 55%, Pred (25) from 45% to 75%, and MdMRE from 30% to 50%. This evidence indicates that the estimation accuracy of ML models is close to the acceptable level (an effort estimation model is widely considered acceptable if MMRE $\leqslant$25% and Pred (25) $\geqslant$75%). Furthermore, according to Table 6 we can find that ANN and SVR overall outperform the other ML models. Nonetheless, we should avoid drawing such a misleading conclusion that ANN and SVR are always preferable without any restriction. In fact, taking ANN as an example, if we achieve the high accuracy just by increasing the number of its hidden layers and nodes, then not only will the training time increase, but also the generalization ability will tends to degenerate, which is known as the so-called overfitting problem [54]. Therefore, to apply ANN successfully, we should address the potential overfitting problem via some methods such as cross-validation [33].

Table 5
Data sets used for ML model construction and validation (W = within-company, C = cross-company).

Data set	Type	# Of studies	Percent	# Of projects	Source
Desharnais	W	24	29	81	[47]
COCOMO	C	19	23	63	[48]
ISBSG	C	17	20	>1000[a]	[49]
Albrecht	W	12	14	24	[50]
Kemerer	W	11	13	15	[51]
NASA	W	9	11	18	[52]
Tukutuku	C	7	8	>100[a]	[53]

[a] The number of projects depends on the version of data set.

3.4. ML models vs. non-ML models (RQ3)

The ML models have been compared with five conventional non-ML models: regression model, COCOMO model [48], expert judgment, function point analysis (FPA) [55,50], and SLIM model [56]. The details of the comparisons between ML models and non-ML models are provided in Table C.11 in Appendix C. The overall results of the comparisons between ML models and non-ML models are shown in Fig. 6, where all the comparisons were conducted on the same experiments in terms of MMRE metric. One model is said to outperform another in an experiment if the MMRE value of the first model achieves at least 5% improvement over that of the second model. As can be seen in Fig. 6, the majority of the experiments indicate that ML models outperform non-ML models. Specifically, Fig. 6 shows that 66% (52 of 79) of experiment results exhibit the superiority of ML models whereas only 34% (27 of 79) of experiment results exhibit the superiority of non-ML models (note that some studies conduct more than one experiment).

As shown in Fig. 6, among the five non-ML models, regression model is the one used most frequently in comparison with the ML models. The comparison results show that CBR and ANN are more accurate than regression model, and this observation is supported by a sufficient number of experiments. For DT, it is hard to determine whether it is more accurate than regression model or not, because the number of experiments reporting that DT outperformed regression model is equal to that of experiments reporting the opposite results. For BN or SVR, no more than three

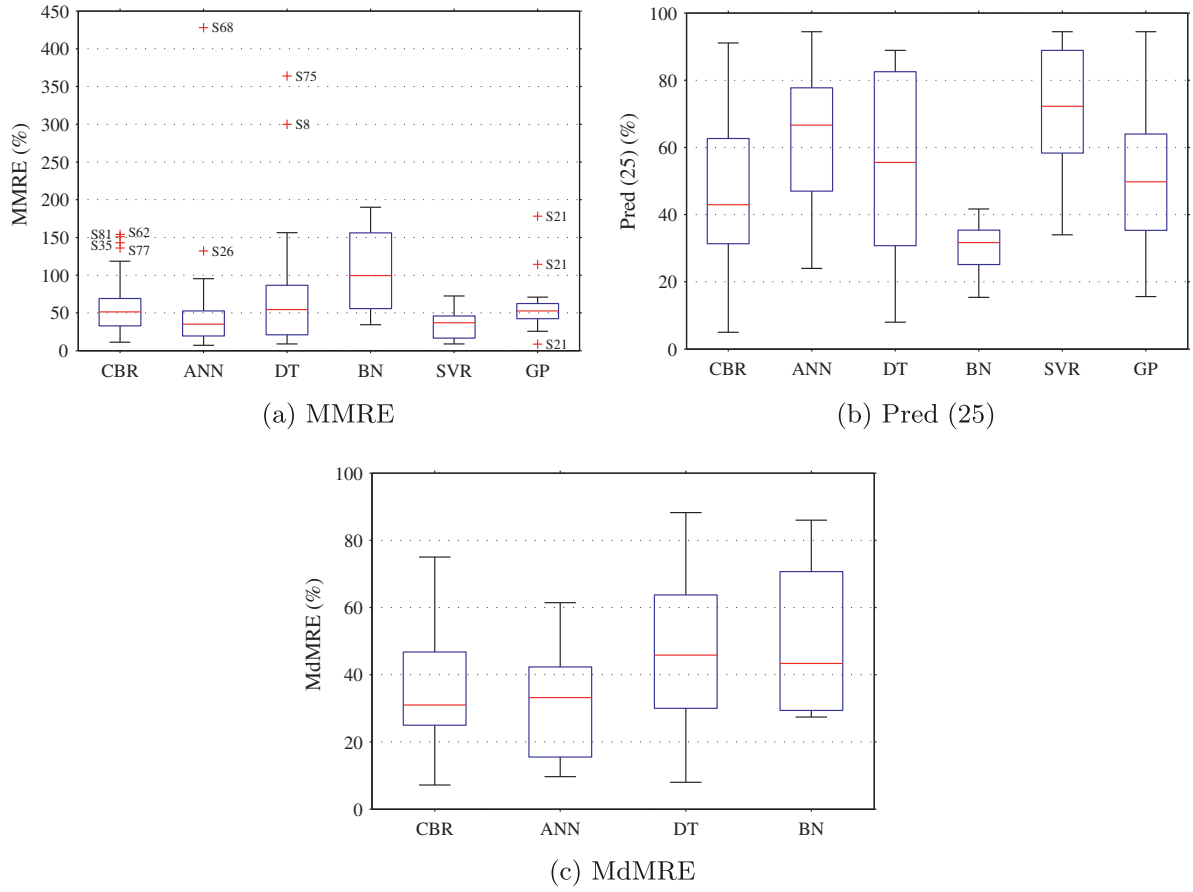

Fig. 5. Box plots of MMRE, Pred (25), and MdMRE (outliers are labeled with associated study IDs).

Table 6
Descriptive statistics of MMRE, Pred (25), and MdMRE.

Model	MMRE						Pred (25)						MdMRE					
	# Of values	Mean (%)	Median (%)	Std. dev.	Min. (%)	Max. (%)	# Of values	Mean (%)	Median (%)	Std. dev.	Min. (%)	Max. (%)	# Of values	Mean (%)	Median (%)	Std. dev.	Min. (%)	Max. (%)
CBR	57	51	49	0.26	11	119	50	46	43	0.20	5	91	38	35	31	0.17	7	75
ANN	39	37	35	0.22	7	95	32	64	67	0.21	24	94	11	32	33	0.18	10	61
DT	17	55	52	0.39	9	156	15	56	56	0.27	8	89	13	48	46	0.25	8	88
BN	6	106	99	0.59	34	190	5	30	32	0.10	15	42	4	50	43	0.27	27	86
SVR	11	34	37	0.20	9	72	11	72	72	0.21	34	94	2	40	40	0.01	40	41
GP	11	49	51	0.14	26	71	14	52	50	0.22	16	94	1	32	32	0.00	32	32
AR	2	49	49	0.27	30	69	2	57	57	0.01	56	58	0	NA	NA	NA	NA	NA

experiments reported the comparisons with regression model. For GP, although there are six experiments which show that GP is less accurate than regression model, these experiments actually come from an identical study (S21). Therefore, the small number of comparisons between the three ML models (i.e., BN, SVR, and GP) and regression models limits the generalization of the comparison results. In addition to regression model, other non-ML models have also been compared with some ML models; however, very few experiments (only one or two experiments) for these comparisons have been reported and thus the comparison results are difficult to be generalized.

According to the above results and analysis, it can be seen that ML models outperform non-ML models in general. However, this argument holds only for CBR vs. regression and ANN vs. regression, which have been supported by a sufficient number of studies. On the other hand, the lack of sufficient studies on comparisons between other ML models and non-ML models is a potential threat

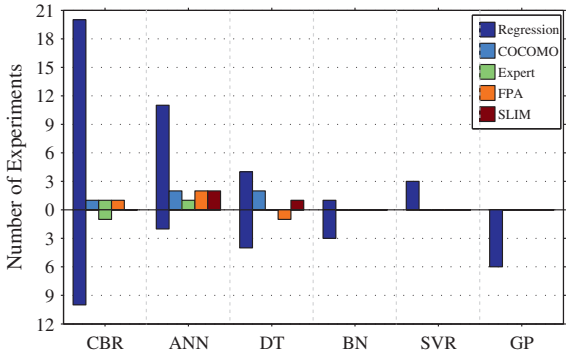

Fig. 6. Comparisons of MMRE between ML models and non-ML models (the bars above zero line indicate that ML models are more accurate, whereas the bars below zero line indicate that non-ML models are more accurate).

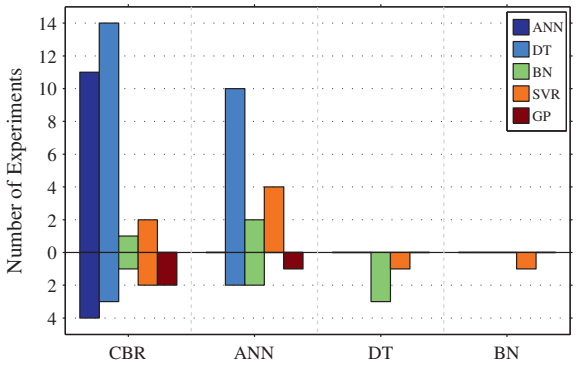

Fig. 7. Comparisons of MMRE between ML models (the bars above zero line indicate that models in horizontal axis are more accurate, whereas the bars below zero line indicate that models in horizontal axis are less accurate).

to the reliability of this argument. Finally, it is worth pointing out that several contradictory results have been reported for the comparisons between ML models and non-ML models, especially for those involving regression models, and the consensus on this issue has not yet been reached.

3.5. ML models vs. other ML models (RQ4)

As for the comparisons between different ML models, we adopted the same analysis scheme as that used in the comparisons between ML models and non-ML models. That is, the comparisons were conducted on the same experiments in terms of MMRE metric; one ML model is considered to outperform another in an experiment if it achieves at least 5% improvement in estimation accuracy. Fig. 7 shows the overall results of the comparisons between different ML models, together with the corresponding number of supporting experiments. More details of the comparison results can be found in Table C.12 in Appendix C.

Three significant comparison results can be found in Fig. 7. First, CBR and ANN are more accurate than DT, and it is supported by most of the experiments conducting the comparisons. Second, for the case of CBR vs. ANN, CBR seems more accurate than ANN. However, this finding is inconsistent with what we have found in Section 3.3, where ANN was found to perform better than CBR (see Fig. 5 or Table 6 in Section 3.3). This contradiction may be due to the following three reasons: (1) all the experiments demonstrating the superiority of CBR over ANN come from the CBR studies, so it is possible that some of them have a bias towards CBR; (2) the ANN studies rarely conducted experiments to compare ANN with CBR, so that the majority of ANN studies reporting high estimation accuracy contribute nothing to this comparison; and (3) the number of experiments concerning this comparison from CBR studies is twice as that from ANN studies. Third, for the comparisons between other ML models, the number of experiments is relatively small, and some comparisons are even found to be inconsistent. Therefore, for these ML models that are rarely compared with each other, it is difficult to determine which is more accurate.

3.6. Favorable estimation contexts of ML models (RQ5)

Given the estimation contexts of an SDEE task, we may concern ourselves with selecting ML models appropriate for the contexts. Such concern can be addressed by investigating the candidate ML models (or, more precisely, the ML techniques) from their characteristics, which are mainly reflected by the strengths and weaknesses of the ML techniques. To this aim, in this review we extracted the strengths and weaknesses of ML techniques and synthesized them based on the type of ML technique. Since different studies assessed the ML techniques in different ways and from different aspects, we decided to synthesize the common strengths and weaknesses that were mentioned by at least two studies. The detailed information on the synthesized strengths and weaknesses of different ML techniques is presented in Table C.13 in Appendix C.

According to Table C.13, we can find that the characteristics of the ML techniques are mainly associated with the following four types of estimation contexts: (1) small data set, (2) outliers, (3) categorical features, and (4) missing values. Based on these estimation contexts, we further extracted from Table C.13 the relevant characteristics of the ML techniques and summarized them in Table 7. Notice that Table 7 presents not only the information extracted from Table C.13, i.e., the information supported by at least two studies, but also the information supported by only one study. To enhance the integrity of Table 7, we make some remarks on this table. First, for the items that were not mentioned by any studies (see the items labeled with "–" in Table 7), we provide the explanations as follows: DT is prone to overfitting on small training data set; ANN and DT cannot deal with missing values. Second, CBR and ANN cannot deal with categorical features in their standard forms while they work as long as the categorical features have been quantified. Third, similarly, CBR, ANN, and DT cannot deal with missing values in their standard forms while they work as long as the missing values have been imputed.

In addition to the characteristics summarized in Table 7, some other distinct characteristics of ML techniques may be considered when choosing appropriate ML models. We summarize them based on Table C.13 as follows: CBR and DT are intuitive and are easy to understand; ANN has the ability to learn complex functions, but it requires large amount of data for training and usually suffers the problem of overfitting, and its explanatory ability is weak; BN is capable of learning causal relationships. For SVR, GP, and AR, we do not list their characteristics in Table C.13 since none of them was mentioned by more than one study.

Some existing works have discussed the topics about model selection, model application, and model combination, which are closely related to the characteristics of ML techniques and estimation contexts. With respect to model selection, Bibi and Stamelos [57] propose a tree-form framework to select appropriate ML models. The framework is designed based on the criteria of data set size, uncertainty, causality, and applicability. These criteria are preliminary and can be extended to include more criteria related to ML models. With respect to applying ML models to software engineering tasks, Zhang and Tsai [6] propose a general procedure that consists of the following steps: problem formulation, problem representation, data collection, domain theory preparation, performing the learning process, analyzing and evaluating learned knowledge, and fielding the knowledge base. There is no doubt that this procedure is also applicable to software effort estimation tasks. With respect to model combination, MacDonell and Shepperd [58] argue that combining a set of diverse techniques can improve estimation accuracy if no dominant technique can be found. An SLR conducted by Jørgensen [27] also found that combined model usually produces better estimate than individual model does. The findings of [58] and [27] are worth further investigation

Table 7
Favorable/unfavorable estimation contexts of ML models ("√" means favorable, "×" means unfavorable, "–" means not mentioned).

	Small data set	Outliers	Categorical features	Missing values
CBR	√	√	×	×
ANN	×	√	×[a(S68)]	–
DT	–	√	√	–
BN	√	√[a(S78)]	√[a(S78)]	√[a(S67)]

[a] Supported by only one study (the study ID is shown in parentheses).

in the context of ML models. It is believed that combining two or more ML techniques may have the potential ability to enhance the power of estimation model. Also, several studies [59–62] have provided strong support for this possibility.

Although ML techniques have been proved in some studies to be effective for SDEE, they do not always perform well on all SDEE tasks. In fact, an absolutely "best" estimation method does not appear to exist, and the performance of a particular estimation method depends heavily on the contexts it applies to. In order to choose ML techniques properly and apply them to real-world SDEE tasks efficiently, practitioners need to understand not only the characteristics of the candidate ML techniques, but also the estimation contexts of the SDEE tasks. As pointed out in [63], choosing the best estimation method "in a particular context" is more fruitful than choosing the "best" estimation method.

4. Implications for research and practice

This review has found that the empirical studies on the application of BN, SVR, GA, GP, and AR techniques are scarce; some ML techniques have not even been applied in SDEE domain. Therefore, researchers are encouraged to conduct more empirical studies on these rarely-used ML techniques to further strengthen the empirical evidence about their performance. Moreover, researchers are also encouraged to explore the possibilities of using the unapplied ML techniques to estimate software development effort. In order to look for the unapplied ML techniques and to use them more efficiently, researchers had better keep track of the related disciplines such as machine learning, data mining, statistics, and artificial intelligence, since these disciplines may provide valuable ideas and methods to address SDEE issues.

High-quality historical software project data set with detailed descriptions of project features and data collection process is critical for constructing and validating ML model. This review has revealed that, on one hand, most of the available project data sets are obsolete, and the numbers of projects in these data sets are small; on the other hand, project data sets are difficult to collect and maintain, and usually contain confidential information. To address these issues and thus to promote SDEE research, we suggest that researchers share their proprietary project data sets in research community after confidential information being removed. If needed, PROMISE repository [64] is a convenient place for sharing SDEE data sets.

Although this review has found that ML models are usually more accurate than non-ML models, the results of accuracy comparisons between some ML models and non-ML models and between some different ML models are still inconclusive. The limited number of relevant studies and the non-uniform experimental designs may account for these inconclusive results. Therefore, besides conducting more experiments, it will be beneficial for research community to develop a uniform experimental framework for evaluating the performance of different estimation models. Indeed, without using a uniform framework, the comparison results may vary when employing different data sets or different experimental designs.

As to the implications for practitioners, this review has found that very few of the selected studies focus on industry practice (only one case study paper, and no survey paper, was found). This evidence may imply that the application of ML models in real-world industry is quite limited. Accordingly, we suggest that practitioners may cooperate with researchers to investigate the possibility of applying the promising ML models in their practices. For example, CBR and ANN have been investigated most extensively in academia, so practitioners can first take them into consideration as a useful complement to existing conventional estimation model. Due to the inconsistent results of comparisons between ML models and conventional non-ML models found in this review, we recommend using both types of models in parallel at the early stage of practice. Only when the ML model performs significantly and consistently better than the existing model, can practitioners consider replacing the existing model with the focused ML model and switch to the new estimation system.

This review has found that both within-company and cross-company data sets are used to construct and validate ML models. For constructing accurate estimation models, within-company data sets have been shown to be more effective than cross-company data sets by some studies [8,65–67]. Thus, practitioners are suggested to use within-company data sets to construct ML models if their companies have sufficient historical project data. For the companies that do not have sufficient historical project data, they had better adopt other companies' data sets or cross-company data sets that come from similar application domains. In this case, the estimation results should be interpreted by experts and dealt with carefully.

As has been shown in this review, different ML techniques favor different estimation contexts. Therefore, before making any decision on the choice of ML models, practitioners not only need to be aware of the estimation contexts, but also need to understand the characteristics of the candidate ML models. Usually, the degree of match between the estimation contexts and the characteristics of the chosen ML model has a direct and significant impact on the performance of the model.

5. Limitations of this review

This review considered only accuracy metrics (e.g., MMRE) when evaluating the performance of ML models or comparing ML models with other models. Accuracy metrics are the most important metrics and were used by most of the studies. Usually, practitioners will reject a model if it fails to reach the minimum threshold of accuracy [17]. However, besides accuracy metrics, other performance metrics such as generalization ability and interpretability, which were ignored in this review, may also be necessary to be considered, especially when selecting appropriate models for given SDEE tasks. Fortunately, the summarized strengths and weaknesses of ML models, i.e., the outcomes of RQ5, are helpful to identify the appropriate ML models, which may alleviate this limitation to some extent.

The estimation accuracy data of ML models were extracted from the studies with varying experimental designs. Therefore, it may be difficult to explain the synthesized estimation accuracy results about under what conditions and to what extent these results hold. In general, experimental design involves choosing data set to construct model and employing validation method to validate model. Furthermore, experimental design may also involve the steps of data preprocessing to remove outliers, to weight features, or to select feature subsets. All these elements of experimental design have impacts on estimation accuracy. Nevertheless, from another perspective, the synthesized estimation accuracy results from the studies with different experimental designs are believed more robust than that from the studies with identical experimental design.

This review has revealed that some results of the comparisons between ML models and conventional non-ML models and between different ML models are inconsistent, so it is difficult to determine which model is more accurate. The insufficient number of the studies that report the desired comparisons may have contributed to these inconsistencies. Generally speaking, the conclusion drawn from a large number of studies is likely to be more reliable [44]. It has been agreed that the issue of insufficient studies for evaluating new software engineering techniques is common in the field of empirical software engineering. Besides SDEE models, many other prediction models in software engineering also suffer from this issue [68].

Table A.8
Selected studies.

ID	Author	Research questions addressed					Ref.	ID	Author	Research questions addressed					Ref.
S1	L. Angelis et al.	1	2	3		5	[69]	S2	R.A. Araújo et al.	1	2		4		[70]
S3	M. Auer et al.	1					[71]	S4	M. Azzeh et al.	1	2			5	[72]
S5	M. Azzeh et al.	1	2				[73]	S6	M. Azzeh et al.	1	2				[74]
S7	S. Berlin et al.	1	2	3			[75]	S8	S. Bibi et al.	1	2			5	[61]
S9	S. Bibi et al.	1				5	[76]	S10	P.L. Braga et al.	1	2		4	5	[77]
S11	P.L. Braga et al.	1	2				[78]	S12	L.C. Briand et al.	1	2	3		5	[37]
S13	L.C. Briand et al.	1	2	3	4		[15]	S14	L.C. Briand et al.	1	2	3	4	5	[79]
S15	L.C. Briand et al.	1	2	3			[80]	S16	C.J. Burgess et al.	1	2		4		[81]
S17	N.-H. Chiu et al.	1	2	3	4		[18]	S18	A. Corazza et al.	1	2	3	4		[82]
S19	A. Corazza et al.	1	2	3	4		[83]	S20	G. Costagliola et al.	1	2				[84]
S21	J.J. Dolado	1	2	3			[10]	S22	M.O. Elish	1	2	3	4	5	[4]
S23	F. Ferrucci et al.	1	2		4		[85]	S24	G.R. Finnie et al.	1	2	3		5	[86]
S25	A.R. Gray et al.	1	2	3		5	[13]	S26	A.R. Gray et al.	1	2	3		5	[7]
S27	A. Heiat	1	2				[9]	S28	S.-J. Huang et al.	1	2	3	4	5	[60]
S29	S.-J. Huang et al.	1	2				[87]	S30	S.-J. Huang et al.	1					[88]
S31	S.-J. Huang et al.	1					[89]	S32	X. Huang et al.	1				5	[90]
S33	A. Idri et al.	1				5	[91]	S34	A. Idri et al.	1	2				[92]
S35	R. Jeffery et al.	1		3			[8]	S36	R. Jeffery et al.	1	2	3	4		[65]
S37	M. Jørgensen et al.	1	2				[93]	S38	E.S. Jun et al.	1	2	3	4		[94]
S39	G. Kadoda et al.	1	2				[95]	S40	J.W. Keung et al.	1					[96]
S41	Y. Kultur et al.	1	2		4	5	[97]	S42	K.V. Kumar et al.	1	2	3	4		[98]
S43	T.K. Le-Do et al.	1	2				[99]	S44	A. Lee et al.	1	2				[100]
S45	J. Li et al.	1					[101]	S46	J. Li et al.	1	2				[102]
S47	J. Li et al.	1	2			5	[103]	S48	Y.F. Li et al.	1	2	3	4		[104]
S49	Y.F. Li et al.	1	2		4		[105]	S50	Y.F. Li et al.	1	2	3	4	5	[106]
S51	S.G. MacDonell et al.	1	2	3			[58]	S52	C. Mair et al.	1	2	3	4	5	[17]
S53	E. Mendes	1	2	3	4		[107]	S54	E. Mendes et al.	1	2				[67]
S55	E. Mendes et al.	1	2	3			[108]	S56	E. Mendes et al.	1					[109]
S57	E. Mendes et al.	1	2	3	4		[110]	S58	E. Mendes et al.	1					[2]
S59	N. Mittas et al.	1	2				[111]	S60	N. Mittas et al.	1	2				[112]
S61	T. Mukhopadhyay et al.	1	2	3		5	[36]	S62	I. Myrtveit et al.	1	2				[11]
S63	I. Myrtveit et al.	1					[68]	S64	A.L.I. Oliveira	1	2	3			[113]
S65	A.L.I. Oliveira et al.	1	2			5	[62]	S66	H. Park et al.	1	2	3		5	[114]
S67	P.C. Pendharkar et al.	1	2		4	5	[115]	S68	B. Samson et al.	1	2	3		5	[116]
S69	Y.-S. Seo et al.	1	2	3	4		[117]	S70	R. Setiono et al.	1	2	3	4	5	[118]
S71	M. Shepperd et al.	1					[63]	S72	M. Shepperd et al.	1	2	3		5	[12]
S73	M. Shin et al.	1	2				[119]	S74	K.K. Shukla	1					[59]
S75	K. Srinivasan et al.	1	2	3	4		[120]	S76	I. Stamelos et al.	1				5	[121]
S77	E. Stensrud et al.	1	2	3			[122]	S78	B. Stewart	1	2		4	5	[123]
S79	I.F.B. Tronto et al.	1	2	3			[3]	S80	F. Walkerden et al.	1	2	3		5	[14]
S81	J. Wen et al.	1	2			5	[124]	S82	I. Wieczorek et al.	1	2				[125]
S83	G. Wittig et al.	1	2			5	[126]	S84	G.E. Wittig et al.	1	2				[127]

The strengths and weaknesses of ML techniques were extracted directly from the selected studies. Therefore, it is possible that some of them may just represent the authors' opinions and thus may be unreliable. To increase the reliability of the synthesized results for RQ5, we present only the synthesized strengths and weaknesses that are supported by two or more selected studies. In addition, the process of study quality assessment can ensure that these strengths and weaknesses are from the studies with acceptable quality. Even so, care has to be taken in dealing with any inferences drawn from these synthesized results.

6. Conclusion and future work

This systematic review investigated machine learning (ML) based software development effort estimation (SDEE) models, i.e., ML models for short, from four perspectives: the type of ML technique, the estimation accuracy of ML model, the comparison between different models (including ML model vs. non-ML model and ML model vs. other ML model), and the estimation contexts of ML model. We have carried out an extensive literature search for relevant studies published in the period 1991–2010 and finally identified 84 primary empirical studies that are pertaining to the five research questions (RQs) raised in this review. The principal findings of this review are summarized as follows.

- (RQ1) The ML techniques that have been applied in SDEE include case-based reasoning (CBR), artificial neural networks (ANN), decision trees (DT), Bayesian networks (BN), support vector regression (SVR), genetic algorithms (GA), genetic programming (GP), and association rules (AR). Among them, CBR, ANN, and DT are used most frequently.
- (RQ2) The overall estimation accuracy of most ML models is close to the acceptable level. More specifically, most of the ML models have their arithmetic means of MMRE approximately ranging from 35% to 55%, Pred (25) from 45% to 75%, and MdMRE from 30% to 50%. Furthermore, the means of MMRE of ANN, CBR, and DT are about 35%, 50%, and 55%, respectively.
- (RQ3) ML model is more accurate than non-ML model in general, which is supported by most of the studies. Regression model is the non-ML model that is most often compared with ML models.
- (RQ4) Both CBR and ANN are more accurate than DT, which are supported by most of the studies conducting the comparisons.
- (RQ5) Different ML techniques have different strengths and weaknesses and thus favor different estimation contexts. In the identified studies, four types of estimation contexts are mentioned most often: small data set, outliers, categorical features, and missing values.

Table B.9
Quality scores of selected studies.

ID	QA1	QA2	QA3	QA4	QA5	QA6	QA7	QA8	QA9	QA10	Score
S1	1	0.5	1	1	1	1	1	1	0	1	8.5
S2	1	0.5	1	1	1	1	1	1	0	1	8.5
S3	1	0	1	1	1	1	0.5	1	0	1	7.5
S4	1	0	1	0	0.5	1	0.5	1	1	0.5	6.5
S5	1	0	1	0	1	1	0.5	1	0.5	0.5	6.5
S6	1	0	1	0	1	1	0.5	1	0	1	6.5
S7	1	1	0.5	1	1	1	1	1	1	1	9.5
S8	1	1	1	1	1	1	0	1	1	1	9
S9	1	1	1	1	1	1	0	1	0	1	8
S10	1	0	0.5	0.5	1	1	0.5	0.5	0	1	6
S11	1	0	0.5	0.5	0.5	1	0.5	0.5	0	1	5.5
S12	1	0	1	0.5	1	1	1	1	0	1	7.5
S13	1	0.5	0.5	1	0.5	1	1	1	0	1	7.5
S14	1	0.5	0.5	1	0.5	1	1	1	0	1	7.5
S15	1	1	1	1	0.5	1	0.5	1	1	1	9
S16	1	0.5	1	0	0.5	1	1	1	0	1	7
S17	1	0	1	0.5	1	1	1	1	0	1	7.5
S18	1	0.5	1	1	0.5	1	1	1	1	1	9
S19	1	0.5	0.5	0.5	0.5	1	1	1	0	1	7
S20	1	0.5	0.5	0.5	0.5	1	1	0.5	0	1	6.5
S21	1	0	0.5	0.5	1	1	1	1	0	1	7
S22	1	0	0.5	0.5	0.5	1	1	1	0	1	6.5
S23	1	0	0.5	0.5	0.5	1	1	1	0	1	6.5
S24	1	0	0.5	0.5	0.5	1	1	0.5	0	1	6
S25	1	0	1	0	0.5	0.5	1	0.5	0	1	5.5
S26	1	0	0.5	0.5	1	1	1	1	0	1	7
S27	1	0	0.5	0.5	1	1	1	1	0	1	7
S28	1	0.5	0.5	1	1	1	1	1	0	1	8
S29	1	0	1	1	0.5	1	1	1	0	1	7.5
S30	1	0	1	0.5	1	1	1	1	0	1	7.5
S31	1	0	0.5	0.5	0.5	1	0.5	0.5	0	1	5.5
S32	1	0	1	1	0.5	1	0.5	1	0	1	7
S33	1	0	1	0	0.5	1	1	1	0	1	6.5
S34	1	0	1	0	1	1	0	0.5	0	1	5.5
S35	1	1	0.5	0.5	1	1	1	1	0	1	8
S36	1	0.5	0.5	0.5	0.5	1	1	1	0	1	7
S37	1	0	1	1	1	1	0.5	1	0	1	7.5
S38	1	0	1	1	0.5	1	0.5	1	0	1	7
S39	1	0	0	1	0.5	1	0.5	1	0	1	6
S40	1	0	1	1	0.5	0	0	1	1	1	6.5
S41	1	0	1	0.5	1	1	1	1	0	1	7.5
S42	1	0	1	1	1	1	1	1	0	1	8
S43	1	0	1	0.5	1	1	0.5	1	0	1	7
S44	1	0	0.5	1	0.5	1	0.5	1	0	1	6.5
S45	1	0.5	0.5	0.5	1	1	0	0.5	1	1	7
S46	1	0	1	1	1	1	1	1	1	1	9
S47	1	1	1	1	1	1	1	1	0	1	9
S48	1	0	1	1	1	1	1	1	0	1	8
S49	1	0	1	1	1	1	1	1	0	1	8
S50	1	0.5	1	1	1	1	1	1	1	1	9.5
S51	1	0	0	0.5	0.5	1	1	1	0	1	6
S52	1	0	0.5	0	0.5	1	1	1	0	1	6
S53	1	0	1	0.5	0.5	1	1	1	0	1	7
S54	1	0	0	1	0.5	1	1	1	0	1	6.5
S55	1	1	0	1	0.5	1	1	1	1	1	8.5
S56	1	0.5	0.5	1	0.5	1	1	1	0	1	7.5
S57	1	0	1	1	0.5	1	1	1	1	1	8.5
S58	1	0	0.5	1	0.5	1	1	1	1	1	8
S59	1	0.5	1	1	1	1	0.5	1	0	1	8
S60	1	0	1	1	1	1	0	1	0	1	7
S61	1	0	1	0.5	0.5	1	1	1	1	1	8
S62	1	0.5	0.5	1	0.5	1	1	1	0	1	7.5
S63	1	0	0	1	0.5	1	1	1	1	1	7.5
S64	1	0	0.5	0.5	0.5	1	1	1	0	1	6.5
S65	1	0	1	1	1	1	1	1	0	1	8
S66	1	0	0.5	1	0.5	1	1	1	0	1	7
S67	1	0	0.5	0.5	0.5	1	0	1	0	1	5.5
S68	1	0	1	0.5	0.5	1	1	1	0	1	7
S69	1	0	0.5	1	0.5	1	1	1	0	1	7
S70	1	0	0.5	1	0.5	1	1	1	0	1	7
S71	1	1	0.5	1	0.5	1	1	1	0	1	8
S72	1	0	1	0.5	1	1	1	1	0	1	7.5
S73	1	0	1	1	0.5	1	0	1	0	1	6.5
S74	1	0	1	1	0.5	1	1	1	0	1	7.5

(*continued on next page*)

Table B.9 (*continued*)

ID	QA1	QA2	QA3	QA4	QA5	QA6	QA7	QA8	QA9	QA10	Score
S75	1	0	0.5	1	1	1	1	1	1	1	8.5
S76	1	0	1	0	0.5	1	0	1	0	1	5.5
S77	1	0	0	0.5	0.5	1	1	1	1	1	7
S78	1	0	1	1	0.5	1	1	1	0	1	7.5
S79	1	0	0.5	1	0.5	1	1	1	0	1	7
S80	1	0	0.5	1	0.5	1	1	1	0	1	7
S81	1	0	1	0.5	1	1	1	0.5	0	1	7
S82	1	0	0.5	0.5	0.5	1	1	1	0	1	6.5
S83	1	0	0.5	0.5	1	1	0	1	0	1	6
S84	1	0	0.5	0.5	0.5	1	0	1	0	1	5.5
Total	84	15.5	60.5	59.5	59.5	82.5	65.5	79.5	16.5	83	606
Average	1	0.18	0.72	0.71	0.71	0.98	0.78	0.95	0.20	0.99	7.21

Table C.10
Estimation accuracy values of ML models, together with the corresponding data sets in which the models were trained and tested (grouped by the type of ML model; "+" means combining data sets, "–" means not applicable).

ID	MMRE (%)	Pred (25) (%)	MdMRE (%)	Data set	ID	MMRE (%)	Pred (25) (%)	MdMRE (%)	Data set
CBR									
S1	31.00[b]	72.00[b]	–	Albrecht	S49	30.00[b]	63.00[b]	27.00[b]	Albrecht
S4	13.55[b]	84.00[b]	–	ISBSG	S49	32.00[b]	44.00[b]	29.00[b]	Desharnais
S5	28.70[b]	54.70[b]	21.80[b]	ISBSG	S50	41.00[b]	50.00[b]	25.00[b]	Albrecht
S5	38.50[b]	42.40[b]	31.70[b]	Desharnais	S50	52.00[b]	36.00[b]	32.00[b]	Desharnais
S6	11.30[b]	91.10[b]	7.20[b]	Desharnais	S50	77.00[b]	35.00[b]	45.00[b]	Maxwell
S6	19.90[b]	70.00[b]	13.90[b]	COCOMO	S50	74.00[b]	31.00[b]	42.00[b]	ISBSG
S13	106.33[a]	19.67[a]	55.00[a]	Finland	S51	54.00[a]	–	–	Medical
S14	–	–	75.00[a]	ESA	S52	57.00	–	–	Desharnais
S14	–	–	56.67[a]	Laturi	S54	100.00	25.40	45.00	Tukutuku
S17	36.00[b]	61.00[b]	19.00[b]	Albrecht	S55	111.93	31.33	45.45	Tukutuku
S17	49.00[b]	43.00[b]	31.00[b]	CF	S59	118.50[a]	29.22[a]	48.08[a]	ISBSG
S20	21.00[b]	73.00[b]	8.00[b]	Web applications	S59	54.36[a]	47.31[a]	25.98[a]	NASA93
S24	36.20	42.00	–	Desharnais	S60	36.15[b]	–	14.23[b]	CF
S28	67.00[b]	30.00[b]	50.00[b]	ISBSG	S60	49.38[b]	–	28.92[b]	COCOMO
S28	33.00[b]	70.00[b]	19.00[b]	Albrecht	S60	68.50[b]	–	46.75[b]	Finnish
S35	37.00[b]	47.00[b]	28.00[b]	Megatec	S61	52.79	–	–	Kemerer
S35	143.00[b]	5.00[b]	72.00[b]	ISBSG	S62	154.00	–	52.00	COTS
S36	30.50	58.00	20.80	ISBSG	S72	62.00	33.00	–	Albrecht
S36	114.50	17.00	70.10	ISBSG	S72	39.00	38.00	–	Atkinson
S37	31.00	–	26.00	Jeffery & Stathis	S72	64.00	36.00	–	Desharnais
S37	39.00	–	31.00	Jørgensen97	S72	41.00	39.00	–	Finnish
S39	55.40[a]	–	–	Desharnais	S72	62.00	40.00	–	Kemerer
S43	53.86[b]	42.86[b]	30.98[b]	Desharnais	S72	78.00	21.00	–	Mermaid
S43	71.31[b]	29.03[b]	47.86[b]	Maxwell	S72	74.00	23.00	–	Real-time 1
S43	86.62[b]	37.42[b]	36.20[b]	ISBSG	S72	39.00	44.00	–	Telecom 1
S46	26.00[b]	72.00[b]	–	ISBSG	S72	37.00	51.00	–	Telecom 2
S46	19.00[b]	83.00[b]	–	Kemerer	S77	136.00	–	–	COTS
S46	59.00[b]	42.00[b]	–	Desharnais	S80	55.00	24.00	–	Australian
S47	16.00	81.82	–	ISBSG	S81	151.00	21.00	–	COCOMO
S48	36.00[b]	43.00[b]	28.00[b]	Desharnais	S81	62.00	43.00	–	Desharnais
S48	28.00[b]	67.00[b]	19.00[b]	Maxwell	S81	26.00	67.00	–	NASA
					S82	51.33[a]	62.67[a]	38.83[a]	Laturi
ANN									
S2	9.81	90.00	–	Desharnais	S42	19.82[b]	66.64[b]	16.38[b]	CF
S2	12.98	90.90	–	COCOMO	S42	7.18[b]	87.50[b]	9.70[b]	Albrecht
S7	52.00[b]	56.38[b]	–	Israeli	S44	46.75[a]	–	–	COCOMO
S10	39.91	66.67	–	Desharnais	S52	47.00	–	–	Desharnais
S10	17.71	94.44	–	NASA	S65	31.54	72.22	–	Desharnais
S11	16.39	88.89	–	NASA	S65	19.50	94.44	–	NASA
S24	35.20	44.00	–	Desharnais	S65	21.94[a]	78.74[a]	–	COCOMO
S25	44.00	63.00	–	Desharnais	S65	68.63	61.67	–	Albrecht
S26	132.00	50.00	–	Miyazaki	S65	33.49	64.00	–	Kemerer
S26	35.00	38.00	–	Dolado	S65	12.19	92.94	–	Koten & Gray
S27	46.94[a]	–	–	Kemerer + Albrecht	S66	59.40[b]	–	–	Korean
S29	22.00	75.00	12.00	COCOMO	S68	428.11	–	–	COCOMO
S34	73.88	73.58	–	Tukutuku	S69	62.32[b]	35.56[b]	36.44[b]	ISBSG
S34	29.81	73.81	–	COCOMO	S69	27.72[b]	70.83[b]	15.23[b]	Bank in Korea
S38	12.13	–	–	Samsung	S70	95.39	30.56	57.25	ISBSG
S41	47.66[a]	38.00[a]	33.17[a]	NASA	S73	18.70	72.22	–	NASA
S41	54.35[a]	32.67[a]	36.91[a]	NASA93	S75	70.00	–	–	COCOMO + Kemerer
S41	85.44[a]	24.00[a]	61.42[a]	USC	S79	41.53[a]	–	–	COCOMO
S41	39.88[a]	55.00[a]	29.10[a]	SDR	S83	27.20	53.00	–	Desharnais

Table C.10 (*continued*)

ID	MMRE (%)	Pred (25) (%)	MdMRE (%)	Data set	ID	MMRE (%)	Pred (25) (%)	MdMRE (%)	Data set
S41	49.82[a]	33.08[a]	44.13[a]	Desharnais	S83	17.00	76.70	–	ASMA
					S84	7.17[a]	–	–	4GL Projects
DT									
S8	300.00	26.30	–	STTF	S36	76.00	25.00	45.10	ISBSG
S8	106.61	33.00	–	ISBSG	S36	156.30	8.00	131.60	ISBSG
S10	60.54	55.56	–	Desharnais	S52	90.00[b]	–	–	Desharnais
S10	16.39	88.89	–	NASA	S65	59.45	61.11	–	Desharnais
S12	50.00[b]	–	–	COCOMO + Kemerer	S65	18.38	83.33	–	NASA
S13	54.30[a]	30.00[a]	43.73[a]	Finland	S65	28.95[a]	71.71[a]	–	COCOMO
S14	–	–	60.00[a]	ESA	S65	47.00	45.83	–	Albrecht
S14	–	–	43.67[a]	Laturi	S65	52.31	46.67	–	Kemerer
S15	74.40	–	31.70	LIOO	S65	11.64	88.24	–	Koten & Gray
S20	17.00[b]	80.00[b]	11.00[b]	Web applications	S75	364.00	–	–	COCOMO + Kemerer
S22	8.97[b]	88.89[b]	–	NASA					
BN									
S53	34.30	33.30	27.40	Tukutuku	S69	104.68[b]	28.44[b]	55.37[b]	ISBSG
S57	190.00[b]	15.38[b]	86.00[b]	Tukutuku	S69	55.69[b]	41.67[b]	31.32[b]	Bank in Korea
S67	156.00	–	–	Subramanian	S78	94.00[a]	31.67[a]	–	ISBSG
SVR									
S10	46.36	55.56	–	Desharnais	S65	36.85[b]	72.22[b]	–	Desharnais
S10	17.17	88.89	–	NASA	S65	16.50[b]	94.44[b]	–	NASA
S18	59.00[b]	34.00[b]	41.00[b]	Tukutuku	S65	20.76[b]	81.76[b]	–	COCOMO
S19	72.30[a]	39.55[a]	39.70[a]	Tukutuku	S65	44.65[b]	70.42[b]	–	Albrecht
S64	16.50[b]	88.89[b]	–	NASA	S65	36.95[b]	66.67[b]	–	Kemerer
					S65	8.95[b]	94.12[b]	–	Koten & Gray
GP									
S16	44.55	23.30	–	Desharnais	S21	42.30	46.90	–	Dolado
S21	25.60	77.30	–	CF	S21	58.40	62.50	–	Kemerer
S21	54.80	64.00	–	Albrecht	S21	50.60	47.90	–	Miyazaki
S21	26.90	73.70	–	NASA	S21	45.60	57.90	–	Telecom 1
S21	71.00	35.30	–	Belady & Lehman	S21	62.30	51.60	–	Desharnais
S21	178.10	15.60	–	COCOMO	S21	114.30	32.40	–	Kitchenham
S21	8.70	94.40	–	Heiat & Heiat	S23	58.00[b]	43.00[b]	32.00[b]	Desharnais
AR									
S8	68.76	58.00	–	STTF	S8	29.88	56.00	–	ISBSG

[a] Mean of accuracy values.
[b] Accuracy value under optimal model configuration.

Table C.11
Comparisons of MMRE between ML models and non-ML models ("+" indicates ML model outperforms non-ML model, "−" indicates non-ML model outperforms ML model; the number following study ID is MMRE improvement in percentage, the study ID in bold indicates the improvement exceeds 5%).

ML model		Non-ML model				
		Regression	COCOMO	Expert	FPA	SLIM
CBR	+	**S1**(6), **S17**(36), **S17**(39), **S24**(26.1), **S28**(123), **S28**(37), **S48**(26), **S50**(53), **S50**(21), **S50**(29), **S50**(8), **S52**(5), **S55**(8.81), **S72**(28), S72(1), S72(2), **S72**(60), **S72**(45), **S72**(148), **S72**(47), **S72**(35), **S80**(13), S82(4)	**S61**(566.2)	**S51**(27.2)	**S61**(49.95)	NA
	−	**S1**(18), **S13**(43.78), **S14**(45.67), **S14**(12.67), S20(4), **S35**(82), **S36**(5.1), **S36**(29.7), **S48**(9), **S51**(9.65), S54(1), **S77**(10)	NA	**S61**(22.07)	NA	NA
ANN	+	S11(4.64), **S24**(27.1), **S25**(41), S27(3.8), **S38**(13.97), **S42**(64.82), **S42**(68.18), **S52**(15), **S66**(91), **S68**(92.6), **S69**(10.69), S69(3.89), **S70**(39.64), **S79**(16.33)	S29(4), **S75**(540), **S79**(568.47)	**S66**(17.2)	**S75**(33), **S79**(61.47)	**S75**(702), **S79**(730.47)
	−	**S7**(46.89), **S26**(56), S26(4)	NA	NA	NA	NA
DT	+	S11(4.64), **S12**(113), **S13**(8.25), S14(0.33), **S15**(22.1), **S22**(14.33)	**S12**(22), **S75**(246)	NA	NA	**S75**(408)
	−	**S14**(30.67), S20(1), **S36**(50.6), **S36**(71.5), **S52**(28)	NA	NA	**S75**(261)	NA
BN	+	**S53**(60.5)	NA	NA	NA	NA
	−	**S57**(40), **S69**(31.67), **S69**(24.08)	NA	NA	NA	NA
SVR	+	S11(3.86), **S18**(91), **S19**(39.2), **S64**(6.8)	NA	NA	NA	NA
	−	NA	NA	NA	NA	NA
GP	+	S16(1.63), S21(0.22), S23(4)	NA	NA	NA	NA
	−	S21(1.96), S21(1.67), S21(2.45), **S21**(8.42), **S21**(44.5), S21(3.42), **S21**(14.05), **S21**(10.61), S21(0.71), **S21**(8.02), **S21**(29.72)	NA	NA	NA	NA
AR	+	NA	NA	NA	NA	NA
	−	NA	NA	NA	NA	NA

Table C.12
Comparisons of MMRE between different ML models ("+" indicates the model given in the row outperforms the model given in the column, "−" indicates the model given in the column outperforms the model given in the row; the number following study ID is MMRE improvement in percentage, the study ID in bold indicates the improvement exceeds 5%).

ML model		ML model					
		ANN	DT	BN	SVR	GP	AR
CBR	+	**S17**(54), **S17**(21), **S28**(103), **S28**(71), **S48**(11), **S49**(19), **S49**(10), **S50**(44), **S50**(15), **S50**(52), **S50**(22)	**S17**(41), **S17**(40), **S28**(122), **S28**(34), **S36**(45.5), **S36**(41.8), **S48**(54), **S49**(140), **S49**(20), **S50**(103), **S50**(19), **S50**(72), **S50**(33), **S52**(33)	**S57**(316)	**S49**(15), **S49**(8)	NA	NA
	−	S24(1), **S38**(10.57), **S42**(28.82), **S42**(29.18), **S52**(10)	**S13**(52.03), **S14**(15), **S14**(13), S20(4)	**S53**(100.4)	**S18**(447), **S19**(517.2)	**S16**(117.75), **S23**(14)	NA
ANN	+		**S10**(20.63), **S41**(18.36), **S41**(339.99), **S41**(237.79), **S41**(394.89), **S41**(69.54), **S42**(69.82), **S42**(69.18), **S52**(43), **S75**(294)	**S69**(42.36), **S69**(27.97)	**S2**(21.73), S2(1.38), **S10**(6.45), S11(0.78), **S42**(41.39), **S42**(36.75)	NA	NA
	−		S10(1.32), **S22**(10.1), **S70**(34.23)	**S67**(112), **S78**(34.67)	S64(2.57)	**S16**(16.08)	NA
DT	+			NA	S10(1.32), S11(0.78), S22(2.42)	NA	NA
	−			**S53**(656.1), **S67**(81), **S78**(22.57)	**S10**(14.18)	NA	NA
BN	+				NA	NA	NA
	−				**S19**(514.7)	NA	NA
SVR	+					NA	NA
	−					NA	NA
GP	+						NA
	−						NA

Table C.13
Strengths and weaknesses of ML techniques (those of SVR, GP and AR are not listed because none of them is supported by more than one study).

Strength		Weakness	
Items	Supporting studies	Items	Supporting studies
CBR			
• Intuitive and easy to understand	S9, S24, S33, S61, S72, S80, S81	• Sensitive to similarity function	S25, S47, S80
• Be able to deal with poorly understood domains that are difficult to model	S47, S52, S72, S80, S81	• Intolerant of irrelevant features	S4, S25, S28
• Reasoning similar to human problem solving, so users are more willing to accept it	S1, S28, S47, S61, S72	• Difficult to handle categorical features	S4, S33, S47
• Appropriate in domains where theoretical model is unavailable or difficult to model	S33, S61, S72, S80	• Cannot deal with missing values	S4, S47
• Tolerate with outliers	S47, S80, S81		
• Perform well even with small data sets	S1, S14, S52		
ANN			
• Can learn complex (non-linear) functions	S24, S66, S68, S83	• Weak explanatory ability	S8, S9, S10, S24, S25, S32, S52, S66, S68, S70
• Capable of dealing with noisy data	S24, S66	• Prone to get overfitting to the training data	S8, S9, S41
		• Sensitive to network architecture and parameter setting	S41, S66, S83
		• Require plentiful data for training	S26, S41
DT			
• Intuitive and easy to understand	S12, S25, S50, S65		
• Be able to deal with categorical features	S12, S50		
• Be able to avoid overfitting by pruning	S8, S22		
• Tolerate with outliers	S12, S22		
BN			
• Capable of combining historical data with expert opinion	S9, S67, S76		
• Capable of learning causal relationships	S9, S67		
• Capable of avoiding overfitting the training data	S67, S78		
• Perform well even with small data sets	S76, S78		

This review provides recommendations for researchers as well as guidelines for practitioners. For researchers, we recommend that they perform more empirical studies on ML based SDEE and thus accumulate more evidence on the feasibility and performance of ML models. We also recommend that researchers develop a general framework for conducting experiments on ML models. As this review has revealed that extremely few case studies or surveys have focused on the application of ML model in industry, we therefore recommend that researchers conduct in-depth case studies or surveys in industry and figure out the barriers to the adoption of ML models in industry. For practitioners, we provide several guidelines for applying ML models in industry as follows: (1) understand

the rationale of ML models and cooperate with experienced researchers in the application of ML models; (2) choose the ML models whose strengths match well with the estimation contexts and goals; (3) give priority to within-company data sets when constructing ML models if the number of projects in the data sets is sufficient; (4) use ML models in parallel with existing conventional models at the early stage of the application of ML models; and (5) share proprietary project data sets with research community.

For future work, in addition to estimation accuracy metrics, other performance metrics such as generalization ability and interpretability can also be considered, so that ML models can be evaluated more completely. Moreover, it is advisable to consult the literature and experts in machine learning field to further verify the correctness of the summarized strengths and weaknesses of ML techniques in this review.

Acknowledgments

The authors thank Prof. Magne Jørgensen for sharing BESTweb bibliographic library. We would also like to thank the reviewers for their insightful comments. We thank Dr. Hai Liu for his valuable suggestions on the English presentation of this paper. This research was supported by the National Natural Science Foundation of China (Grant Nos.: 61003066, 70801020, and 60940033), the Natural Science Foundation of Guangdong Province in China (Grant No.: 10151063101000046), the Science and Technology Planning Project of Guangdong Province in China (Grant No.: 2010B010600034), the Foundation for Distinguished Young Talents in Higher Education of Guangdong Province in China (Grant Nos.: LYM08074 and LYM10097), and the "211 Project" of Guangdong University of Foreign Studies.

Appendix A

See Table A.8.

Appendix B

See Table B.9.

Appendix C

See Tables C.10, C.11, C.12 and C.13.

References

[1] M. Jørgensen, M. Shepperd, A systematic review of software development cost estimation studies, IEEE Transactions on Software Engineering 33 (1) (2007) 33–53.
[2] E. Mendes, I. Watson, C. Triggs, N. Mosley, S. Counsell, A comparative study of cost estimation models for web hypermedia applications, Empirical Software Engineering 8 (2) (2003) 163–196.
[3] I.F.B. Tronto, J.D.S. Silva, N.S. Anna, An investigation of artificial neural networks based prediction systems in software project management, Journal of Systems and Software 81 (3) (2008) 356–367.
[4] M.O. Elish, Improved estimation of software project effort using multiple additive regression trees, Expert Systems with Applications 36 (7) (2009) 10774–10778.
[5] B. Boehm, K. Sullivan, Software economics: status and prospects, Information and Software Technology 41 (14) (1999) 937–946.
[6] D. Zhang, J.J.P. Tsai, Machine learning and software engineering, Software Quality Journal 11 (2) (2003) 87–119.
[7] A.R. Gray, S.G. MacDonell, Software metrics data analysis – exploring the relative performance of some commonly used modeling techniques, Empirical Software Engineering 4 (4) (1999) 297–316.
[8] R. Jeffery, M. Ruhe, I. Wieczorek, A comparative study of two software development cost modeling techniques using multi-organizational and company-specific data, Information and Software Technology 42 (14) (2000) 1009–1016.
[9] A. Heiat, Comparison of artificial neural network and regression models for estimating software development effort, Information and Software Technology 44 (15) (2002) 911–922.
[10] J.J. Dolado, On the problem of the software cost function, Information and Software Technology 43 (1) (2001) 61–72.
[11] I. Myrtveit, E. Stensrud, A controlled experiment to assess the benefits of estimating with analogy and regression models, IEEE Transactions on Software Engineering 25 (4) (1999) 510–525.
[12] M. Shepperd, C. Schofield, Estimating software project effort using analogies, IEEE Transactions on Software Engineering 23 (11) (1997) 736–743.
[13] A.R. Gray, S.G. MacDonell, A comparison of techniques for developing predictive models of software metrics, Information and Software Technology 39 (6) (1997) 425–437.
[14] F. Walkerden, R. Jeffery, An empirical study of analogy-based software effort estimation, Empirical Software Engineering 4 (2) (1999) 135–158.
[15] L.C. Briand, K.E. Emam, D. Surmann, I. Wieczorek, K.D. Maxwell, An assessment and comparison of common software cost estimation modeling techniques, in: Proceedings of the 21st International Conference on Software Engineering, Los Angeles, CA, USA, 1999, pp. 313–322.
[16] E. Stensrud, Alternative approaches to effort prediction of ERP projects, Information and Software Technology 43 (7) (2001) 413–423.
[17] C. Mair, G. Kadoda, M. Lefley, K. Phalp, C. Schofield, M. Shepperd, S. Webster, An investigation of machine learning based prediction systems, Journal of Systems and Software 53 (1) (2000) 23–29.
[18] N.-H. Chiu, S.-J. Huang, The adjusted analogy-based software effort estimation based on similarity distances, Journal of Systems and Software 80 (4) (2007) 628–640.
[19] K. Moløkken-Østvold, M. Jørgensen, S.S. Tanilkan, H. Gallis, A.C. Lien, S.E. Hove, A survey on software estimation in the norwegian industry, in: Proceedings of the 10th International Symposium on Software Metrics, Chicago, Illinois, USA, 2004, pp. 208–219.
[20] M. Jørgensen, A review of studies on expert estimation of software development effort, Journal of Systems and Software 70 (1–2) (2004) 37–60.
[21] B.W. Boehm, Software engineering economics, IEEE Transactions on Software Engineering SE-10 (1) (1984) 4–21.
[22] F.J. Heemstra, Software cost estimation, Information and Software Technology 34 (10) (1992) 627–639.
[23] F. Walkerden, R. Jeffery, Software cost estimation: a review of models, process and practice, Advances in Computers 44 (1997) 59–125.
[24] B. Boehm, C. Abts, S. Chulani, Software development cost estimation approaches – a survey, Annals of Software Engineering 10 (2000) 177–205.
[25] L.C. Briand, I. Wieczorek, Resource estimation in software engineering, Tech. Rep. ISERN 00-05, International Software Engineering Research Network, 2000.
[26] M. Shepperd, Software project economics: a roadmap, in: Proceedings of the Conference on Future of Software Engineering, Minneapolis, MN, USA, 2007, pp. 304–315.
[27] M. Jørgensen, Forecasting of software development work effort: evidence on expert judgement and formal models, International Journal of Forecasting 23 (3) (2007) 449–462.
[28] C. Mair, M. Shepperd, The consistency of empirical comparisons of regression and analogy-based software project cost prediction, in: Proceedings of the International Symposium on Empirical Software Engineering, 2005, pp. 509–518.
[29] S. Grimstad, M. Jørgensen, K. Moløkken-Østvold, Software effort estimation terminology: the tower of babel, Information and Software Technology 48 (4) (2006) 302–310.
[30] B.A. Kitchenham, E. Mendes, G.H. Travassos, Cross versus within-company cost estimation studies: a systematic review, IEEE Transactions on Software Engineering 33 (5) (2007) 316–329.
[31] S.G. MacDonell, M.J. Shepperd, Comparing local and global software effort estimation models – reflections on a systematic review, in: Proceedings of the First International Symposium on Empirical Software Engineering and Measurement, Madrid, Spain, 2007, pp. 401–409.
[32] B.A. Kitchenham, S. Charters, Guidelines for performing systematic literature reviews in software engineering, Tech. Rep. EBSE-2007-01, Keele University and University of Durham, 2007.
[33] T.M. Mitchell, Machine Learning, McGraw-Hill, New York, 1997.
[34] E. Alpaydin, Introduction to Machine Learning, The MIT Press, Cambridge, Massachusetts, 2004.
[35] C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
[36] T. Mukhopadhyay, S.S. Vicinanza, M.J. Prietula, Examining the feasibility of a case-based reasoning model for software effort estimation, MIS Quarterly 16 (2) (1992) 155–171.
[37] L.C. Briand, V.R. Basili, W.M. Thomas, A pattern recognition approach for software engineering data analysis, IEEE Transactions on Software Engineering 18 (11) (1992) 931–942.
[38] J.P. Higgins, S. Green, Cochrane Handbook for Systematic Reviews of Interventions, Version 5.0.2 [updated September 2009], The Cochrane Collaboration, 2009, <www.cochrane-handbook.org>.
[39] T. Dybå, T. Dingsøyr, Empirical studies of agile software development: a systematic review, Information and Software Technology 50 (9–10) (2008) 833–859.
[40] D.I.K. Sjøeberg, J.E. Hannay, O. Hansen, V.B. Kampenes, A. Karahasanović, N.-K. Liborg, A.C. Rekdal, A survey of controlled experiments in software engineering, IEEE Transactions on Software Engineering 31 (9) (2005) 733–753.

[41] S.L. Pfleeger, Soup or art? The role of evidential force in empirical software engineering, IEEE Software 22 (1) (2005) 66–73.
[42] P. Brereton, B.A. Kitchenham, D. Budgen, M. Turner, M. Khalil, Lessons from applying the systematic literature review process within the software engineering domain, Journal of Systems and Software 80 (4) (2007) 571–583.
[43] B. Kitchenham, R. Pretorius, D. Budgen, O.P. Brereton, M. Turner, M. Niazi, S. Linkman, Systematic literature reviews in software engineering – a tertiary study, Information and Software Technology 52 (8) (2010) 792–805.
[44] L.M. Pickard, B.A. Kitchenham, P.W. Jones, Combining empirical results in software engineering, Information and Software Technology 40 (14) (1998) 811–821.
[45] G.W. Noblit, R.D. Hare, Meta-Ethnography: Synthesizing Qualitative Studies, Sage Publications, Inc., 1988.
[46] C. Wohlin, An analysis of the most cited articles in software engineering journals – 2002, Information and Software Technology 51 (1) (2009) 2–6.
[47] J.M. Desharnais, Analyse statistique de la productivite des projets de developpement en informatique a partir de la technique des points de fonction, Master's thesis, University of Montreal, 1989.
[48] B.W. Boehm, Software Engineering Economics, Prentice Hall PTR, New Jersey, 1981.
[49] International Software Benchmarking Standards Group (ISBSG), <http://www.isbsg.org>.
[50] A.J. Albrecht, J.E. Gaffney, Software function, source lines of code, and development effort prediction: a software science validation, IEEE Transactions on Software Engineering 9 (6) (1983) 639–648.
[51] C.F. Kemerer, An empirical validation of software cost estimation models, Communications of the ACM 30 (5) (1987) 416–429.
[52] J.W. Bailey, V.R. Basili, A meta-model for software development resource expenditures, in: Proceedings of the 5th International Conference on Software Engineering, San Diego, California, USA, 1981, pp. 107–116.
[53] E. Mendes, N. Mosley, S. Counsell, Investigating web size metrics for early web cost estimation, Journal of Systems and Software 77 (2) (2005) 157–172.
[54] G. Zhang, B.E. Patuwo, M.Y. Hu, Forecasting with artificial neural networks: the state of the art, International Journal of Forecasting 14 (1) (1998) 35–62.
[55] A.J. Albrecht, Measuring application development productivity, in: Proceedings of the Joint SHARE/GUIDE/IBM Application Development Symposium, Monterey, CA, USA, 1979, pp. 83–92.
[56] L.H. Putnam, A general empirical solution to the macro software sizing and estimating problem, IEEE Transactions on Software Engineering SE-4 (4) (1978) 345–361.
[57] S. Bibi, I. Stamelos, Selecting the appropriate machine learning techniques for the prediction of software development costs, in: I. Maglogiannis, K. Karpouzis, M. Bramer (Eds.), Artificial Intelligence Applications and Innovations, vol. 204, Springer, Boston, 2006, pp. 533–540.
[58] S.G. MacDonell, M.J. Shepperd, Combining techniques to optimize effort predictions in software project management, Journal of Systems and Software 66 (2) (2003) 91–98.
[59] K.K. Shukla, Neuro-genetic prediction of software development effort, Information and Software Technology 42 (10) (2000) 701–713.
[60] S.-J. Huang, N.-H. Chiu, Optimization of analogy weights by genetic algorithm for software effort estimation, Information and Software Technology 48 (11) (2006) 1034–1045.
[61] S. Bibi, I. Stamelos, L. Angelis, Combining probabilistic models for explanatory productivity estimation, Information and Software Technology 50 (7–8) (2008) 656–669.
[62] A.L.I. Oliveira, P.L. Braga, R.M.F. Lima, M.L. Cornélio, GA-based method for feature selection and parameters optimization for machine learning regression applied to software effort estimation, Information and Software Technology 52 (11) (2010) 1155–1166.
[63] M. Shepperd, G. Kadoda, Comparing software prediction techniques using simulation, IEEE Transactions on Software Engineering 27 (11) (2001) 1014–1022.
[64] G. Boetticher, T. Menzies, T. Ostrand, PROMISE Repository of Empirical Software Engineering Data, West Virginia University, Department of Computer Science, 2007. <http://promisedata.org/>.
[65] R. Jeffery, M. Ruhe, I. Wieczorek, Using public domain metrics to estimate software development effort, in: Proceedings of the 7th International Software Metrics Symposium, London, UK, 2001, pp. 16–27.
[66] B.A. Kitchenham, E. Mendes, A comparison of cross-company and within-company effort estimation models for web applications, in: Proceedings of the 8th International Conference on Empirical Assessment in Software Engineering, Edinburgh, Scotland, UK, 2004, pp. 47–55.
[67] E. Mendes, B. Kitchenham, Further comparison of cross-company and within-company effort estimation models for web applications, in: Proceedings of the 10th International Symposium on Software Metrics Symposium, Chicago, IL, USA, 2004, pp. 348–357.
[68] I. Myrtveit, E. Stensrud, M. Shepperd, Reliability and validity in comparative studies of software prediction models, IEEE Transactions on Software Engineering 31 (5) (2005) 380–391.
[69] L. Angelis, I. Stamelos, A simulation tool for efficient analogy based cost estimation, Empirical Software Engineering 5 (1) (2000) 35–68.
[70] R.A. Araújo, A.L.I. Oliveira, S. Soares, Hybrid intelligent design of morphological-rank-linear perceptrons for software development cost estimation, in: Proceedings of the 22nd International Conference on Tools with Artificial Intelligence, Arras, France, 2010, pp. 160–167.
[71] M. Auer, A. Trendowicz, B. Graser, E. Haunschmid, S. Biffl, Optimal project feature weights in analogy-based cost estimation: improvement and limitations, IEEE Transactions on Software Engineering 32 (2) (2006) 83–92.
[72] M. Azzeh, D. Neagu, P. Cowling, Software project similarity measurement based on fuzzy C-means, in: Proceedings of the International Conference on Software Process, Leipzig, Germany, 2008, pp. 123–134.
[73] M. Azzeh, D. Neagu, P. Cowling, Improving analogy software effort estimation using fuzzy feature subset selection algorithm, in: Proceedings of the 4th International Workshop on Predictor Models in Software Engineering, Leipzig, Germany, 2008, pp. 71–78.
[74] M. Azzeh, D. Neagu, P. Cowling, Software effort estimation based on weighted fuzzy grey relational analysis, in: Proceedings of the 5th International Conference on Predictor Models in Software Engineering, Vancouver, British Columbia, Canada, 2009, pp. 1–10.
[75] S. Berlin, T. Raz, C. Glezer, M. Zviran, Comparison of estimation methods of cost and duration in IT projects, Information and Software Technology 51 (4) (2009) 738–748.
[76] S. Bibi, I. Stamelos, G. Gerolimos, V. Kollias, BBN based approach for improving the software development process of an SME – a case study, Journal of Software Maintenance and Evolution: Research and Practice 22 (2) (2010) 121–140.
[77] P.L. Braga, A.L.I. Oliveira, S.R.L. Meira, Software effort estimation using machine learning techniques with robust confidence intervals, in: Proceedings of the 19th International Conference on Tools with Artificial Intelligence, vol. 1, Patras, Greece, 2007, pp. 181–185.
[78] P.L. Braga, A.L.I. Oliveira, G.H.T. Ribeiro, S.R.L. Meira, Bagging predictors for estimation of software project effort, in: Proceedings of the International Joint Conference on Neural Networks, Orlando, Florida, USA, 2007, pp. 1595–1600.
[79] L.C. Briand, T. Langley, I. Wieczorek, A replicated assessment and comparison of common software cost modeling techniques, in: Proceedings of the 22nd International Conference on Software Engineering, Limerick, Ireland, 2000, pp. 377–386.
[80] L.C. Briand, J. Wüst, Modeling development effort in object-oriented systems using design properties, IEEE Transactions on Software Engineering 27 (11) (2001) 963–986.
[81] C.J. Burgess, M. Lefley, Can genetic programming improve software effort estimation? a comparative evaluation, Information and Software Technology 43 (14) (2001) 863–873.
[82] A. Corazza, S.D. Martino, F. Ferrucci, C. Gravino, E. Mendes, Investigating the use of support vector regression for web effort estimation, Empirical Software Engineering (2010) 1–33.
[83] A. Corazza, S.D. Martino, F. Ferrucci, C. Gravino, F. Sarro, E. Mendes, How effective is tabu search to configure support vector regression for effort estimation?, in: Proceedings of the 6th International Conference on Predictor Models in Software Engineering, Timisoara, Romania, 2010, pp. 1–10.
[84] G. Costagliola, S.D. Martino, F. Ferrucci, C. Gravino, G. Tortora, G. Vitiello, Effort estimation modeling techniques: a case study for web applications, in: Proceedings of the 6th International Conference on Web Engineering, Palo Alto, CA, United States, 2006, pp. 9–16.
[85] F. Ferrucci, C. Gravino, R. Oliveto, F. Sarro, Genetic programming for effort estimation: an analysis of the impact of different fitness functions, in: Proceedings of the 2nd International Symposium on Search Based Software Engineering, Benevento, Italy, 2010, pp. 89–98.
[86] G.R. Finnie, G.E. Wittig, J.-M. 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 (3) (1997) 281–289.
[87] S.-J. Huang, N.-H. Chiu, Applying fuzzy neural network to estimate software development effort, Applied Intelligence 30 (2) (2009) 73–83.
[88] S.-J. Huang, N.-H. Chiu, L.-W. Chen, Integration of the grey relational analysis with genetic algorithm for software effort estimation, European Journal of Operational Research 188 (3) (2008) 898–909.
[89] S.-J. Huang, C.-Y. Lin, N.-H. Chiu, Fuzzy decision tree approach for embedding risk assessment information into software cost estimation model, Journal of Information Science and Engineering 22 (2) (2006) 297–313.
[90] X. Huang, D. Ho, J. Ren, L.F. Capretz, Improving the COCOMO model using a neuro-fuzzy approach, Applied Soft Computing 7 (1) (2007) 29–40.
[91] A. Idri, A. Abran, T.M. Khoshgoftaar, Fuzzy case-based reasoning models for software cost estimation, in: E. Damiani, L.C. Jain, M. Madravio (Eds.), Soft Computing in Software Engineering, Springer, 2004, pp. 64–96.
[92] A. Idri, A. Zahi, E. Mendes, A. Zakrani, Software cost estimation models using radial basis function neural networks, in: Proceedings of the International Workshop on Software Measurement, Palma de Mallorca, Spain, 2007, pp. 21–31.
[93] M. Jørgensen, U. Indahl, D. Sjøberg, Software effort estimation by analogy and "regression toward the mean", Journal of Systems and Software 68 (3) (2003) 253–262.
[94] E.S. Jun, J.K. Lee, Quasi-optimal case-selective neural network model for software effort estimation, Expert Systems with Applications 21 (1) (2001) 1–14.
[95] G. Kadoda, M. Cartwright, L. Chen, M. Shepperd, Experiences using case-based reasoning to predict software project effort, in: Proceedings of the Conference on Evaluation and Assessment in Software Engineering, Keele University, UK, 2000.

[96] J.W. Keung, B.A. Kitchenham, D.R. Jeffery, Analogy-X: providing statistical inference to analogy-based software cost estimation, IEEE Transactions on Software Engineering 34 (4) (2008) 471–484.
[97] Y. Kultur, B. Turhan, A. Bener, Ensemble of neural networks with associative memory (ENNA) for estimating software development costs, Knowledge-Based Systems 22 (6) (2009) 395–402.
[98] K.V. Kumar, V. Ravi, M. Carr, N.R. Kiran, Software development cost estimation using wavelet neural networks, Journal of Systems and Software 81 (11) (2008) 1853–1867.
[99] T.K. Le-Do, K.-A. Yoon, Y.-S. Seo, D.-H. Bae, Filtering of inconsistent software project data for analogy-based effort estimation, in: Proceedings of the 34th Annual IEEE Computer Software and Applications Conference, Seoul, South Korea, 2010, pp. 503–508.
[100] A. Lee, C.H. Cheng, J. Balakrishnan, Software development cost estimation: integrating neural network with cluster analysis, Information and Management 34 (1) (1998) 1–9.
[101] J. Li, A. Al-Emran, G. Ruhe, Impact analysis of missing values on the prediction accuracy of analogy-based software effort estimation method AQUA, in: Proceedings of the 1st International Symposium on Empirical Software Engineering and Measurement, Madrid, Spain, 2007, pp. 126–135.
[102] J. Li, G. Ruhe, Analysis of attribute weighting heuristics for analogy-based software effort estimation method AQUA+, Empirical Software Engineering 13 (1) (2008) 63–96.
[103] J. Li, G. Ruhe, A. Al-Emran, M.M. Richter, A flexible method for software effort estimation by analogy, Empirical Software Engineering 12 (1) (2007) 65–106.
[104] Y.F. Li, M. Xie, T.N. Goh, A study of mutual information based feature selection for case based reasoning in software cost estimation, Expert Systems with Applications 36 (3) (2009) 5921–5931.
[105] Y.F. Li, M. Xie, T.N. Goh, A study of project selection and feature weighting for analogy based software cost estimation, Journal of Systems and Software 82 (2) (2009) 241–252.
[106] Y.F. Li, M. Xie, T.N. Goh, A study of the non-linear adjustment for analogy based software cost estimation, Empirical Software Engineering 14 (6) (2009) 603–643.
[107] E. Mendes, A comparison of techniques for web effort estimation, in: Proceedings of the 1st International Symposium on Empirical Software Engineering and Measurement, Madrid, Spain, 2007, pp. 334–343.
[108] E. Mendes, S.D. Martino, F. Ferrucci, C. Gravino, Cross-company vs. single-company web effort models using the tukutuku database: an extended study, Journal of Systems and Software 81 (5) (2008) 673–690.
[109] E. Mendes, N. Mosley, Further investigation into the use of CBR and stepwise regression to predict development effort for web hypermedia applications, in: Proceedings of the International Symposium on Empirical Software Engineering, Nara, Japan, 2002, pp. 79–90.
[110] E. Mendes, N. Mosley, Bayesian network models for web effort prediction: a comparative study, IEEE Transactions on Software Engineering 34 (6) (2008) 723–737.
[111] N. Mittas, L. Angelis, LSEbA: Least squares regression and estimation by analogy in a semi-parametric model for software cost estimation, Empirical Software Engineering 15 (5) (2010) 523–555.
[112] N. Mittas, M. Athanasiades, L. Angelis, Improving analogy-based software cost estimation by a resampling method, Information and Software Technology 50 (3) (2008) 221–230.
[113] A.L. Oliveira, Estimation of software project effort with support vector regression, Neurocomputing 69 (13–15) (2006) 1749–1753.
[114] H. Park, S. Baek, An empirical validation of a neural network model for software effort estimation, Expert Systems with Applications 35 (3) (2008) 929–937.
[115] P.C. Pendharkar, G.H. Subramanian, J.A. Rodger, A probabilistic model for predicting software development effort, IEEE Transactions on Software Engineering 31 (7) (2005) 615–624.
[116] B. Samson, D. Ellison, P. Dugard, Software cost estimation using an Albus perceptron (CMAC), Information and Software Technology 39 (1) (1997) 55–60.
[117] Y.-S. Seo, K.-A. Yoon, D.-H. Bae, An empirical analysis of software effort estimation with outlier elimination, in: Proceedings of the 4th International Workshop on Predictor Models in Software Engineering, Leipzig, Germany, 2008, pp. 25–32.
[118] R. Setiono, K. Dejaeger, W. Verbeke, D. Martens, B. Baesens, Software effort prediction using regression rule extraction from neural networks, in: Proceedings of the 22nd International Conference on Tools with Artificial Intelligence, vol. 2, Arras, France, 2010, pp. 45–52.
[119] M. Shin, A.L. Goel, Empirical data modeling in software engineering using radial basis functions, IEEE Transactions on Software Engineering 26 (6) (2000) 567–576.
[120] K. Srinivasan, D. Fisher, Machine learning approaches to estimating software development effort, IEEE Transactions on Software Engineering 21 (2) (1995) 126–137.
[121] I. Stamelos, L. Angelis, P. Dimou, E. Sakellaris, On the use of bayesian belief networks for the prediction of software productivity, Information and Software Technology 45 (1) (2003) 51–60.
[122] E. Stensrud, I. Myrtveit, Human performance estimating with analogy and regression models: an empirical validation, in: Proceedings of the 5th International Software Metrics Symposium, 1998, pp. 205–213.
[123] B. Stewart, Predicting project delivery rates using the naive-bayes classifier, Journal of Software Maintenance and Evolution: Research and Practice 14 (3) (2002) 161–179.
[124] J. Wen, S. Li, L. Tang, Improve analogy-based software effort estimation using principal components analysis and correlation weighting, in: Proceedings of the 16th Asia-Pacific Software Engineering Conference, 2009, pp. 179–186.
[125] I. Wieczorek, M. Ruhe, How valuable is company-specific data compared to multi-company data for software cost estimation?, in: Proceedings of the 8th International Software Metrics Symposium, Ottawa, Canada, 2002, pp. 237–246.
[126] G. Wittig, G. Finnie, Estimating software development effort with connectionist models, Information and Software Technology 39 (7) (1997) 469–476.
[127] G.E. Wittig, G.R. Finnie, Using artificial neural networks and function points to estimate 4GL software development effort, Australian Journal of Information Systems 1 (2) (1994) 87–94.